Check for updates

Article

The International Journal of Robotics Research
2016, Vol. 35(1–3) 244–264
© The Author(s) 2015
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364915594474
ijr.sagepub.com

Pre- and post-contact policy decomposition for planar contact manipulation under uncertainty

Michael C. Koval, Nancy S. Pollard and Siddhartha S. Srinivasa

Abstract

We consider the problem of using real-time feedback from contact sensors to create closed-loop pushing actions. To do so, we formulate the problem as a partially observable Markov decision process (POMDP) with a transition model based on a physics simulator and a reward function that drives the robot towards a successful grasp. We demonstrate that it is intractable to solve the full POMDP with traditional techniques and introduce a novel decomposition of the policy into pre- and post-contact stages to reduce the computational complexity.

Our method uses an offline point-based solver on a variable-resolution discretization of the state space to solve for a post-contact policy as a pre-computation step. Then, at runtime, we use an A search to compute a pre-contact trajectory. We prove that the value of the resulting policy is within a bound of the value of the optimal policy and give intuition about when it performs well. Additionally, we show the policy produced by our algorithm achieves a successful grasp more quickly and with higher probability than a baseline QMDP policy on two different objects in simulation. Finally, we validate our simulation results on a real robot using commercially available tactile sensors.*

Keywords

Tactile sensing, planning under uncertainty, manipulation under uncertainty, non-prehensile manipulation

1. Introduction

Humans effortlessly manipulate objects by leveraging their sense of touch, as demonstrated when a person feels around on a nightstand for a glass of water or in a cluttered kitchen cabinet for a salt-shaker. In each of these tasks the person makes *persistent contact* with the environment and uses their tactile sense for real-time feedback. Robotic manipulators should be able to similarly use contact sensors to achieve this kind of dexterity. In this paper, we present a strategy for generating a robust policy for contact manipulation that takes advantage of tactile feedback.

Contact manipulation is an inherently noisy process: a robot perceives its environment with imperfect sensors, has uncertain kinematics, and uses simplified models of physics to predict the outcome of its actions. Recent work (Section 2) has formulated manipulation as a *partially observable Markov decision process* (POMDP) (Kaelbling et al., 1998) with a reward function that drives the robot towards the goal (Horowitz and Burdick, 2013; Hsiao, 2009; Hsiao et al., 2007; Platt et al., 2011). Unfortunately, the *contact manipulation POMDP* is intractable for most real-world problems, like Figure 1, where a robot hand manipulates an object into its hand with a closed-loop tactile policy.

Our key insight is that the optimal policy for the contact manipulation POMDP naturally decomposes into two stages: (1) an open-loop *pre-contact trajectory* that terminates when contact is achieved 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 (Hwang and Ahuja, 1992) found in early manipulation research.

We can accurately detect the transition from pre- to post-contact because contact sensors *discriminate* whether or not contact has occurred (Koval et al., 2013a). As a result, any contact observation indicates that the object lies on the *contact manifold* (Hauser and Ng-Thow-Hing, 2011; Koval et al., 2013a; Lozano-Pèrez, 1983), the lower-dimensional set of poses in which the object is in non-penetrating contact with the hand.

The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA

Corresponding author:
Michael C. Koval, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA.
Email address: mkoval@cs.cmu.edu

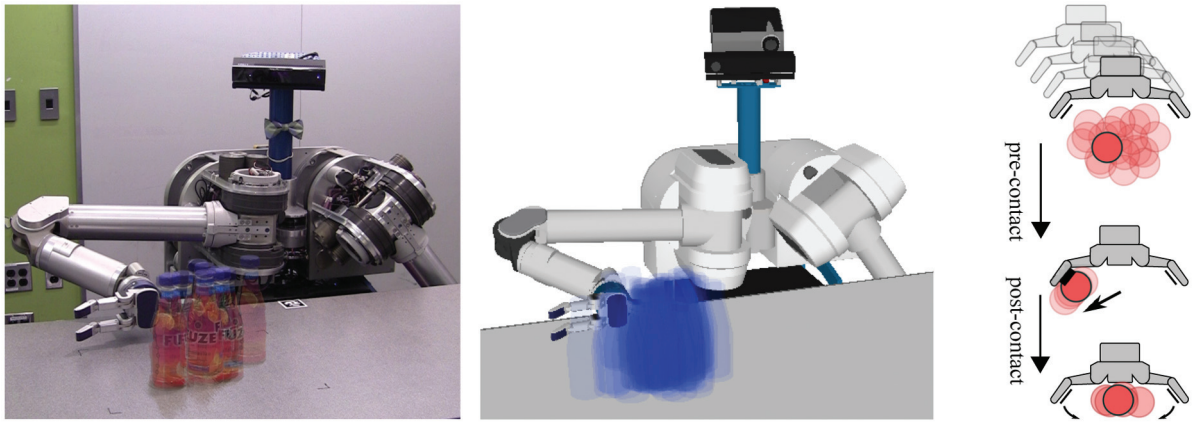


Fig. 1. HERB (left) using real-time feedback from contact sensors to grasp a bottle under pose uncertainty. During execution, the robot tracks the pose of the object using a Bayesian filter (middle). The left photo shows the 10 trials that were actually executed in Section 8. HERB begins by executing an open-loop trajectory (right-top) towards the object. Once contact is observed (right-middle), HERB switches to executing a closed-loop policy (right-bottom) to complete the grasp.

We exploit this structure to find a near-optimal policy for the contact manipulation POMDP. First, as an offline pre-computation step, we find a post-contact policy using Successive Approximations of the Reachable Space under Optimal Policies (SARSOP) (Kurniawati et al., 2008), a point-based POMDP solver. 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 A* search to plan a pre-contact trajectory that aims to make contact with the object.

In this paper, we specifically consider the task of pushing objects on a planar support surface (Figure 1) with binary contact sensors for feedback. This problem is a fundamental research topic in manipulation (Howe and Cutkosky, 1996; Lynch and Mason, 1996; Mason, 1986) and enables robots to perform a wide variety of tasks that would not otherwise be possible. Pushing enables robots to move objects that are too large or heavy to be grasped (Dogar and Srinivasa, 2011), for pre-grasp manipulation (Chang et al., 2010; Kappler et al., 2010), and to grasp objects under uncertainty (Brost, 1988; Dogar and Srinivasa, 2010; Dogar et al., 2012).

We build on this large body of work by developing a closed-loop pushing action that is robust to large amounts of uncertainty. We show, through a large suite of simulation experiments, that our uncertainty-aware policy outperforms a baseline QMDP policy (Littman et al., 1995) that makes use of real-time feedback. The QMDP policy is efficient to compute, but assumes that all uncertainty disappears after one timestep and, thus, does not plan multi-step information gathering actions. We also demonstrate that these policies work on HERB, a robot designed and built by the Personal Robotics Lab at Carnegie Mellon University (Srinivasa et al., 2012).

We make the following contributions:

- Policy decomposition. We introduce a novel algorithm that exploits the structure of contact manipulation to efficiently find a provably near-optimal policy (Section 4). Our key insight is that the optimal policy for the contact manipulation POMDP naturally decomposes into an open-loop move-until-touch trajectory followed by a closed-loop policy.
- Post-contact policy. We present a method of finding a post-contact policy (Section 5) using a point-based POMDP solver. Finding a solution is efficient because we explicitly discretize the contact manifold to accurately represent the object's interaction with the hand.
- Simulation results. We demonstrate that the proposed algorithm successfully grasps an object in simulation experiments (Section 7.4) on two different objects. Our uncertainty-aware policy achieves a successful grasp more quickly and with higher probability than the baseline QMDP policy.
- Real-robot experiments. We show results from real-robot experiments on HERB (Section 8) using strain gages for real-time tactile feedback. These results confirm that the policies produced by our algorithm perform well when executed on real hardware.

We also discuss several limitations of our work. Key among them is the requirement that the post-contact POMDP must be discretized to pre-compute the post-contact policy. This is possible for local, planar policies (such as grasping an object) but precludes policies that require large, global movement. For example, our algorithm cannot efficiently generate policies for problems that require long transit phases or coordinating two distant end-effectors.

This paper is an improved and extended version of our prior work (Koval et al., 2014). These improvements include a more detailed description of our method of finding the post-contact policy (Section 5) and discussion of our results (Section 9). We present new simulation results for a disk object (Section 7) and analyze the effect of

sensor coverage on performance (Section 7.5). Finally, we present real-robot experiments (Section 8) on HERB.

2. Related work

This paper builds prior work in pushing 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 on motion planning under uncertainty (Section 2.4), we formulate manipulation under uncertainty 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). More recently, the same approach has been applied to the problems of grasping (Dogar and Srinivasa, 2010), and rearrangement planning (Dogar and Srinivasa, 2012) 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 algorithm also uses the quasistatic assumption and, in some cases, generates uncertainty-reducing actions (see Section 8) 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.

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 observations 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 (Li et al., 2013; Zhang and Chen, 2000). 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 algorithm 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 contact sensors to predict grasp stability (Dang et al., 2011) and object identity (Schneider et al., 2009; Xu et al., 2013).

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 a particle filter 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 (Gadeyne et al., 2005; Meeussen et al., 2007). We use a particle filter, similar to these, to track our belief state during execution of the pre-contact trajectory.

Our own prior work (Koval et al., 2013a) introduced the manifold particle filter for object pose estimation using contact sensors. The manifold particle filter explicitly samples particles from an approximate representation of the contact manifold to avoid particle starvation during contact. We use the same analytic representation of the contact manifold (Koval et al., 2013b) as used by the manifold particle filter to efficiently discretize the state space explored by the post-contact policy. However, unlike passive state estimation, our algorithm generates 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 (Hyafil and Bacchus, 2003; Smith and Weld, 1998), and hybrid task planners (Kaelbling and Lozano-Pérez, 2013). In this paper, we specifically consider the problem of low-level policy generation.

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. 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 replanning steps (Platt et al., 2012). Unfortunately, these 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 paper, we specifically consider planning for manipulation tasks using feedback from contact sensors. Physics-based manipulation under uncertainty is particularly challenging because the state and actions spaces are continuous, evaluating a physics model is computationally expensive, and the observations generated by contact sensors are inherently discontinuous.

Early work on robotic assembly used feedback from force sensors to perform fine manipulation (Simunovic, 1979). A common strategy is to use guarded moves, that is, 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 (Hebert et al., 2013; Javdani et al., 2013; Petrovskaya and Khatib, 2011), where the robot plans a sequence of guarded moves 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). Our technique entirely eliminates 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 formulating 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 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. We adapt a similar approach to finding the post-contact policy by explicitly discretizing an analytic representation of the contact manifold (Koval et al., 2013b). We generalize this approach to a wider class of contact manipulation problems, such as those those with long planning horizons, by decomposing the policy into two stages.

3. Contact manipulation problem

We focus on the class of *contact manipulation* tasks where a robotic manipulator maintains persistent contact with its environment, for example pushing an object to a desired pose or executing a grasp. Unfortunately, contact manipulation is inherently uncertain: a robot perceives its environment with noisy sensors, has uncertain kinematics, and uses simplified models of physics for reasoning about the consequences of its actions. Thus, incorporating and even seeking out new information during execution is often critical for success.

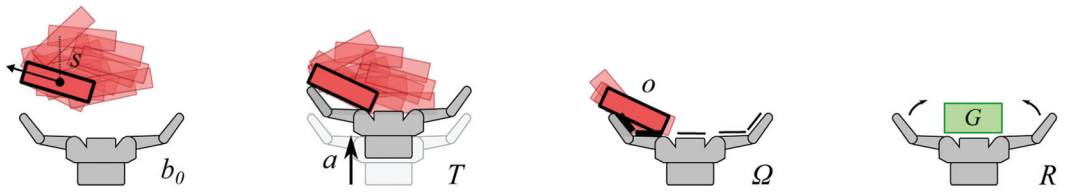
3.1. POMDP formulation

We formulate the contact manipulation problem as a POMDP with continuous state, but discrete action and observation spaces. A POMDP is a tuple (S, A, O, T, Ω, R) where S is the set of states, A is the set of actions, O is the set of observations, $T(s, a, s') = p(s'|s, a)$ is the transition model, $\Omega(o, s, a) = p(o|s, a)$ is the observation model, and $R : S \times A \rightarrow \mathbb{R}$ is the reward function (Kaelbling et al., 1998).

In a POMDP the agent does not know its true state but instead tracks its belief state $b : S \rightarrow [0, 1]$ with $\int_S b(s)\, ds = 1$, a distribution over S, with a state estimator. The set of all belief states $\Delta = \{b : S \rightarrow [0, 1] : \int_S b(s)\, ds = 1\}$ is known as *belief space*. The goal is to find a policy $\pi : \Delta \rightarrow A$ over belief space that maximizes the sum of expected future reward $E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$ discounted by $\gamma \in [0,1)$.

We consider the planar contact manipulation problem (Figure 2) where a state $s \in S = SE(2)$ is the pose of the object relative to the hand. An action $a = (v_a, T_a) \in A$ commands the hand to follow the generalized velocity $v_a \in se(2)$ for T_a seconds, possibly making contact with the object.

During contact the motion of the object is modeled by a quasistatic physics simulator (Lynch et al., 1992). The *quasistatic assumption* states that an object will stop moving as soon as it leaves contact with the hand. As a result, the state space consists only of pose $S = SE(2)$ instead of the tangent bundle $S = SE(2) \times se(2)$, halving the dimensionality of



(a) Belief state (b) Action (c) Observation (d) Reward

Fig. 2. Contact manipulation POMDP. The robot starts in (a) belief state $b_0 \in \Delta$, (b) takes action $a \in A$ following transition model T, and (c) updates its belief state with observation $o \in O$ generated by the observation model Ω. The robot's goal is to maximize its (d) reward R by pushing the object into the goal region $G \subseteq S$.

the state space. This approximation has been shown to be accurate for the planar manipulation of household objects at relatively low speeds (Dogar and Srinivasa, 2010, 2011).

The transition model $T(s, a, s')$ can additionally incorporate state-action-dependent uncertainty. This includes inaccuracies in the physics simulator and the unknown physical properties of the environment (e.g. friction coefficients). We can also model noise in the motion of the hand while executing actions, assuming that the noise is independent of the full configuration of the manipulator. We do, however, assume that the geometry of the hand and object is known.

After taking action a, the robot receives an observation $o \in O$ that indicates whether the object is touching a contact sensor. This is equivalent to testing whether $s \in S_o$, where $S_o \subset S$ is the *observable contact manifold*: the set of all states that are in non-penetrating contact with one or more sensors (Koval et al., 2013b). Similar to prior work (Hebert et al., 2013; Javdani et al., 2013; Koval et al., 2013a,b; Petrovskaya and Khatib, 2011), we assume that observations perfectly discriminate between contact ($o \in O_c$) and no-contact ($o = o_{nc}$), but may not perfectly localize the object along the hand. For example, a binary contact sensor that returns "contact" or "no-contact" for the entire hand, but provides no additional information about the pose of the object, satisfies this assumption.

3.2. Reward function

For the remainder of this paper, we assume that the robot starts with a prior belief $b_0 \in \Delta$ (possibly initialized with vision or knowledge of the environment) and wishes to quickly push the object into a hand-relative goal region $G \subseteq S$ as quickly as possible. This goal corresponds to an objective function that minimizes the expected time $E[T_{a_1} + \cdots + T_{a_n}]$ required to reach a belief $b(s_t)$ that satisfies $\int_G b(s_t)\, ds_t \geq 1 - \epsilon$. Once the goal has been achieved, we can execute a grasp with a high probability of success.

This objective cannot be written as a state-action-dependent reward function $R(s, a)$. Instead, we choose a reward function of the form

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

that assigns zero reward to the goal and negative reward to all actions. This encourages the robot to drive the object into the goal region. This reward function trades off between reaching G quickly with low probability and taking a long time to reach G with high probability.

For the remainder of the paper, we assume that the objective is to optimize infinite horizon reward, in other words, there is no termination condition. In most applications, this means that an external observer is responsible for deciding when to terminate execution, for example when $\Pr(s_t \in G)$ is sufficiently high. We discuss integrating this type of termination condition into the reward function in Section 9.4.

3.3. Value function

Given a reward function, each policy π induces a *value function* $V^\pi : \Delta \to \mathbb{R}$ that is equal to the sum of expected future reward of following policy π in belief state b. The value function V^* of the optimal policy π^* is a fixed point of the Bellman equation

$$V^*(b) = \max_{a \in A} \left[R(b,a) + \gamma \int_\Delta T(b,a,b') \sum_{o \in O} \Omega(o,b',a) V^*(b')\, db' \right]$$

where $R(b,a) = \sum_{s \in S} R(s,a) b(s)$ is the expected reward of executing action a in belief state b (Bellman, 1957). The summation over O computes an expected value over future, and thus unknown, observations.

3.4. Tractability

Optimally solving a discrete POMDP has been shown to be PSPACE-complete with a finite horizon (Littman, 1996) and undecidable with an infinite horizon (Madani et al., 1999). As a result, finding the optimal policy for a POMDP is only tractable for small problems. Instead, we must rely on approximate techniques.

Point-based methods, first introduced by Pineau et al. (2003), are a class of offline solvers that break the *curse of history* by performing backups at a discrete set of *belief points*. 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 an arbitrary sequence of actions and observations, is small.

With few exceptions (Brunskill et al., 2008; Porta et al., 2006), point-based methods are restricted discrete POMDPs. The contact manipulation POMDP has continuous state and action spaces that must be discretized. Discretizing a 1 m × 1 m region at a 2 cm × 2 cm × 12° resolution, which is the same resolution used in our experimental results (Section 7), would result in an enormous state space of size $|S| = 75{,}000$. Solving this POMDP is at the edge of feasibility for modern point-based methods, for example SARSOP (Kurniawati et al., 2008) and PGVI (Zhang et al., 2014). Finding an adequate solution is likely possible, but may be time-consuming.

More importantly, the policy computed by a point-based method is only guaranteed to perform well for the initial belief state b_0. Generalizing to a new problem instance requires re-planning from the new belief state. This may be costly and discards the primary advantage of offline planning: the ability to quickly apply a pre-computed policy.

Online planning algorithms (Ross et al., 2008) forgo pre-computation entirely finding a local policy during each step of execution. Actions are selected by performing a forward-search of the action-observation tree rooted at the current belief state. Online planning algorithms can operate in continuous state spaces and perform well when upper/lower bounds or a heuristic is available to guide the search and ample time is available for action selection. Recent online solvers, like POMCP (Silver and Veness, 2010), ABT (Kurniawati and Yadav, 2013), and DESPOT (Somani et al., 2013), have shown impressive performance in large, continuous state spaces. Of these, ABT has also been extended to continuous action spaces (Seiler et al., 2015).

Unfortunately, performing this search online is challenging given the real-time constraints on the contact manipulation problem. Simply performing a Bayesian update on the continuous belief state, which is a fundamental operation of an online planner, requires running a large number of computationally expensive physics simulations and is challenging to perform in real time (Koval et al., 2013a; Zhang and Trinkle, 2012).

In the next section, we present an algorithm that combines the advantages of both approaches. We use a point-based method to pre-compute a general post-contact policy and use an online solver at runtime to adapt to changes in the initial belief state.

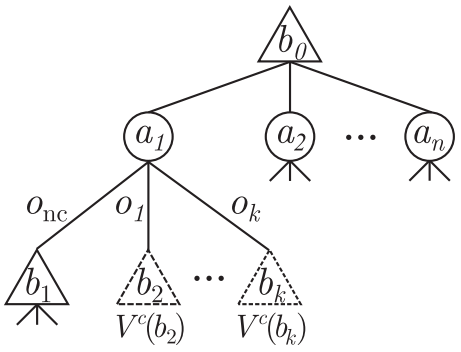


Fig. 3. Online POMDP solvers must branch over both actions and observations. The pre-contact search only branches on actions by evaluating all post-contact belief states with the post-contact value function V^c.

4. Policy decomposition

Our key observation is that a policy for the contact manipulation POMDP is naturally split into pre- and post-contact stages due to the discriminative nature of contact sensors. Before observing contact, the robot executes an open-loop *pre-contact trajectory* $\xi \in A \times A \times \ldots$ and receives a series of no-contact observations $o_1 = \ldots = o_{t-1} = o_{\text{nc}}$. 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.

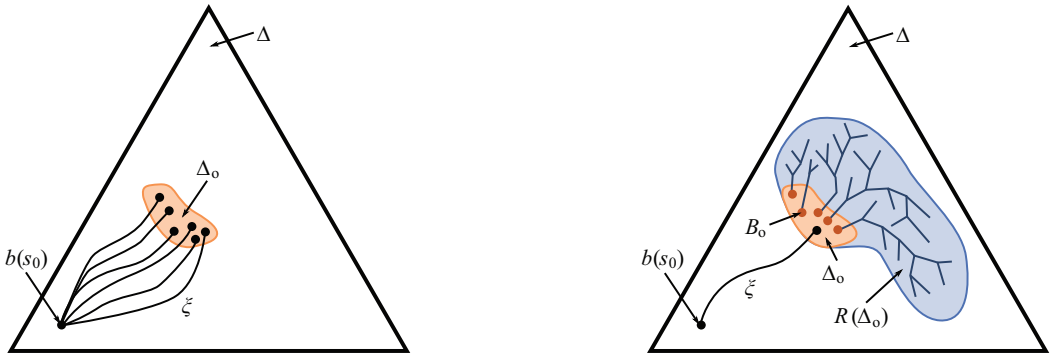
Decomposing the policy into pre- and post-contact stages is equivalent to splitting the value function

$$V(b) = \max_{a \in A} \left[R(b,a) + \gamma \int_{\Delta} T(b,a,b') \left(\underbrace{\Omega(o_{\text{nc}}, b', a) V(b')}_{\text{pre-contact}} + \underbrace{\sum_{o \in O_{\text{c}}} \Omega(o, b', a) V^{\text{c}}(b')}_{\text{pre-contact}} \right) db' \right] \tag{1}$$

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 ($o = o_{\text{nc}}$). As we describe in Section 5, we know that all post-contact belief states lie in a small region of belief space.

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 $o \in O_{\text{c}} = O \setminus \{o_{\text{nc}}\}$ 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 5). Our intuition, as shown in Figure 4(b), is that exhaustively computing this policy is tractable because



(a) Pre-contact trajectory (b) Post-contact policy

Fig. 4. We decompose the policy π into a (a) pre-contact trajectory ξ and a (b) post-contact policy π^{c}. The two stages are coupled by the small set of post-contact belief states $\Delta_o \subseteq \Delta$. Therefore, we can efficiently compute a policy over the set of reachable post-contact beliefs $\mathcal{R}(\Delta_o)$ using a point-based POMDP solver.

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 (Figure 4(a)). As shown in Figure 3 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.

4.1. 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 T-horizon policy V_T in terms of the value function V_{T-1} of the $(T-1)$-horizon policy:

$$
\begin{aligned}
||V_T - V^{*}||_{\infty} &\leq \gamma ||V_{T-1} - V^{*}||_{\infty} + \gamma P_{\max} ||V^{c} - V^{*}||_{\infty} \\
&\leq \gamma^{T} ||V_0 - V^{*}||_{\infty} + \sum_{t=1}^{T} \gamma^{t} P_{\max} ||V^{c} - V^{*}||_{\infty} \\
\eta &\leq \gamma^{T} \eta_{nc} + \frac{\gamma(1-\gamma^{T})}{1-\gamma} P_{\max} \eta_{c}
\end{aligned}
\tag{2}
$$

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 subtree. Finally, we evaluate the geometric series and express the result in terms of the sub-optimality of our evaluation function $\eta_{nc} = ||V_0 - V^{*}||_{\infty}$ and post-contact policy $\eta_{c} = ||V^{c} - V^{*}||_{\infty}$. In the worst case we can bound $\eta_{nc} \leq -R_{\min}/(1-\gamma)$ by setting $V_0 = 0$ since the reward function is bounded by $R_{\min} \leq R(s,a) \leq 0$.

As expected, equation (2) shows that $\eta \rightarrow 0$ as η_{c}, $\eta_{nc} \rightarrow 0$, the same result as in traditional online search algorithms (Ross et al., 2008). However, the post-contact error does not approach zero as $T \rightarrow \infty$ because the full policy can never outperform a sub-optimal post-contact policy π^{c}.

5. Post-contact policy

Suppose the robot is in belief state b while executing ξ, takes action a, receives contact observation $o \in O_c$, and transitions to the posterior belief state b'. We call $b' \in \Delta_o$ a *post-contact belief state* because the robot reached b' by receiving a contact observation. Our goal is to find a policy π^{c} that performs near-optimally over the *post-contact belief space* Δ_o consisting of all such belief states.

Our key insight is that contact occurs on a lower-dimensional manifold and produces a relatively small set of post-contact belief states (Section 5.1; Koval et al., 2013a,b, 2015). We use a point-based POMDP solver to find a policy that performs well over this space (Section 5.2). Section 5.3 discusses how to do so efficiently by re-using computation between similar belief states. Finally, we describe how to discretize the state space while preserving the structure of the continuous space (Section 5.4).

5.1. Contact manifold and observable contact manifold

Contact naturally partitions the state space S into three sets: (1) penetrating contact, (2) non-penetrating contact, and (3) no contact. This section summarizes the definitions of the contact manifold and observable contact manifold used in our prior work (Koval et al., 2013a,b, 2015).

Let $P_h \subseteq R^2$ be the geometry of the hand and $P_o(s) \subseteq \mathbb{R}^2$ be the geometry of the object at configuration $s \in S$. The

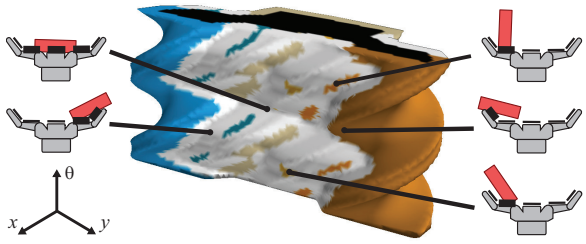


Fig. 5. Observable contact manifold S_o for a two-dimensional BarrettHand pushing a rectangular box. Each point on the manifold corresponds to a configuration of the object $s \in S_c$ that is in non-penetrating contact with the hand and is uniquely colored by the active contact sensors. Configurations that are in contact with multiple sensors are white. Reproduced from Koval et al. (2015).

set of all object poses that are in collision with the hand form the *configuration space obstacle* (Lozano-Pèrez, 1983)

$$S_{obs} = \text{CObstacle}(P_h) = \{s \in S : P_h \cap P_o(s) \neq \emptyset\}$$

of the hand in the object's configuration space. The *contact manifold* $S_c = S_{obs} \setminus \text{int}(S_{obs})$ consists of the lower-dimensional set of states that place the object in non-penetrating contact with the hand.

The state s lies on the contact manifold S_c during contact. However, the robot's contact sensors may not be able to sense contact over the entire surface of the hand. We define the *observable contact manifold* $S_o \subseteq S_c$ as the set of configurations that are in contact with one or more sensors and, thus, generate a contact observation $o \in O_c$.

Figure 5 shows the contact manifold colored by which sensors are active at each point. Points inside the obstacle are in penetrating contact, points outside the obstacle are in free space, and points on the surface lie on the contact manifold S_c. In this case, S_c is repeated twice along the θ-axis because the box exhibits rotational symmetry. The color of each point indicates which contact sensor(s) are active. For example, states in the large, dark orange region of the manifold are in contact with the left distal contact sensor. Regions of the contact manifold that are in simultaneous contact with multiple sensors are drawn as white.

5.2. Computing the post-contact policy

Receiving a contact observation $o \in O_c$ indicates that the state lies on the observable contact manifold $s \in S_o$. The set of all belief states that can be generated in this way form the *post-contact belief space* $\Delta_o \subseteq \Delta$. We discretize the continuous state space S (see Section 5.4) and use a point-based method to find a post-contact policy π^c that performs well over Δ_o.

Since the post-contact belief states are constrained to S_o, the discretized post-contact belief states exhibit sparse support, in other words, all non-contact states have zero probability. Furthermore, many belief states $\mathcal{R}(\Delta_o)$ reachable from Δ_o share the same sparsity because the state evolves on S_c during periods of contact. As a result, the discretized problem is particularly well suited to being solved by a point-based method (Lee et al., 2007).

A point-based solver represents the value function as a piecewise-linear convex function

$$V^{\pi^c}(b) = \max_{\alpha_i \in \Gamma} \langle \alpha_i, b \rangle$$

where each $\alpha_i \in \mathbb{R}^{|S|}$ is called an α-*vector* and $\langle \cdot, \cdot \rangle$ denotes the inner product. The solver iteratively expands the set of α-vectors Γ by performing backups of the Bellman equation at a discrete set of *belief points*. Most solvers assume that the initial belief state $b_0 \in \Delta$ is known and sample belief points from the reachable belief space $\mathcal{R}(b_0)$ by simulating sequences of actions and observations from b_0.

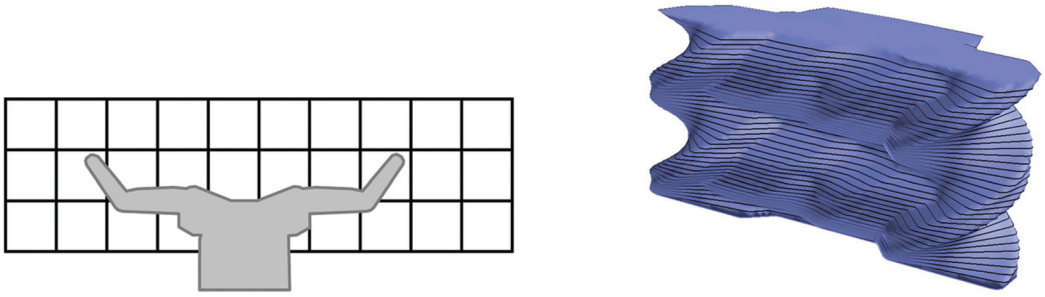
Unfortunately, the post-contact policy is not rooted at a single belief state b_0. Instead, we only know that b_0 lies in the post-contact belief space Δ_o. It is not possible to exhaustively enumerate Δ_o because the set is uncountably infinite. Instead, we initialize the solver with a discrete set of *post-contact belief points* $B \subseteq \Delta_o$ that are similar to the post-contact belief states that we expect to encounter during execution. In future work, we hope to refine B over time by adding the post-contact belief states observed during planning.

Given a set of n post-contact belief points $B = \{b_i\}_{i=1}^n$, we can directly use a point-based method to find a separate policy π_i^c for each belief point $b_i \in B$. The post-contact policy over B is defined by the union of the α-vectors of the constituent policies $\Gamma = \Gamma_1 \cup \ldots \cup \Gamma_n$. Each α-vector is a global lower bound on the true value function, so the policy defined by the union of these α-vectors Γ is a tighter bound on the value function than the α-vectors Γ_i of any one policy in isolation.

5.3. Computation via an augmented POMDP

Computing a separate policy π_i^c for each post-contact belief point $b_i \in B$ ignores the fact that information can be shared between policies; in other words, it is generally the case that $\mathcal{R}(b_i) \cap \mathcal{R}(b_j) \neq 0$. Our intuition, since the post-contact belief space is relatively small, is that many of the policies π_i^c will be similar.

We leverage this structure by converting our POMDP with the set of initial belief states B into an augmented POMDP with one initial belief. To do so, we construct an augmented state space $\hat{S} = S \times D$ where each augmented state $\hat{s} = (s, d) \in \hat{S}$ includes our state $s \in S$ in the underlying POMDP and a discrete variable $d \in D = \{0, \ldots, |B|\}$ that indicates which post-contact belief point is active. Our initial belief for the augmented POMDP is defined by a uniform prior distribution $d \sim$ uniform $[1, |B|]$ over belief



(a) Free space (b) Contact manifold

Fig. 6. The post-contact policy is computed on a discrete approximation of the underlying continuous state space. The discrete state space consists of two components: (a) a uniform discretization of free space in the trust region S_{trust} and (b) an explicit discretization of the contact manifold. Each black line overlaid on the contact manifold shows one polygonal slice of the configuration-space obstacle (C-obstacle) used for discretization.

points and a conditional distribution $b(s|d = i) = b_i$ defined by each post-contact belief point $b_i \in B$.

This construction yields a foliated belief space, similar to that encountered in a mixed observable Markov decision process (MOMDP) (Ong et al., 2009), where each belief point $b_i \in B$ is initially active with equal probability $1/|B|$. If left unchanged, this is equivalent to solving a separate POMDP policy for each value of D. Instead, we augment the transition function $\hat{T}$ to unconditionally transition to the indicator value $d = 0$. The value of $d = 0$ remains set for the remainder of planning and forces the solver to produce one unified policy over all of B.

5.4. State space discretization

Implementing the above algorithm requires discretizing the state space. This is challenging for two reasons: (1) the state space is unbounded and (2) contact introduces discontinuities into the transition and observation models that are difficult to capture in a uniform discretization. We describe how to address both of these challenges in this section.

The state space S is unbounded and, in theory, a policy could move the hand arbitrarily far away from the object. However, our intuition is that the optimal policy will not allow the object to stray far from the hand. 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 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 overestimate the support of $\mathcal{R}^*(\Delta_o)$ and only discretize this smaller state space.

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, for example 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, observation, or reward function will depend 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 (Horowitz and Burdick, 2013; Koval et al., 2013a). 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 $S_{\text{nc}} = S_{\text{trust}} \setminus S_{\text{c}}$ (Figure 6(a)) and (2) an explicit discretization of the contact manifold S_{c} (Figure 6(b)). Assuming the hand is polygonal and the object is radially symmetric, we can use the Minkowski sum to compute the configuration-space obstacle (C-obstacle) of the hand in the object's configuration space (Lozano-Pèrez, 1983). The contact manifold is the polygonal boundary of this C-obstacle (Koval et al., 2013b). We discretize the contact manifold by splitting the perimeter of this polygon into a sequence of equal-length line segments. If the object is not radially symmetric, we first discretize orientation and repeat this procedure on each orientation iso-contour of the contact manifold.

Using this discretization strategy, the observation model and reward functions both satisfy the Markov property. The transition model is not guaranteed to be Markovian: whether or not a discrete state transitions from S_{nc} to S_{c} depends on the underlying continuous state. However, in practice, we have found that the discrete belief dynamics closely match the continuous belief dynamics given a high enough resolution.

6. Pre-contact policy

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 (Figure 4(b)) by running an optimal graph search algorithm, such as A* (Hart et al., 1968), in an augmented belief space by recursively expanding V in equation (1) to

$$V(b) = \max_{\xi} \left[\sum_{t=0}^{\infty} \gamma^t \left(\prod_{i=0}^{t} \Omega(o_{\text{nc}}, b_{i+1}, a_i) \right) \left(R(b_{t+1}, a_t) + \sum_{o \in O_c} \Omega(o, b_{t+1}, a_i) V^{\text{c}}(b_{t+1}) \right) \right] \tag{3}$$

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.

6.1. 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

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

which is 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 $\sum_{o \in O_c} \Omega(o, b', a) V^{\text{c}}(b')$ 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}.

6.2. 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 *Markov decision process (MDP)* (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 $t_{\min}$ as

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

where 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 overestimating 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 $x = (b, t, p_{\text{nc}})$.

6.3. 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

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

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 the number of expansions, but, in practice, expands fewer nodes than A*. This is beneficial when, such as in our case, it is computationally expensive to generate successor nodes.

This search algorithm is closely related to online solvers that use a "default policy" to evaluate leaf nodes, for example POMCP (Silver and Veness, 2010). The key difference in this work is the post-contact policy is only defined over the trust region and is assumed to be optimal when it is available. In future work, we are interested in improving this technique by replacing A* with a modern online POMDP solver such as DESPOT (Somani et al., 2013) that is modified to incorporate the prior information provided by the post-contact policy (Gelly and Silver, 2007).

7. 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 timestep and, thus, does not plan to take multi-step information-gathering actions.

Specifically considering the post-contact policy, we hypothesize:

H1. *The POMDP post-contact policy will achieve higher value than the QMDP baseline.*
H2. *The POMDP post-contact policy will achieve success with higher probability than the QMDP baseline.*

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 affect the success rate of either 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 the state is fully observed, we expect the improvement to be larger for the QMDP policy than for the POMDP policy.

7.1. Experimental setup

We simulated our algorithm in a custom two-dimensional kinematic environment with planar, polygonal geometry. Each experiment consisted of a BarrettHand pushing an object with initial pose uncertainty (Figure 2, left) and uncertain physics parameters. We consider two objects: a disk of 3.5 cm radius and a 3.5 cm $\times$ 5 cm box. These dimensions are loosely based on the dimensions of common household objects that are commonly manipulated by HERB, for example the bottle used in Section 8.

In both cases, we consider the object's symmetry (radial symmetry for the disk and second-order symmetry for box) when computing the object's state. As a result, the state space for the disk is two-dimensional and the state space for the box is three-dimensional.

7.1.1. Transition model. We simulated the motion of the object using a penetration-based quasistatic physics model (Lynch et al., 1992), which we have used in prior work (Koval et al., 2013a,b), with a 2.5 mm and 3° step size. During each timestep, we sampled the finger–object coefficient of friction μ and the radius of the object's pressure

distance c from Gaussian distributions. We truncated the distributions by constraining $\mu \geq 0$ and $c \geq 1$ cm to ensure that the simulator remained stable.

We considered 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.

7.1.2. Observation model. We simulated binary contact sensor observations by checking collision between each contact sensor (a subset of the surface of the hand) and the object. We considered one sensor covering the interior of each finger's distal link. These observations perfectly discriminated between contact and no contact, but had a 10% probability of generating an incorrect contact observation while in contact with the object.

7.1.3. State estimator. During execution of the pre-contact trajectory ξ, we use a particle filter to perform belief updates. We know that $o_t = o_{\text{nc}}$ during execution of ξ, so we do not encounter the starvation problems typically found when applying the particle filter to contact sensing (Koval et al., 2013a; Zhang and Trinkle, 2012). We must use a continuous-space state estimator during execution of the pre-contact policy because the discrete belief dynamics are only defined over S_{trust}.

Once we transition to the post-contact policy, for example 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 described in Section 5.4. 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 exactly matches the belief dynamics used during planning.[1]

7.1.4. Evaluation metric. 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 take advantage of observations during execution, but not execute multi-step information-gathering actions to reduce uncertainty.

Ideally, we would also compare the proposed algorithm against a policy computed by SARSOP over the full state space. We would expect this algorithm to achieve similar performance to SARSOP, but require significantly less computation time to adapt to new instances of the problem. This comparison is omitted because we were not able to get the implementation of SARSOP provided by the APPL toolkit (Kurniawati et al., 2008) to successfully load the full discrete POMDP model. This is likely due the model's prohibitively large size (Section 3.4).

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. Since our actions are of size 2 cm, this corresponds to simulating 1 m of end-effector motion.

As described above, performing well on the discrete system dynamics does not guarantee that the same policy will be effective at grasping an object subject to the continuous system dynamics. We 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. A good policy will grasp the object more quickly and with higher probability than the baseline.

7.2. Discretization

We selected a trust region S_{trust} of size 15 cm $\times$ 50 cm around the hand. This region is large enough to include objects between the fingers, immediately in front of the hand, and rolling off the fingertips. 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 using exact Minkowski differences (Koval et al., 2013b).

In both cases, as described in Section 5.4, we used Monte Carlo sampling to compute a discrete transition model, observation model, and reward function for this state space. We approximate each of these functions using Monte Carlo sampling. First, we sample 25 states uniformly at random from the set of states that discretize to the function's input. Next, we evaluate the continuous function on each sample. Finally, we approximate the value of the discrete function by averaging the samples.

The cylinder's two-dimensional discrete state space consisted of $|S| = 281$ states, consisting of $|S_{\text{nc}}| = 175$, $|S_{\text{c}}| = 105$, and one unknown state to represent all states that lie outside S_{trust}. The box, due to its larger three-dimensional state space, had $|S| = 4239$ states split between $|S_{\text{nc}}| = 2625$, $|S_{\text{c}}| = 1613$, and one unknown state.

7.3. 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 1604 backups to converge over 0.53 s (0.33 ms per backup) for the disk and 1729 backups over 8.36 s seconds (4.84 ms per backup) for the box. The resulting policies, each consisting of five α-vectors, respectively took 0.71 ms and 7.64 ms to evaluate on a discrete belief state.

The large difference in performance is a result of the box's 15-fold larger discrete state space.

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 $\mu_i = [x_i, y_i, \theta_i]^{\mathrm{T}}$ and covariance matrix $\Sigma_i^{1/2} = \mathrm{diag}[5\ \mathrm{cm},\ 5\ \mathrm{cm}, 15^{\circ}]$. 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 cm $\leq y \leq 10$ cm and an orientation selected uniformly $0 \leq \theta < 360°$. The set B was constructed in this way in an attempt to cover the full range of post-contact beliefs that we expect to encounter during execution.

We solved the resulting discrete POMDP using the SARSOP implementation provided by the APPL tookit (Kurniawati et al., 2008). SARSOP is a point-based method that uses an approximation of the optimally reachable belief space $\mathcal{R}^*(b_0)$ to guide its planning. We ran SARSOP for five minutes, during which it produced a policy consisting 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 of the policy. Each rollout was initialized with a belief drawn from B and was forward-simulated for 50 timesteps. Figure 8(a) shows that the POMDP policy outperformed the QMDP policy on both objects. This result confirms hypothesis H1: it is advantageous to formulate the contact manipulation problem as a POMDP instead of a more computationally efficient MDP or deterministic planning problem.

Note that the POMDP policy achieves similar value on both objects, whereas the QMDP policy achieves lower value on the box than the disk. This is supported by our intuition: adding an additional partially observable dimension to the state space increases the importance of reasoning about uncertainty. For example, it is possible to accurately localize the position of the disk relative to the hand after receiving one contact observation. This is not true for the box because there can still be significant uncertainty over the object's orientation.

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 state 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, that is, achieved success. Figure 8(b) and (c) 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.

7.4. Pre-contact trajectory (H3, H4)

The post-contact policies described above are only valid in the small region S_{trust} near the hand. We used the search algorithm described in Section 6 to extend these policies to a longer horizon using the continuous belief dynamics.

We sampled 250 prior beliefs with $\Sigma^{1/2} = \mathrm{diag}\ [5\ \mathrm{cm}^2, 5\ \mathrm{cm}^2, 15°]$ 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.

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 lay in S_{trust}, or (3) the search timed out after 20 s.

The robot began each trial by executing ξ until it observed contact $o_t \in O_c$ or exhausted the trajectory by reaching $t > |\xi|$. Figure 7 shows the POMDP pre-contact trajectory and several snapshots (a) to (e) of the post-contact policy for two different trials with the box. As expected, the pre-contact trajectory attempts to make contact with the object as quickly as possible by moving the hand towards the prior distribution. Once (b) contact occurs, the post-contact policy quickly (c) localizes the object and (e) pushes it into the goal region.

Figure 9 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 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.

7.5. 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 of the analytic representation of the contact manifold (Koval et al., 2013b) to compute that nine (disk) 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.

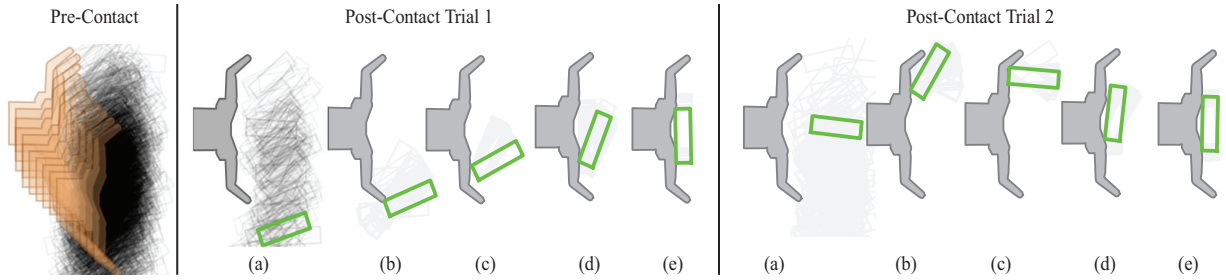


Fig. 7. Two rollouts of a policy produced by our algorithm with a post-contact policy produced by the POMDP policy. The robot begins by executing the pre-contact trajectory ξ in (a) the initial belief state until (b) contact is observed. Then (c)–(d) the robot executes the post-contact policy π^c until (e) the object is pushed into the goal region.

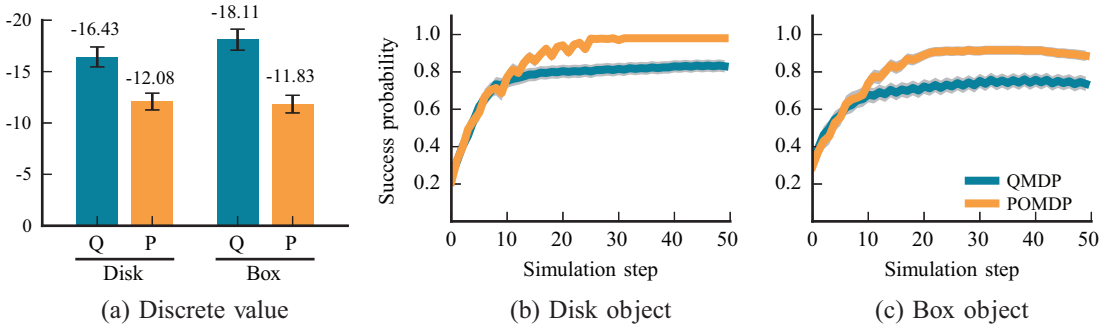


Fig. 8. Performance of the post-contact policy for the disk and box objects. (a) Expected value, at depth 50, of the QMDP (Q) and POMDP (P) post-contact policies evaluated on the discrete system dynamics. Note that $R(\cdot,\cdot) < 0$, so less-negative values are better. (b) and (c) Success rate of the same policies evaluated on the continuous system dynamics. The error bars and gray shaded region denote a 95% confidence interval.

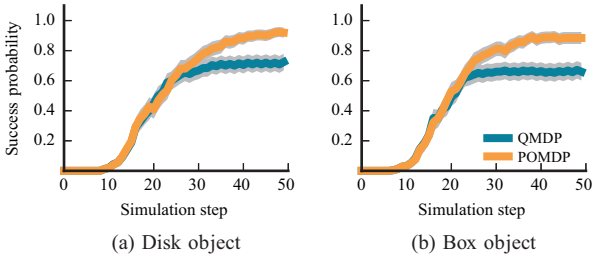


Fig. 9. Performance of the full pre- and post-contact policy for the (a) disk and (b) box objects. Transitioning between the pre- and post-contact policy occurs at a different time for each simulated scene. The gray shaded region denotes a 95% confidence interval.

Figure 10(a) shows that increasing sensor coverage unilaterally 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. For the disk, Figure 10(b) shows that the QMDP policy improves in both speed and success rate. The POMDP policy achieves the same success rate as before, but reaches that level of success more quickly. For the box, Figure 10(b) shows that increasing sensor coverage significantly improves both the success rate and speed of both policies.

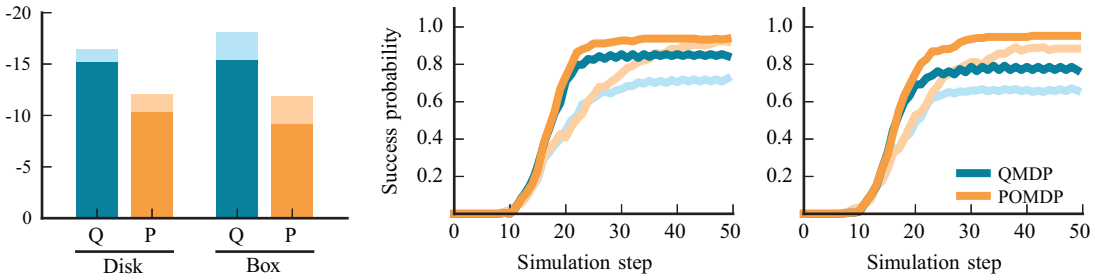
Figure 10 indicates that increasing sensor coverage significantly improves the performance of the QMDP policy. In the limit, when state is fully observed, the QMDP policy is optimal. However, since our sensing is still quite limited, the POMDP policy still outperforms the QMDP policy. This is particularly true when manipulating the radially asymmetric box for the same reasons as described in Section 7.3.

8. Real-robot experiments

In this section, we introduce a simple technique (Sections 8.1 and 8.2) for executing the policies generated by our algorithm on a manipulator subject to kinematic constraints. We also present results from real-robot experiments (Sections 8.3 and 8.4) on HERB, a bi-manual mobile manipulator designed and built by the Personal Robotics Lab at Carnegie Mellon University (Srinivasa et al., 2012).

8.1. Kinematic constraints

Our POMDP model assumes that an isolated end-effector is able to make unconstrained motion in the plane to achieve



(a) Post-contact value (b) Disk object, full policy (c) Box object, full policy

Fig. 10. Effect of increasing sensor coverage on the performance of the full pre- and post-contact policy. Again, note that less-negative values are better. (a) Both the QMDP (Q) and POMDP (P) policies achieve higher value with full sensor coverage (dark bars) than with only fingertip sensors (light bars). (b) and (c) Both policies perform better with full sensor coverage (dark lines) than with only fingertip sensors (light lines). Error bars are omitted to reduce visual clutter.

the goal. However, when manipulating objects on a real robot, the end-effector is attached to a manipulator that moves in some configuration space $Q = \mathbb{R}^n$. Executing the policy requires that we constrain the motion of the end-effector, which moves in $SE(3)$ to the plane while obeying the kinematic constraints imposed by the manipulator.

Consider a robot with configuration space Q manipulating an object on a plane. We constrain the pose of the robot's end-effector to be a fixed height above the plane with its palm orthogonal to the surface. In configuration space, this is equivalent to constraining the configuration of the robot to a constraint manifold of inverse kinematic solutions $Q_{\text{plane}} \subseteq Q$.

We begin at configuration $q_0 \in Q_{\text{plane}}$ in belief state $b(s_0)$ and begin executing policy π. At each timestep t, we recursively update our belief state from $b(s_{t-1})$ to $b(s_t)$ using a Bayes filter and evaluate our policy to choose action $a_t = \pi(b)$. This action $a_t = (v_a, T_a)$ is a planar velocity $v_a = (\dot{x}, \dot{y}, \dot{\theta}) \in se(2)$ that cannot be directly executed on the robot. To circumvent this, we use the Jacobian pseudo-inverse (Buss, 2004)

$$\dot{q} = J^{\dagger}\hat{v}_a = (J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}\hat{v}_a \tag{4}$$

to find a joint velocity $\dot{q}$ that realizes v_a. In this equation, $J \in \mathbb{R}^{6\times n}$ is the manipulator Jacobian evaluated at q and $\hat{v}_a = (\dot{x}, \dot{y}, 0, 0, 0, \dot{\theta})$ is the desired velocity of the end-effector in $SE(3)$. Note that the three elements corresponding to out-of-plane motion are set to zero.

We iteratively evaluate Equation (4), incorporating updated values of q as they become available, until T_a time has elapsed. At this point, we repeat the process at time $t + 1$ by selecting action a_{t+1}. Since $q_0 \in Q_{\text{plane}}$ and $\hat{v}_a$ only has non-zero components in the plane, the update in Equation (4) preserves the invariant that $q \in Q_{\text{plane}}$.

8.2. Kinematic feasibility

Assuming that $\text{rank}(J) \geq 6$, the Jacobian pseudo-inverse execution strategy described above guarantees that the manipulator remains in the plane. However, Equation (4) does not guarantee that the joint velocity $\dot{q}$ necessary to execute a_t is feasible to execute on the robot.

Infeasibility may occur for several reasons. First, executing $\dot{q}$ may drive the robot into a joint limit, self-collision, or collision with the environment. Second, it may not be possible for the controller to remain on Q_{plane} when q is near a singularity, that is, J is poorly conditioned. Finally, unmodeled parts of the robot may come into contact with the object. For example, HERB's forearm, which is not modeled during planning, could sweep through a region of high probability in $b(s_t)$.

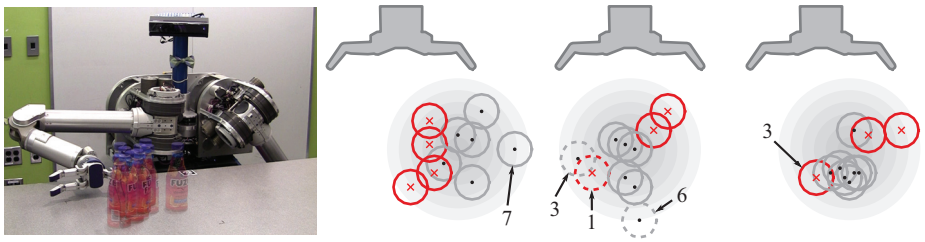
Ideally, we could avoid all of these issues by directly planning in the joint configuration space $Q \times S$ of arm configurations and object poses. This would allow us to account for the kinematic feasibility of actions during planning and even perform whole-arm manipulation, for examples sweeping with the forearm. This is, unfortunately, intractable to discretize and solve. For HERB, $\dim(S \times Q) = 10$, resulting in a threefold increase in dimensionality of the POMDP we formulated in Section 3.1.

Instead, in this paper, we simply halt execution if the selection action is infeasible. We are interested in addressing kinematic constraints in a principled way in future work (Section 9.3).

8.3. Experimental design

We executed the policies produced in Section 7 on HERB, a bi-manual mobile manipulator designed and built by the Personal Robotics Lab at Carnegie Mellon University (Srinivasa et al., 2012). HERB is equipped with a seven-degree-of-freedom Barrett WAM arm (Salisbury et al., 1988) and the BarrettHand end-effector (Townsend, 2000). We used HERB to push a bottle across the table (Figure 11(a)) using an experimental setup similar to the one we used for the disk in Section 7.

We generated three Gaussian initial belief states with means μ_1 = [35 cm, 12 cm], μ_2 = [35 cm, 0 cm], and μ_3 = [35 cm, 6 cm] relative to HERB's hand. All three initial belief states had covariance $\Sigma^{1/2}$ = diag[5 cm, 5 cm] and the goal region was a 4 cm × 8 cm rectangle in front

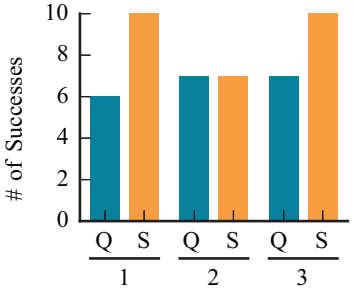


(a) Experimental setup (b) Belief 1 (c) Belief 2 (d) Belief 3

Fig. 11. (a) HERB in the initial configuration used to begin each trial. The 10 samples drawn from belief 1 are shown as a semi-transparent overlay. (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 × 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 1.

Table 1. Outcome of executing the QMDP and SARSOP policies on HERB. Each policy was executed on 10 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 unmodelled errors in the observation model. All other failures occurred because the object came to rest on a sensorless part of the hand outside the goal region.

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



of HERB's palm. We sampled each Gaussian 10 times and used the same 30 samples to evaluate both QMDP and SARSOP. The three initial belief states and 30 samples used for evaluation are shown in Figure 11(b) to (d).

HERB used the BarrettHand's strain gages 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 the Jacobian pseudo-inverse controller described in Section 8.1. 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 was considered successful if the bottle was in the goal region at the moment of termination.

8.4. Experimental results

Table 1 shows the outcome of all 30 trials executed on HERB (Srinivasa et al., 2012). The position of the bottle used in each trial is shown in Figure 11. The SARSOP policy achieved success in 27/30 trials (90%). In contrast, the QMDP policy only achieved success in 20/30 trials (67%). These results are consistent with our simulation experiments in Section 7.5: the SARSOP policy is predicted to succeed with 100% probability and the QMDP policy is only predicted to succeed with 70% probability.

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 timesteps, 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 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 (Hebert et al., 2013; Javdani et al., 2013; Petrovskaya and Khatib, 2011). Figure 12, 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.

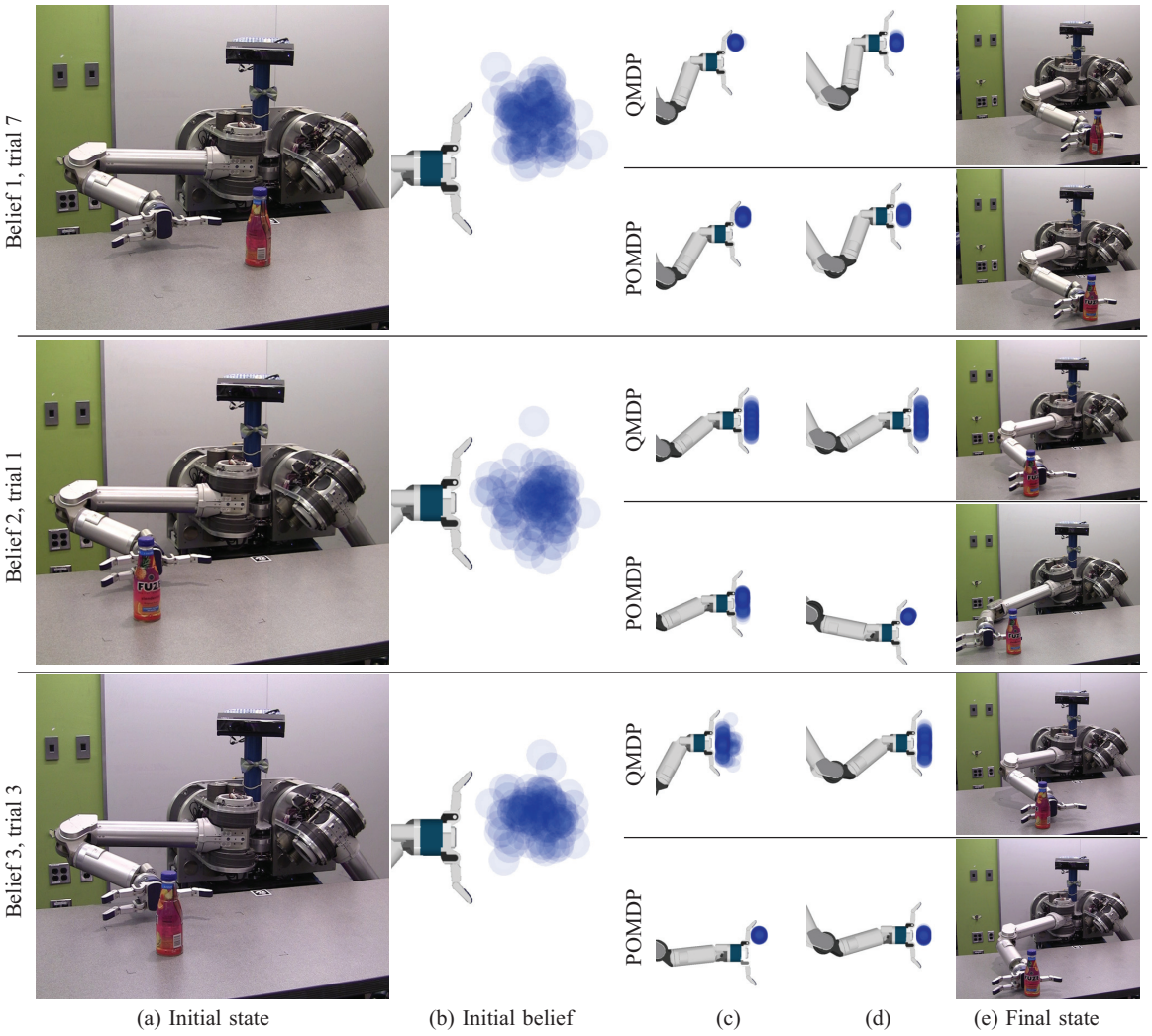


Fig. 12. Three trials executed on HERB with the QMDP and POMDP policies. Columns (a) and (e) show, respectively, the initial and final poses of the object relative to the hand. Column (b) shows several samples drawn from the initial belief state. Columns (c) and (d) show the belief state over object pose at two points during execution. The trial numbers referenced in this figure correspond to those used in Figure 11 and Table 1.

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 timesteps 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 1; Figure 12, 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 Figure 11). The bottle came into contact with a sensor during execution of the pre-contact policy and, thus, was localized.

The SARSOP policy used information-gathering actions to localize the object regardless of its initial configuration. As a result, the SARSOP policy succeeded in 20/20 trials on beliefs 1 and 3. Surprisingly, the SARSOP policy failed in three trials from belief 2. Two trials (marked with † in Table 1; Figure 12, 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. This type of *kinematic failure* is avoidable by planning in the joint space $S \times Q$ of object pose S and robot configuration Q, instead of the greedy Jacobian-based procedure outlined in Section 8.1.

The SARSOP policy also failed in belief 2 trial 6 (marked with ‡ in Table 1) for a different reason. HERB's fingertip contacted the bottle and generated a contact observation. This is inconsistent with the observation model used for planning, which assumes that the sensor only activates from contact with the inside of the finger. As a result, HERB incorrectly inferred that the bottle was touching the inside of finger, instead of the fingertip, and failed to complete the grasp. This failure highlights a

fundamental limitation of our approach: the policy produced by our algorithm can only compensate for uncertainty that is included in the POMDP model used for planning.

9. Conclusion and future work

In this paper, we formulated the problem of using real-time feedback from contact to create closed-loop pushing actions. To do so, we formulated the problem as a POMDP (Section 3.1) with a transition model based on a physics simulator and a reward function that drives the robot towards a successful grasp.

We demonstrated that the optimal policy can be efficiently decomposed (Section 4) into an open-loop pre-contact trajectory and a closed-loop post-contact policy with a bounded impact on the performance of the overall policy. First, as an offline pre-computation step, we exhaustively solved for the post-contact policy using a point-based POMDP solver (Section 5). Then, when presented with a specific problem instance, we found an open-loop pre-contact trajectory using an A* search (Section 6). The robot begins by executing the pre-contact trajectory, then transitions to the post-contact policy once contact is observed.

Our simulation results (Section 7) show that (across two objects and two sensor configurations) the policy produced by this algorithm achieves a successful grasp more quickly and with a higher success rate than the baseline QMDP policy. Finally, we present real-robot experiments on HERB (Section 8) that demonstrate that the policies produced by this algorithm are effective on a real robot.

Finally, we conclude by discussing several limitations of this approach (Section 9.1) and directions for future work. Our algorithm, as described in this paper, requires discretizing the POMDP to find a post-contact policy (Section 9.2). This limitation, along with the challenges of planning in configuration space, prohibit us from considering kinematic feasibility during planning (Section 9.3). Finally, we only plan for the motion of the end-effector and rely on input from an external observer to execute a grasp (Section 9.4). We are excited about addressing all three of these limitations in future work.

9.1. Policy decomposition

Our key insight is that we can decompose the optimal policy into an open-loop pre-contact trajectory followed by a closed-loop post-contact policy. This decomposition is valid on any POMDP and is not restricted to the manipulation problem. However, performing this decomposition requires two important capabilities: (1) the ability to pre-compute an offline policy that is valid for all post-contact belief states Δ_o and (2) a good heuristic $h(x)$ to guide the pre-contact search. The structure of the contact manipulation problem provides for both of these capabilities.

First, the discriminative nature of contact sensors limits the size of the set of post-contact belief states Δ_o. This property allows us to compute π^c by initializing a point-based solver with several initial belief points $B \subseteq \Delta_o$ drawn from Δ_o. If a different sensor was being used for feedback, for example a camera or noisy range sensor, Δ_o could be much larger and this solution method might not be possible. In the extreme case, where $\Delta_o = \Delta$, finding the optimal post-contact policy is as hard as solving the full problem.

Second, the quasistatic assumption (Mason, 1986) reduces the dimensionality of our state space. The quasistatic assumption states that an object will stop moving as soon as it leaves contact with the hand. Prior work has shown that the quasistatic assumption is a good approximation for the planar manipulation of many household objects, since frictional forces dominate the effect of dynamics (Dogar and Srinivasa, 2010, 2011). However, as a result of this assumption, our approach cannot directly plan for objects with non-trivial dynamics, for example a ball that rolls when pushed.

Finally, we exploit our intuition that it is unlikely for the optimal post-contact policy to allow the object to stray far from the hand. We formalize this intuition by restricting the post-contact policy to a small trust region S_{trust} near the hand. This optimization limits the types of policies our algorithm can generate: we cannot generate post-contact policies that require large, global movement of the hand. For example, our algorithm cannot efficiently generate policies for problems that require long transit phases or coordinating two distant end-effectors.

Despite these limitations, we believe that the policy decomposition described in Section 4 may be useful for domains outside of manipulation. This decomposition closely mirrors the "simple after detection" property of POMDPs observed in aerial collision avoidance (Bai et al., 2011). This property states that all but one observation lead to belief states that admit simple sub-policies. In our case, we subvert the need for simple sub-policies by leveraging the small number of post-contact beliefs to pre-compute an exhaustive post-contact policy.

9.2. Discretization

Our implementation assumes that the robot lives in a two-dimensional world with $S = SE(2)$, fixed hand geometry, and a discrete action set. We plan to extend this algorithm to $SE(3)$, to more complex interaction with the environment (e.g. toppling) and to articulated hands with internal degrees of freedom. In both cases, these generalizations increase the dimensionality of the state space.

Unfortunately, increasing the dimensionality of the problem exponentially increases $|S|$ and $|A|$. The size of these spaces makes it intractable to compute and fully discretize the POMDP model using the technique described in Section 5.4. The lack of a compact discrete model makes the POMDP significantly harder to solve. We believe that the increased computational complexity could be partially addressed by using a continuous POMDP solver to entirely avoid discretizing the problem. There has been recent work

(Koval et al., 2013a; Zhang and Trinkle, 2012; Zhang et al., 2013) on using a sample-based representation of belief space to track the continuous pose of the object over time. The key challenge is to find a representation of the post-contact policy that can both be represented compactly and still generalize between belief states represented in this way.

9.3. Kinematic feasibility

Our formulation of the contact manipulation POMDP considers state to only be the pose of the object relative to the hand. As described in Section 8.2, an action may be infeasible to self-collision, robot–environment collision, joint limits, or other limitations of the robot's hardware. We intentionally avoided these cases in our real-robot experiments on HERB (Section 8) by carefully selecting HERB's starting configuration and executing the policy on an empty table. Even so, the SARSOP policy failed in two trials by violating kinematic constraints.

Manipulating objects in human environments requires dealing with kinematic infeasibility and clutter. We can easily incorporate arbitrary kinematic constraints into the pre-contact search by tracking the full configuration of the robot each time a new node is expanded. This is equivalent to planning the pre-contact trajectory in a constrained subset of the full configuration space of the robot, rather than the pose of a hand in the plane.

Unfortunately, it is not feasible to pre-compute a post-contact policy for all possible robot and obstacle configurations. Instead, we must rely on online techniques to find π^{c}. We hope to combine the advantage of both approaches by using value function V^{c}, which is computed for the end-effector in isolation, to guide a search in the full-dimensional state space. Prior work has shown that it is possible to use an offline policy to efficiently guide an online search (Gelly and Silver, 2007). We are encouraged by the performance of recent online POMDP solvers (Kurniawati and Yadav, 2013; Silver and Veness, 2010; Somani et al., 2013) on continuous state spaces and are interested in pursuing this idea in future work.

9.4. Grasping reward function

In Section 8, we made the observation that closing the hand is a critical part of policy execution. However, the POMDP model described in Section 3 only plans to execute relative motions of the end-effector and leaves the critical decision of when to close the hand to an external observer. Instead, if our goal is to grasp the object, we can encode the decision to execute a grasp directly into our POMDP model.

In the simplest case, if we have no model of the effect of executing a grasp on the pose of the object, we consider executing a grasp to be a terminal action. To do so, we augment our action space $\hat{A} = A \cup \{a_{\text{grasp}}\}$ with a discrete action a_{grasp} and our state space $\hat{S} = S \cup \{s_{\text{term}}\}$ with a terminal state s_{term}. Executing a_{grasp} immediately ends the trial by unconditionally transitioning to s_{term}; in other words, $T(\cdot, a_{\text{grasp}}, s_{\text{term}}) = 1$. The grasp is considered successful if $s_t \in G$ and is otherwise considered a failure.

We encode success and failure into the reward function

$$R(s, a_{\text{grasp}}) = \begin{cases} R_{\text{succ}} & s \in G \\ -T_{\max} & \text{otherwise} \end{cases}$$

where $T_{\max} = \max_{a \in A} T_a$ is the maximum duration of an action and $R_{\text{succ}} < 0$ encodes the cost of executing a grasp. We assign zero reward to the terminal state $R(s_{\text{term}}, \cdot) = 0$ and negative reward $R(s, a) = -T_a$ to all actions $a \neq a_{\text{grasp}}$ for $s \neq s_{\text{term}}$.

The value of R_{succ}, similar to the parameter ϵ described above, controls the trade-off between execution time and success probability. At one extreme, where $R_{\text{succ}} < -\frac{1}{1-\gamma} T_{\max}$, the optimal policy will never execute a_{grasp}. Conversely, if $R_{\text{succ}} > -T_{\min}$ for $T_{\min} = \min_{a \in A} T_a$, the optimal policy will immediately execute a_{grasp} regardless of the probability of success. Values between these extremes trade off between execution time and success probability.

Alternatively, if we do have a model of how the object moves during a grasp, we can consider a_{grasp} to be a high-cost, non-terminal action. This formulation would allow a policy to treat grasp execution as an information-gathering action. For example, the policy may decide to close the hand in a high-probability region of state space to test whether the object is present in that region. We are interested in considering both of these formulations of the reward function in future work.

Acknowledgments

We would like to thank David Hsu, Geoff Gordon, Aaron Johnson, Jennifer King, Joelle Pineau, Prasanna Velagapudi, Stefanos Nikolaidis, and our anonymous reviewers for their helpful input. We would also like to thank the members of the Personal Robotics Lab for their support.

Funding

This work was supported by the NASA Space Technology Research Fellowship (award NNX13AL62H), the National Science Foundation (award IIS-1218182) and the Toyota Motor Corporation (award 1011344).

Note

1. 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 an earlier version of this work (Koval et al., 2014). In those experiments, we used the manifold particle filter (Koval et al., 2013a) to track the 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.

References

Agha-mohammadi AA, Chakravorty S and Amato NM (2011) FIRM: Feedback controller-based information-state

roadmap – a framework for motion planning under uncertainty. In: *IEEE/RSJ international conference on intelligent robots and systems*.

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

Bai H, Hsu D, Kochenderfer M, et al. (2011) Unmanned aircraft collision avoidance using continuous-state POMDPs. In: *Robotics: Science and systems*.

Bellman R (1957) *Dynamic Programming*. Princeton, NJ: Princeton University Press.

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

Brokowski M, Peshkin M and Goldberg K (1993) Curved fences for part alignment. In: *IEEE international conference on robotics and automation*.

Brost R (1988) Automatic grasp planning in the presence of uncertainty. *The International Journal of Robotics Research* 7(1): 3–17.

Brunskill E, Kaelbling L, Lozano-Pèrez T, et al. (2008) Continuous-state POMDPs with hybrid dynamics. In: *International symposium on artificial intelligence and mathematics*.

Buss S (2004) Introduction to inverse kinematics with Jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation* 17: 1–19.

Chang L, Srinivasa S and Pollard N (2010) Planning pre-grasp manipulation for transport tasks. In: *IEEE International conference on robotics and automation*.

Dang H, Weisz J and Allen P (2011) Blind grasping: Stable robotic grasping using tactile feedback and hand kinematics. In: *IEEE International conference on robotics and automation*, pp. 5917–5922.

Dogar M and Srinivasa S (2010) Push-grasping with dexterous hands: Mechanics and a method. In: *IEEE/RSJ International conference on intelligent robots and systems*.

Dogar M and Srinivasa S (2011) A framework for push-grasping in clutter. In: *Robotics: Science and systems*.

Dogar M and Srinivasa S (2012) A planning framework for non-prehensile manipulation under clutter and uncertainty. *Autonomous Robots* 33(3): 217–236.

Dogar M, Hsiao K, Ciocarlie M, et al. (2012) Physics-based grasp planning through clutter. In: *Robotics: Science and systems*.

Erdmann M and Mason M (1988) An exploration of sensorless manipulation. *IEEE Journal of Robotics and Automation* 4(4): 369–379.

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

Gelly S and Silver D (2007) Combining online and offline knowledge in UCT. In: *International conference on machine learning*.

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

Hauser K and Ng-Thow-Hing V (2011) Randomized multi-modal motion planning for a humanoid robot manipulation task. *The International Journal of Robotics Research* 30(6): 678–698.

Hebert P, Howard T, Hudson N, et al. (2013) The next best touch for model-based localization. In: *IEEE international conference on robotics and automation*.

Horowitz M and Burdick J (2013) Interactive non-prehensile manipulation for grasping via POMDPs. In: *IEEE international conference on robotics and automation*.

Howe R and Cutkosky M (1996) Practical force-motion models for sliding manipulation. *The International Journal of Robotics Research* 15(6): 557–572.

Hsiao K (2009) *Relatively robust grasping*. PhD Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Hsiao K, Kaelbling L and Lozano-Pèrez T (2007) Grasping POMDPs. In: *IEEE international conference on robotics and automation*.

Hsiao K, Lozano-Pérez T and Kaelbling L (2008) Robust belief-based execution of manipulation programs. In: *Workshop on the algorithmic foundations of robotics*.

Hwang Y and Ahuja N (1992) Gross motion planning – a survey. *ACM Computing Surveys* 24(3): 219–291.

Hyafil N and Bacchus F (2003) Conformant probabilistic planning via CSPs. In: *International conference on automated planning and scheduling*.

Javdani S, Klingensmith M, Bagnell J, et al. (2013) Efficient touch based localization through submodularity. In: *IEEE international conference on robotics and automation*.

Kaelbling L and Lozano-Pérez T (2013) Integrated task and motion planning in belief space. *The International Journal of Robotics Research* 32(9–10): 1194–1227.

Kaelbling L, Littman M and Cassandra A (1998) Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101(1–2): 99–134.

Kappler D, Chang L, Przybylski M, et al. (2010) Representation of pre-grasp strategies for object manipulation. In: *IEEE-RAS international conference on humanoid robots*.

Koval M, Dogar M, Pollard N, et al. (2013a) Pose estimation for contact manipulation with manifold particle filters. In: *IEEE/RSJ international conference on intelligent robots and systems*.

Koval M, Pollard N and Srinivasa S (2013b) Manifold representations for state estimation in contact manipulation. In: *International symposium of robotics research*.

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

Koval M, Pollard N and Srinivasa S (2015) Pose estimation for planar contact manipulation with manifold particle filters. *The International Journal of Robotics Research* 34(7): 922–945.

Kurniawati H and Yadav V (2013) An online POMDP solver for uncertainty planning in dynamic environment. In: *International symposium of robotics research*.

Kurniawati H, Hsu D and Lee W (2008) SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: *Robotics: Science and systems*.

LaValle S and Hutchinson S (1998) An objective-based framework for motion planning under sensing and control uncertainties. *The International Journal of Robotics Research* 17(1): 19–42.

Lee W, Rong N and Hsu D (2007) What makes some POMDP problems easy to approximate? In: *Advances in neural information processing systems*.

Littman M (1996) *Algorithms for sequential decision making*. PhD Thesis, Brown University, Providence, RI.

Littman M, Cassandra A and Kaelbling L (1995) Learning policies for partially observable environments: Scaling up. In: *International conference on machine learning*.

Li Q, Schürmann C, Haschke R, et al. (2013) A control framework for tactile servoing. In: *Robotics: Science and systems*.

Lozano-Pèrez T (1983) Spatial planning: A configuration space approach. *IEEE Transactions on Computers* C-32(2): 108–120.

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

Lynch K and Mason M (1996) Stable pushing: Mechanics, controllability, and planning. *The International Journal of Robotics Research* 15(6): 533–556.

Lynch K, Maekawa H and Tanie K (1992) Manipulation and active sensing by pushing using tactile feedback. In: *IEEE/RSJ international conference on intelligent robots and systems*.

Madani O, Hanks S and Condon A (1999) On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In: *National conference on artificial intelligence*.

Mason M (1986) Mechanics and planning of manipulator pushing operations. *The International Journal of Robotics Research* 5(3): 53–71.

Meeussen W, Rutgeerts J, Gadeyne K, et al. (2007) Contact-state segmentation using particle filters for programming by human demonstration in compliant-motion tasks. *IEEE Transactions on Robotics* 23(2): 218–231.

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

Ong S, Png S, Hsu D, et al. (2009) POMDPs for robotic tasks with mixed observability. In: *Robotics: Science and systems*.

Pastor P, Righetti L, Kalakrishnan M, et al. (2011) Online movement adaptation based on previous sensor experiences. In: *IEEE/RSJ international conference on intelligent robots and systems*.

Pearl J (1984) *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Boston, TX: Addison-Wesley.

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

Pineau J, Gordon G and Thrun S (2003) Point-based value iteration: An anytime algorithm for POMDPs. In: *International joint conference on artificial intelligence*.

Platt R, Fagg A and Grupen R (2010a) Nullspace grasp control: Theory and experiments. *IEEE Transactions on Robotics* 26(2): 282–295.

Platt R, Kaelbling L, Lozano-Pèrez T, et al. (2011) Simultaneous localization and grasping as a belief space control problem. In: *International symposium of robotics research*.

Platt R, Kaelbling L, Lozano-Pèrez T, et al. (2012) Non-Gaussian belief space planning: Correctness and complexity. In: *IEEE international conference on robotics and automation*.

Platt R, Tedrake R, Kaelbling L, et al. (2010b) Belief space planning assuming maximum likelihood observations. In: *Robotics: Science and systems*.

Pohl I (1977) *Practical and theoretical considerations in heuristic search algorithms*. *Machine Intelligence* 8: 55–72.

Porta J, Vlassis N, Spaan M, et al. (2006) Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research* 7: 2329–2367.

Ross S, Pineau J, Paquet S, et al. (2008) Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research* 32(1): 663–704.

Salisbury K, Townsend W, Eberman B, et al. (1988) Preliminary design of a whole-arm manipulation system (WAMS). In: *IEEE international conference on robotics and automation*.

Schneider A, Sturm J, Stachniss C, et al. (2009) Object identification with tactile sensors using bag-of-features. In: *IEEE/RSJ international conference on intelligent robots and systems*.

Seiler K, Kurniawati H and Singh S (2015) An online and approximate solver for POMDPs with continuous action space. In: *IEEE international conference on robotics and automation*.

Silver D and Veness J (2010) Monte-Carlo planning in large POMDPs. In: *Advances in neural information processing systems*.

Simunovic S (1979) *An information approach to parts mating*. PhD Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Smith D and Weld D (1998) Conformant graphplan. In: *National conference on artificial intelligence*.

Somani A, Ye N, Hsu D, et al. (2013) DESPOT: Online POMDP planning with regularization. In: *Advances in neural information processing systems*.

Srinivasa S, Berenson D, Cakmak M, et al. (2012) HERB 2.0: Lessons learned from developing a mobile manipulator for the home. *Proceedings of the IEEE* 100(8): 1–19.

Stulp F, Theodorou E, Buchli J, et al. (2011) Learning to grasp under uncertainty. In: *IEEE international conference on robotics and automation*, pp. 5703–5708.

Thrun S, Burgard W and Fox D (2005) *Probabilistic Robotics*. Cambridge, MA: MIT Press.

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

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

Van den Berg J, Abbeel P and Goldberg K (2011) LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *The International Journal of Robotics Research* 30(7): 895–913.

Will P and Grossman D (1975) An experimental system for computer controlled mechanical assembly. *IEEE Transactions on Computers* 100(9): 879–888.

Xiao J (1993) Automatic determination of topological contacts in the presence of sensing uncertainties. In: *IEEE international conference on robotics and automation*.

Xu D, Loeb G and Fishel J (2013) Tactile identification of objects using Bayesian exploration. In: *IEEE international conference on robotics and automation*.

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

Zhang L and Trinkle J (2012) The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing. In: *IEEE international conference on robotics and automation*.

Zhang L, Lyu S and Trinkle J (2013) A dynamic Bayesian approach to simultaneous estimation and filtering in grasp acquisition. In: *IEEE international conference on robotics and automation*.

Zhang Z, Hsu D and Lee W (2014) Covering number for efficient heuristic-based POMDP planning. In: *International conference on machine learning*, pp. 28–36.