Pre- and Post-Contact Policy Decomposition for Planar Contact Manipulation Under Uncertainty

Michael C. Koval, Nancy S. Pollard, and Siddhartha S. Srinivasa
Robotics Institute, Carnegie Mellon University
{mkoval, siddh, nsp}@cs.cmu.edu

***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 policy.

I. 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 II) has formulated manipulation as a *partially observable Markov decision process (POMDP)* [19] with a reward function that drives the robot towards the goal [13, 15, 16, 40]. Unfortunately, the *contact manipulation POMDP* is intractable for most real-world problems like Fig. 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* 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 [17] 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 [21]. As a result, any contact observation indicates that the object lies on the *contact manifold* [11, 21, 30], the lower-dimensional set of poses for which the object is in non-penetrating contact with the hand.

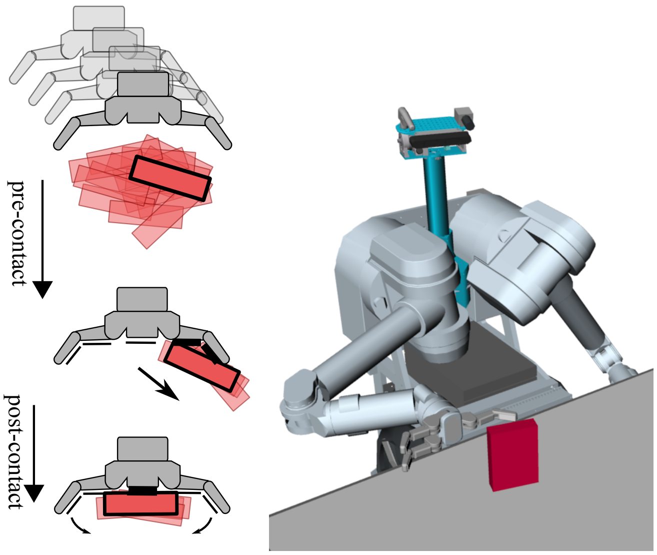


Fig. 1: Robot using real-time tactile feedback to grasp a box under pose uncertainty. The robot begins by executing an open-loop, trajectory (top) towards the object. Once the robot observes contact (middle), it switches to executing a closed-loop policy (bottom) to complete the grasp.

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 a point-based method [37]. 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 makes contact with the object.

In this paper, we specifically consider the task of pushing objects on a planar support surface (Fig. 1) with binary contact sensors for feedback. This problem is a fundamental research topic in manipulation [14, 32, 34] and enables robots to perform a wide variety of tasks that would not be otherwise possible. Pushing enables robots to move objects that are too large or heavy to be grasped [6], for pre-grasp manipulation [5, 20], and to grasp objects under uncertainty [4, 7, 8].

We build on this large body of work by developing a closed-loop pushing action that is robust to large amounts of uncertainty. We demonstrate, through a large suite of simulation experiments, that our uncertainty-aware policy outperforms a baseline policy that makes use of real-time feedback.

We make the following contributions:

Policy Decomposition. We show 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 (Section IV). We introduce a novel algorithm that exploits structure of contact manipulation to efficiently find a provably near-optimal policy.

Post-Contact Policy. We present a method of finding a post-contact policy (Section IV-A) using a point-based POMDP solver [37]. 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 V-C). Our uncertainty-aware policy achieves a successful grasp more quickly and with higher probability than a baseline closed-loop policy.

Real-Time Performance. Decomposing the policy into two stages enables us to perform the computationally expensive calculation of the post-contact policy in an offline pre-computation step. Executing the pre-contact trajectory is entirely open-loop and evaluating the post-contact policy takes milliseconds (Section V-B).

We also discuss several limitations of our work. Key among them is the requirement that the post-contact policy be pre-computed. This is possible for local policies—such as grasping an object or actuating controls—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.

II. Related Work

Early work in contact manipulation focused on sensorless manipulation, where the robot attempts to plan an open-loop trajectory that achieves a task—such as inserting a peg in a hole [31] or localizing an object on tray [9]—despite initial pose uncertainty. More recently, the push-grasp [7, 8] has applied the same approach to grasping by using a long straight-line push to funnel the object into the hand before closing the fingers. These techniques model the problem using non-deterministic uncertainty [24] and use worst-case analysis to guarantee success. In contrast, our algorithm considers probabilistic uncertainty and directly minimizes the time required to achieve the goal. The policies produced by our algorithm leverage real-time sensor feedback to more quickly achieve the goal.

Other research has approached contact manipulation as a control problem by directly mapping observations to actions. Prior work has developed controllers that can locally refine the quality of a grasp [38] or achieve a desired tactile sensor reading [26, 43]. These techniques achieve real-time control rates of up to 1.9 kHz [26] and impressive performance in controlled environments. However—unlike our approach—these algorithms require a high-level planner to analyze the scene and provide a setpoint to the controller.

Recent work, such as this paper, has attempted to combine the advantages of both approaches by formulating the contact manipulation problem as a POMDP [16] and synthesizing a closed-loop policy that can reason about uncertainty. Many of these approaches, such as tactile localization [18, 36], split the problem into an information-gathering stage, which attempts to localize the object, followed by a goal-directed stage. Alternative approaches gather information while executing goal-directed trajectories and dynamically re-plan based on observations received during execution [39, 40]. Our technique improves on these results by eliminating the need for an explicit information-gathering stage. Instead, we naturally gather information during execution and only when it necessary to achieve the goal.

Most recently, Horowitz et al. applied SARSOP [23]—the same point-based POMDP solver [37] we use to to find the post-contact policy—to synthesize a grasp of a lugnut [13]. We generalize this approach to the wider class of contact manipulation problems, including those with long planning horizons, by decomposing the policy. Our pre- and post-contact decomposition closely mirrors the "simple after detection" property of POMDPs observed in aerial collision avoidance [1]. 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.

III. Contact Manipulation Problem

We focus on the class of *contact manipulation* tasks where a robotic manipulator maintains persistent contact with its environment; e.g. 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.

A. POMDP Formulation

We formulate the contact manipulation problem as a partially observable Markov decision process (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 $R : S \times A \to \mathbb{R}$ is the reward function [19].

In a POMDP the agent does not know its true state but instead tracks its belief state $b : S \to [0, 1]$ with $\int_S b(s)ds = 1$, a distribution over S, with a state estimator. The set of all belief states $\Delta = \left\{b : S \to [0, 1] : \int_S b(s)ds = 1\right\}$ is known as *belief space*. The goal is to find a policy $\pi : \Delta \to A$ over

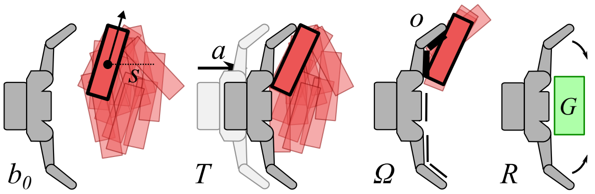


Fig. 2: Contact manipulation POMDP. The robot starts in belief state $b_0 \in \Delta$, takes action $a \in A$, and updates its belief state with observation $o \in O$. The robot's goal is to maximize its reward R by pushing the object into the goal region $G \subseteq S$.

belief space that maximizes the sum of expected of future reward $E[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$ discounted by $\gamma \in [0, 1)$.

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

During contact the motion of the object is modeled by a quasistatic physics simulator [33]. 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 $S = SE(2) \times se(2)$. This approximation has shown to be accurate for the planar manipulation of household objects at relatively low speeds [6, 8].

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 motion of the hand while executing actions, assuming that the noise is independent of the full configuration of the manipulator.

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 [22]. Similar to prior work [12, 18, 21, 22, 36], 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.

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 push the object into a hand-relative goal region $G \subseteq S$ as quickly as possible. We encode G in a reward function

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

that assigns zero reward to the goal and negative reward to all actions. This encourages the robot to quickly move the object into the goal region. However, our approach generalizes to any state-action dependent reward function.

B. Value 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

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

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 [3].

C. Tractability

Optimally solving a POMDP has been shown to be PSPACE-complete [28] and is only tractable for small problems. Even worse, most POMDP solvers operate on discrete state, action, and observation spaces.

Point-based methods [37] are a class of offline solvers that approximate value function by performing backups at a discrete set of belief points. These methods perform well when the initial belief b_0 is known a priori and the *reachable belief space* $\mathcal{R}(b_0) \subseteq \Delta$, the set of beliefs that are reachable from b_0 given an arbitrary sequence of actions and observations, is small.

Unfortunately, the contact manipulation POMDP has a continuous state space that must be discretized for most point-based solvers. Discretizing a 1 m × 1 m region at a 2 cm × 2 cm × 10° resolution, which is the same resolution used in our experimental results, would result in a state space of size $|S| = 90,000$. This is approximately an order of magnitude larger than the problems solved by state-of-the-art point-based methods [23]. More importantly, the resulting policy would only be valid for a single initial belief state b_0 and would need to be recomputed for each problem instance.

Online planning algorithms [42] forgo pre-computing the full value function by 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 tight upper and lower bounds are available to guide the search and ample time is available for action selection.

Unfortunately, performing this search online is intractable 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 [21, 44].

IV. Policy Factorization

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.

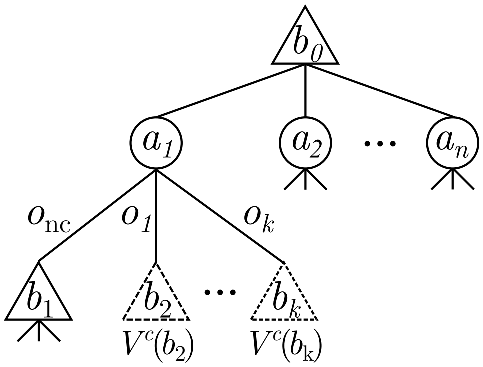
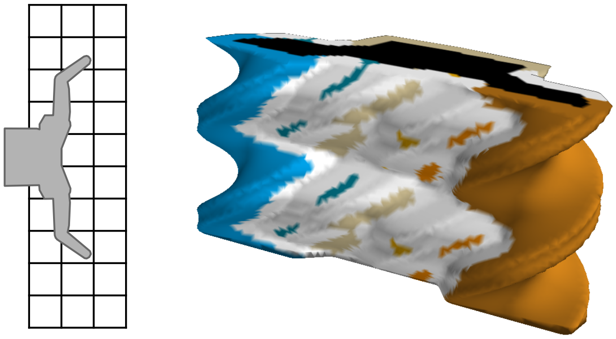


Fig. 3: Online POMDP solvers must branch over both actions and observations. The pre-contact search only branches on actions by evaluating all post-contact belief states with the post-contact value function V^c.



(a) Free Space (b) Contact Manifold

Fig. 4: The state space considered by the post-contact planner is partitioned into (a) free space S_{nc} and (b) the contact manifold S_{c}. Each sensor's contribution to the observable contact manifold S_{o} is labeled with a unique color.

Before observing contact, the robot executes an open-loop *pre-contact trajectory* $\xi \in A \times A \times \dots$ and receives a series of no-contact observations $o_1 = \cdots = o_{t-1} = o_{\text{nc}}$. 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

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

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 ($o = o_{\text{nc}}$).

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 [37] in an offline pre-computation step (Section IV-A). Then, when given a problem instance, we solve for the pre-contact trajectory ξ that is optimal with respect to π^{c} using an online search. As shown in Fig. 3 this is equivalent to truncating an online POMDP search [42] once contact has occurred and using V^{c} to evaluate the value of the truncated subtrees.

A. 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'. At this point—due to the discriminative nature of contact sensors—we know that the object is in non-penetrating contact with one or more contact sensors [21]. Formally, we know that the true state $s \in S$ lies on the *observable contact manifold* $S_{\text{o}} \subseteq S_{\text{c}}$: the subset of the contact manifold S_{c} that is also in contact with one or more sensors [22].

Fig. 4b shows S_{c} and S_{o} for the hand, object, and sensor configuration as depicted in Fig. 2. The contact manifold is a two-dimension structure embedded in $SE(2)$ where the vertical axis represents the orientation of the object relative to the hand. Each color in on the manifold represents a region of S_{o} that is in contact with a unique sensor. White regions of manifold indicate simultaneous contact between the object and multiple sensors.

Since the state is known to lie on S_o, we additionally know that the belief state b' is in *post-contact belief space* $\Delta_o = \{b \in \Delta : b(s) = 0 \ \forall s \notin S_o\}$ and exhibits sparse support. 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 post-contact POMDP is particularly well suited to being solved by a point-based method [25].

1) Discretization: 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 PSPACE-complete and is just as hard as solving the full POMDP [23]. Instead, we define a *trust region* $S_{\text{trust}} \subseteq S$ that we believe to over-estimate the support of $\mathcal{R}^*(\Delta_o)$ and solve the post-contact POMDP over this 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 [33], 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 this choice disallows policies that require large global motions; e.g. performing an orthogonal push-grasp once the object has been localized along one axis.

We compute the discrete transition, observation, and reward functions over S_{trust} by taking an expectation over the continuous models under the assumption that there is a uniform distribution over the underlying continuous states. In practice, we approximate the expectation through Monte Carlo rollouts.

It is important to discretize S_{trust} such that the Markov property still holds in the discretized state space. Unfortunately, uniformly discretizing S_{trust} poorly represents the discontinuous nature of contact [13, 21]: two states in S may be arbitrarily close together, but generate completely different observations depending upon whether the object is in contact with the hand. Instead, we compose the trust region S_{trust} from two components: (1) a uniform discretization of free space $S_{\text{nc}} = S_{\text{trust}} \setminus S_{\text{c}}$ (Fig. 4a) and (2) an explicit discretization of the contact manifold S_{c} (Fig. 4b). We first discretize the contact manifold into a set of orientation iso-contours that are computed using a polygonal Minkowski sum [22]. Then, we discretize the perimeter of each iso-contour into a sequence of equal-length segments. Each segment is a discrete contact state.

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.

2) Initial Belief Points: Unlike in the traditional application of a point-based method—where the prior belief state b_0 is known—we only know that $b_0 \in \Delta_o$. We cannot seed the point-based solver with the full set Δ_o because it is uncountably infinite. Instead, we initialize the solver with with a discrete set of samples $B \subseteq \Delta_c$ similar to the initial beliefs that we expect to see during execution; e.g. Gaussian distributions with means sampled uniformly at random from S_{trust}. This could be later refined by adding the post-contact beliefs observed during execution to B and running additional iterations of point-based value iteration.

B. 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 by running an optimal graph search algorithm, such as A* [10], in an augmented belief space by recursively expanding V in Equation (1) to

$$V(b) = \max_{\xi} \left[\sum_{t=0}^{\infty} \gamma^t \left(\prod_{i=0}^{t} \Omega(o_{\text{nc}}, b_{i+1}, a_i) \right) \left(R(b_{t+1}, a_t) + \sum_{o \in O_c} \Omega(o, b_{t+1}, a_i) V^{\text{c}}(b_{t+1}) \right) \right]. \quad (2)$$

Each term in the summation corresponds to taking a single action in ξ. The product is equal to the probability of reaching time t without having observed contact.

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

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

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

precisely one term in the above summation. The no-contact probability evolves as $t' = t + 1$ and $p'_{\text{nc}} = p_{\text{nc}} \Omega(o_{\text{nc}}, b', a)$ because the Markov property guarantees $o_t \perp o_{t-1} | \, s_t$. When $p_{\text{nc}} = 0$ the cost of executing ξ is $-V(b_0)$. Therefore, finding the minimum-cost ξ is equivalent to finding the optimal value function $V(b_0)$.

Intuitively, the cost of an edge consists of two parts: (1) the immediate reward $R(b', a)$ from taking action a in belief state b and (2) the expected reward $\sum_{o \in O_c} \Omega(o, b', a) V^{\text{c}}(b')$ 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}.

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* [35]. Heuristic functions are typically designed by finding the optimal solution to a relaxed form of the original problem.

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

We do this by solving for the value function of the *MDP approximation* [42] 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

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

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

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

where $R_{\text{max}} = \max_{a \in A, s \in S} R(s,a)$ is the maximum reward and t_{min} is the minimum number of steps required to make contact with the object.[1] We can compute a lower bound on bound on t_{min} as

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

where $\text{dist}(s, s')$ is the straight line distance between two positions of the states, and d_{max} is the maximum displacement of all actions.

This is an upper bound on V^{MDP} because we are over-estimating reward and under-estimating the time required to achieve the reward in an environment with $R(s,a) \leq 0$. Therefore, from the definition of the MDP approximation [42], 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 $x = (b, t, p_{\text{nc}})$.

3) Search Algorithm: We employ weighted A*, a variant of A*, to search the graph for an optimal path to the goal [41]. Weighted A* operates identically to A* but sorts the nodes in the frontier with the priority function

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

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

For $\epsilon_w > 1$, weighted A* is no longer guaranteed to return the optimal path, but the cost of the solution returned is guaranteed to be no more than ϵ_w times the solution cost of the true optimal path [41]. Weighted A* has no bounds on number of expansions, but—in practice—expands fewer nodes that A*. This is beneficial when, such as in our case, it is computationally expensive to generate successor nodes.

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

$$\begin{aligned} ||V_T - V^*||_\infty &\leq \gamma ||V_{T-1} - V^*||_\infty + \gamma P_{\text{max}} ||V^{\text{c}} - V^*||_\infty \\ &\leq \gamma^T ||V_0 - V^*||_\infty + \sum_{t=1}^{T} \gamma^t P_{\text{max}} ||V^{\text{c}} - V^*||_\infty \\ \eta &\leq \gamma^T \eta_{\text{nc}} + \tfrac{\gamma(1-\gamma^T)}{1-\gamma} P_{\text{max}} \eta_{\text{c}} \qquad (3) \end{aligned}$$

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

As expected, Equation (3) shows that $\eta \to 0$ as $\eta_{\text{c}}, \eta_{\text{nc}} \to 0$, the same result as in traditional online search algorithms [42]. However, the post-contact error does not approach zero as $T \to \infty$ because the full policy can never outperform a sub-optimal post-contact policy π^{c}.

V. Simulation Experiments

We evaluated the performance of the policies produced by our algorithm in a suite of simulation experiments. First, we evaluate the performance of the post-contact policies produced for our discretization of the contact manipulation POMDP. Then, we demonstrate that pre-contact search effectively extends the horizon of the post-contact policies.

A. Experimental Setup

We simulated the algorithm in a custom a two-dimensional kinematic environment with polygonal geometry. Each experiment consisted of a BarrettHand pushing a rectangular box with initial pose uncertainty (Fig. 2-Left).

1) Transition Model: We simulated the motion of the object using a penetration-based quasistatic physics simulator [33] with a 2 mm step size. During each update, the finger-object coefficient of friction and the radius of the object's pressure distribution were sampled from a stationary Gaussian distribution. We considered a set of $|A| = 5$ purely translational actions with length $||a_i|| \approx 2$ cm for planning.

2) Observation Model: We simulated binary observations for each of the hand's $n = 7$ links, shown in Fig. 2, by checking collision between the contact sensor and the object. Prior analysis of the observable contact manifold Δ_o revealed that only $|O| = 20$ of the potential 2^7 observations were geometrically feasible. All observations perfectly discriminated between contact and no-contact.

3) State Estimator: We represented the belief state using a set of weighted particles and performed recursive Bayes updates using the manifold particle filter (MPF) [21, 22]. The MPF is a variant of the particle filter that samples from the contact manifold to avoid particle starvation during periods of persistent contact. Using a particle filter enabled us to track the underlying continuous distribution instead of using the discretized models used for planning. At runtime, we discretized the continuous representation of the belief state each time π^{c} was evaluated.

Note that our use of the MPF for state estimation is a choice that we made for practical reasons. In theory, it is more appropriate to use a discrete Bayes filter that exactly matches the belief dynamics used during planning.

[1] We cannot simply use $\text{dist}_{s' \in G}(s, s')$ as the heuristic because it omits the discount factor γ.

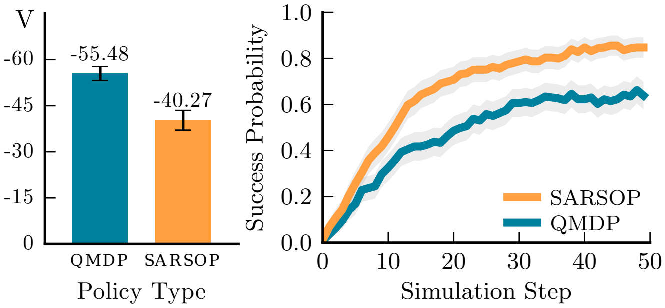
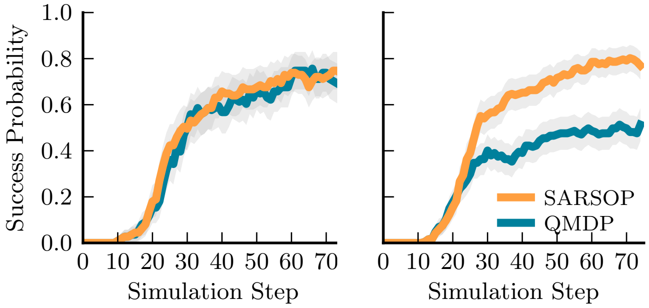


Fig. 5: Left: SARSOP finds a post-contact policy that achieves higher value than the QMDP policy using the discrete belief dynamics. Right: The higher value corresponds to SARSOP achieving success more reliably than QMDP.



(a) All Sensors

(b) Fingertips Only

Fig. 6: (a) SARSOP and QMDP perform equally well when the entire hand is instrumented with sensors. (b) However, SARSOP significantly outperforms QMDP when sensors are only located on the fingertips.

4) Evaluation Metric: We evaluated the *success rate* of the policy by computing the probability $\Pr(s \in G)$ of the object being in the goal region after each timestep. Good policies should quickly achieve reward and, thus, a high success rate.

5) Baseline: We compare the policy generated by our algorithm to a policy computed using the *QMDP approximation*. The QMDP approximation computes an upper-bound on the optimal POMDP value function using the optimal value function of the underlying MDP [29]. QMDP takes advantage of observations during execution and has been shown to perform well in several domains, but does not take information-gathering actions to reduce uncertainty.

B. Post-Contact Policy

We discretized S_{nc} in a 15 cm × 50 cm region around the hand at a coarse 2 cm × 2 cm × 0.2 rad resolution. We discretized an analytic representation of S_{c} at a 2 cm$\times$0.2 rad resolution. The resulting discretization consisted of $|S| = 7238$ states split between $|S_{\text{nc}}| = 5425$, $|S_{\text{c}}| = 1812$, and one unknown state to represent all states that lie outside S_{trust}. We set the discount factor of the discretized post-contact POMDP to $\gamma = 0.99$.

We solved for the post-contact QMDP policy by running 1776 iterations of MDP value iteration on the discrete POMDP. Value iteration converged within an error bound of 10^{-6} in 17.73 seconds. The resulting policy contained one α-vector per action, for a total of $|A| = 5$ α-vectors, and took 48 μs to evaluate on a discrete belief.

We repeated this procedure for the post-contact POMDP policy by running the SARSOP implementation provided by APPL Tookit on $|B| = 50$ initial belief points [23]. Each belief point in B consisted of a Gaussian distribution with its mean in S_{trust}. We ran SARSOP for 10 minutes, during which it sampled 3360 belief points, performed 8700 backups, and was able to bound the optimal value function within 4.26 reward. The final policy consisted of 7608 α-vectors and took 39 ms to evaluate on a discrete belief.

We evaluated the quality of both post-contact policies by performing 250 rollouts using the discrete belief dynamics. Each rollout was initialized with a belief drawn from B and was forward-simulated for 100 timesteps. Fig. 5-Left shows that SARSOP achieved significantly higher average value of $V^{\text{c}} = -40.27$ than QMDP with $V^{\text{c}} = -55.48$. QMDP performs poorly because it is unable to reason about information-gathering actions [29]. This result confirms that it is advantageous to formulate the contact manipulation problem as a POMDP instead of a more computationally efficient MDP or deterministic search.

Next, we repeated 250 rollouts using continuous belief dynamics tracked using a manifold particle filter [21]. The state estimator used an analytic representation of the contact manifold [22] and was configured to use 500 conventional particles, 50 dual particles, and a 10% mixing rate. The robot discretized the continuous state estimate after each timestep, which took an average of 23 ms, and took the action dictated by the discrete QMDP and SARSOP policies. Fig. 5-Right shows that SARSOP's higher-quality policy from the discrete state space translates to a high-quality policy in the continuous state space. SARSOP successfully grasps the object both more quickly and with higher probability than QMDP.

C. Pre-Contact Trajectory

The post-contact policies described above rely on approximate, discrete belief dynamics and are only valid in the small region S_{trust} near the hand. We use the search algorithm described in Section IV-B to extend these policies to a longer horizon using the continuous belief dynamics.

We sampled 100 prior beliefs with $\Sigma^{1/2} = \text{diag}[5\text{ cm}^2, 5\text{ cm}^2, 1.2\text{ rad}^2]$ 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 ξ, we ran a separate weighted A* search for each post-contact policy with a heuristic inflation factor of $\epsilon_w = 2$. The search terminated once a node was expanded that satisfied one of the following criteria: (1) ξ achieved contact with 100% probability, (2) 85% of the remaining belief lied in S_{trust}, or (3) the search timed out after 20 s.

The robot began each trial by executing ξ until it observed contact $o_t \in O_c$ or exhausted ξ by reaching $t > |\xi|$.

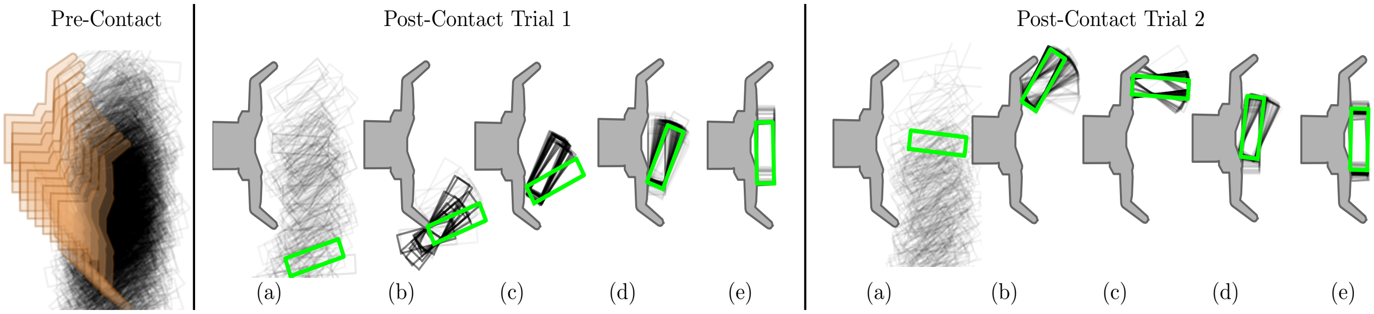


Fig. 7: Two rollouts of a policy produced by the POMDP algorithm. The robot begins by executing the pre-contact trajectory ξ in (a) the initial belief state until (b) contact is observed. Then (c–d) the robot executes the post-contact policy π^c until the object is (e) successfully grasped.

Fig. 7 shows the SARSOP pre-contact trajectory and several snapshots (a)–(e) of the post-contact policy for two different trials. 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.

Fig. 6a shows the success rate of the robot executing the combined pre- and post-contact using its full suite of sensors for feedback. Surprisingly, we found no difference in performance between QMDP and SARSOP: both achieve success rates similar to the performance of SARSOP on the pre-contact policy (Fig. 6). This occurred because the move-until-touch pre-contact trajectory acts as an information gathering action by localizing the object. This allows QMDP—which is optimal in absence of uncertainty—to quickly achieve a successful grasp.

We verified our hypothesis by repeating the same experiment using a hand with inferior sensors. Fig. 6b shows the result of running the experiments with a hand that only has sensors on its fingertips ($n = 2$, $|O| = 2$). As we hypothesized, the performance of QMDP dropped and SARSOP again achieved a significantly higher success probability. This result confirms our intuition that information-gathering actions are most useful when limited sensing resources are available.

VI. Discussion and Conclusion

Our simulation experiments demonstrated that SARSOP achieves higher reward than QMDP in the discrete domain (Fig. 5-Left) used for planning. These results suggests that: (1) information-gathering actions are useful and (2) our adaptive discretization of the state space (Section IV-A1) captures the discriminative nature of contact sensors. Our results in the continuous domain (Fig. 5-Right) mirror those in the discrete domain and suggest that the discrete belief dynamics used for planning are consistent with the continuous belief dynamics tracked by the manifold particle filter [21].

Surprisingly, we found that QMDP performed the same as SARSOP while executing the policy with full sensors because the pre-contact trajectory acted as an information-gathering move-until-touch action. As expected, the performance of QMDP significantly dropped when applied to a hand only equipped with fingertip contact sensors. With this sensor configuration, the SARSOP post-contact policy was able to perform information-gathering actions by moving sideways and forcing the belief state into a fingertip. The QMDP policy, which does not reason about sensing, would never perform this action.

A. Limitations and Future Work

We made several simplifying assumptions to find an efficient solution to the contact manipulation POMDP:

1) Kinematic Feasibility: Our formulation of the contact manipulation POMDP considers state to only be the pose of the object relative to to the hand. As a consequence, a direct implementation of our algorithm does not consider global obstacles or kinematic feasibility while planning. This limitation can easily be addressed for the pre-contact search by simply evaluating the feasibility of each node before expanding it. We believe that we can take a similar approach to π^c by using the pre-computed value function as a heuristic guide an online search in the robot's full configuration space.

2) Dimensionality: 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 $|A|$ and makes the POMDP significantly harder to solve . We believe that the increased computational complexity could be partially addressed by using a sample-based representation of S_c that avoids uniformly discretizing the contact manifold [22]. Another possible solution is to use a continuous POMDP technique [2, 27] to avoid discretizing post-contact POMDP.

Acknowledgments

This work was supported by a NASA Space Technology Research Fellowship. We would like to thank Geoff Gordon, Joelle Pineau, David Hsu, Prasanna Velagapudi, our anonymous reviewers, and the members of the Personal Robotics Lab for their helpful input.

REFERENCES

[1] H. Bai, D. Hsu, M.J. Kochenderfer, and W.S. Lee. Unmanned aircraft collision avoidance using continuous-state POMDPs. In *RSS*, 2011.

[2] H. Bai, D. Hsu, W.S. Lee, and V.A. Ngo. Monte Carlo value iteration for continuous-state POMDPs. In *WAFR*, 2011.

[3] R. Bellman. *Dynamic programming*. Princeton University Press, 1957.

[4] R.C. Brost. Automatic grasp planning in the presence of uncertainty. *IJRR*, 1988.

[5] L.Y. Chang, S.S. Srinivasa, and N.S. Pollard. Planning pre-grasp manipulation for transport tasks. In *IEEE ICRA*, 2010.

[6] M. Dogar and S.S. Srinivasa. A framework for push-grasping in clutter. In *RSS*, 2011.

[7] M. Dogar, K. Hsiao, M. Ciocarlie, and S.S. Srinivasa. Physics-based grasp planning through clutter. In *RSS*, 2012.

[8] M.R. Dogar and S.S. Srinivasa. Push-grasping with dexterous hands: mechanics and a method. In *IEEE/RSJ IROS*, 2010.

[9] M.A. Erdmann and M.T. Mason. An exploration of sensorless manipulation. *IEEE T-RA*, 1988.

[10] P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE G-SSC*, 1968.

[11] K. Hauser and V. Ng-Thow-Hing. Randomized multi-modal motion planning for a humanoid robot manipulation task. *IJRR*, 2011.

[12] P. Hebert, T. Howard, N. Hudson, J. Ma, and J.W. Burdick. The next best touch for model-based localization. In *IEEE ICRA*, 2013.

[13] M. Horowitz and J. Burdick. Interactive non-prehensile manipulation for grasping via POMDPs. In *IEEE ICRA*, 2013.

[14] R.D. Howe and M.R. Cutkosky. Practical force-motion models for sliding manipulation. *IJRR*, 1996.

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

[16] K. Hsiao, L.P. Kaelbling, and T. Lozano-Pèrez. Grasping POMDPs. In *IEEE ICRA*, 2007.

[17] Y.K. Hwang and N. Ahuja. Gross motion planning–a survey. *ACM CSUR*, 1992.

[18] S. Javdani, M. Klingensmith, J.A. Bagnell, N.S. Pollard, and S.S. Srinivasa. Efficient touch based localization through submodularity. In *IEEE ICRA*, 2013.

[19] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. Planning and acting in partially observable stochastic domains. *AI*, 1998.

[20] D. Kappler, L.Y. Chang, M. Przybylski, N. Pollard, T. Asfour, and R. Dillmann. Representation of pre-grasp strategies for object manipulation. In *IEEE-RAS Humanoids*, 2010.

[21] M.C. Koval, M.R. Dogar, N.S. Pollard, and S.S. Srinivasa. Pose estimation for contact manipulation with manifold particle filters. In *IEEE/RSJ IROS*, 2013.

[22] M.C. Koval, N.S. Pollard, and S.S. Srinivasa. Manifold representations for state estimation in contact manipulation. In *ISRR*, 2013.

[23] H. Kurniawati, D. Hsu, and W.S. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *RSS*, 2008.

[24] S.M. LaValle and S.A. Hutchinson. An objective-based framework for motion planning under sensing and control uncertainties. *IJRR*, 1998.

[25] W.S. Lee, N. Rong, and D.J. Hsu. What makes some POMDP problems easy to approximate? In *NIPS*, 2007.

[26] Q. Li, C. Schürmann, R. Haschke, and H. Ritter. A control framework for tactile servoing. In *RSS*, 2013.

[27] Z.W. Lim, D. Hsu, and L. Sun. Monte Carlo value iteration with macro-actions. In *NIPS*, 2011.

[28] M.L. Littman. *Algorithms for sequential decision making*. PhD thesis, Brown University, 1996.

[29] M.L. Littman, A.R. Cassandra, and L.P. Kaelbling. Learning policies for partially observable environments: scaling up. *ICML*, 1995.

[30] T. Lozano-Pèrez. Spatial planning: a configuration space approach. *IEEE T-C*, 1983.

[31] T. Lozano-Pèrez, M.T. Mason, and R.H. Taylor. Automatic synthesis of fine-motion strategies for robots. *IJRR*, 1984.

[32] K.M. Lynch and M.T. Mason. Stable pushing: mechanics, controllability, and planning. *IJRR*, 1996.

[33] K.M. Lynch, H. Maekawa, and K. Tanie. Manipulation and active sensing by pushing using tactile feedback. In *IEEE/RSJ IROS*, 1992.

[34] M.T. Mason. Mechanics and planning of manipulator pushing operations. *IJRR*, 1986.

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

[36] A. Petrovskaya and O. Khatib. Global localization of objects via touch. *IEEE T-RO*, 2011.

[37] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *IJCAI*, 2003.

[38] R. Platt, A.H. Fagg, and R.A. Grupen. Nullspace grasp control: theory and experiments. *IEEE T-RO*, 2010.

[39] R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Pèrez. Belief space planning assuming maximum likelihood observations. In *RSS*, 2010.

[40] R. Platt, L. Kaelbling, T. Lozano-Pèrez, and R. Tedrake. Simultaneous localization and grasping as a belief space control problem. In *ISRR*, 2011.

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

[42] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa. Online planning algorithms for POMDPs. *JAIR*, 2008.

[43] H. Zhang and N.N. Chen. Control of contact via tactile sensing. *IEEE T-RO*, 2000.

[44] L. Zhang and J.C. Trinkle. The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing. In *IEEE ICRA*, 2012.