Planning Robust Manual Tasks in Hierarchical Belief Spaces

Michael W. Lanighan, Takeshi Takahashi, Roderic A. Grupen
College of Information and Computer Sciences
University of Massachusetts Amherst
Amherst, MA 01003, USA
{lanighan,ttakahas,grupen}@cs.umass.edu

Abstract

This paper introduces a technique for planning in hierarchical belief spaces and demonstrates the idea in an autonomous assembly task. The objective is to effectively propagate belief across multiple levels of abstraction to make control decisions that expose useful state information, manage uncertainty and risk, and actively satisfy task specifications. This approach is demonstrated by performing multiple instances of a simple class of assembly tasks using the uBot-6 mobile manipulator with a two-level hierarchy that manages uncertainty over objects and assembly geometry. The result is a method for composing a sequence of manual interactions that causes changes to the environment and exposes new information in order to support belief that the task is satisfied. This approach has the added virtue that it provides a natural way to accomplish tasks while simultaneously suppressing errors and mitigating risk. We compare performance in an example assembly task against a baseline hybrid approach that combines uncertainty management with a symbolic planner and show statistically significant improvement in task outcomes. In additional demonstrations that challenge the system we highlight useful artifacts of the approach—risk management and autonomous recovery from unexpected events.

Introduction

Autonomy in robot systems is a valuable attribute that remains an elusive goal. Noisy sensors, stochastic actions, and variation in unstructured environments all lead to unavoidable errors that can be inconsequential or catastrophic depending on the circumstances. Developing techniques capable of mitigating uncertainty at runtime has, therefore, been a significant and challenging focus of the robotics community. Tasks such as opening doors, picking up objects, or turning valves require adequate levels of situational certainty that varies over instances of the same task and introduces enough risk to jeopardize the task, the robot, and the environment (Atkeson et al. 2015; Correll et al. 2016).

In these tasks, in addition to stochastic actions and observations, there is often uncertainty over object identity and pose. Most work in task planning considers spatial uncertainty given object identities or considers uncertainty over object identities given spatial certainty. Using a traditional flat Partially Observable Markov Decision Process (POMDP) to address both of these problems at once is difficult. As a result, traditional approaches in this setting are prohibitively costly for realistic systems and tasks.

To address this problem, we propose a hierarchical belief space planning framework that manages uncertainty and risk autonomously at different levels of abstraction as runtime situations require. The approach is demonstrated using a mobile manipulator to reliably perform multiple instances of a simple class of assembly tasks.

Related Work

Examples of robot control in the literature often require that all behavior be anticipated by the system designer (Atkeson et al. 2015; Correll et al. 2016) leading to problems when system reliability depends on subtle details of the runtime environment. In these cases, models of expected behavior can be used to compare predicted future states to observations and, thus, to identify cases when plans may fail unexpectedly. Rodriguez *et al.* build empirical models called "grasp signatures" that are used with PCA and a Bayesian SVM to detect grasps that lead probabilistically to failure (Rodriguez et al. 2011). The authors demonstrated an "early abort and retry" recovery technique using Markov chains that reset the system to a fixed state if the predicted grasps are likely to be unsatisfactory. Similarly, Ku *et al.* demonstrate how Aspect Transition Graphs are used to predict when run-time observations are incompatible with the belief state of manipulation tasks and avoids unintended consequences of future actions by aborting execution (Ku et al. 2015). Di Lello *et al.* use Bayesian nonparametric time series via a sticky-Hierarchical Dirichlet Process Hidden Markov Model to detect the unexpected outcomes of actions during execution (Di Lello et al. 2013). However, they do not propose mechanisms for automatic recovery.

Belief Space Planning

In general, robot systems operate under partially observable conditions—the complete state required for making optimal control decisions is not available to any single sensor geometry. In fully observable systems, state estimation is fully determined for each sensor geometry and the dynamics of the underlying state space is assumed to be Markovian. The underlying state space in partially observable systems is a

POMDP (Kaelbling, Littman, and Cassandra 1998), a six-tuple $< S, A, T, R, \Omega, O >$ where: S is a set of states, A is a set of actions, T is a conditional transition probability function between states $\Pr(s'|s, a)$, $R : S \times A \rightarrow \mathbb{R}$ is a reward function, Ω is a set of observations with elements $z \in \Omega$, and O is an observation function, $\Pr(z'|s')$.

In a POMDP, perceptual states cannot be completely differentiated using the current observation alone. Memory over a history of actions and observations is required and in some cases, the problem of perception (or acting on percepts) becomes much harder. Papadimitriou and Tsitsiklis proved that finding exact optimal solutions to POMDP problems is PSPACE-complete and thus, intractable (Papadimitriou and Tsitsiklis 1987). A common approach to approximating solutions to POMDPs at runtime involves belief space planning, which transforms a POMDP into an MDP in belief space (Kaelbling, Littman, and Cassandra 1998). As a consequence, the full range of techniques for solving MDPs can be applied. Although the newly formed *belief state* is fully observable, it is defined over continuous space and thus infinite.

To find solutions in belief space in spite of the state-space explosion, several methods have been proposed. Roy proposed a belief-compression algorithm (Roy, Gordon, and Thrun 2005) that allows planning to be performed in a lower-dimensional belief space than the original belief space. Maximum likelihood approaches maintain distributions over state but act greedily on the most likely underlying state (Platt et al. 2010; Ruiken et al. 2016b). Sampling based techniques have been leveraged to explore belief space efficiently (Hauser 2010; Bry and Roy 2011).

Belief space planning approaches generally combine information gathering with belief condensation to states that solve a task. Heuristic techniques have been used to select actions that address the task while minimizing the impact of uncertainty (Smith and Simmons 2004). "Dual-control" techniques use actions that explicitly reduce uncertainty and actions that explicitly maximize reward (Cassandra, Kaelbling, and Kurien 1996). These approaches work well in settings with state-dependent rewards. Instead of rewards dependent on a particular state-action pair (as is common in POMDPs), ρPOMDPs reward the system with respect to the belief distribution (Araya et al. 2010).

Active Perception and Active Belief

In order to actively improve belief in perceptual feedback, Aloimonos (Aloimonos, Weiss, and Bandyopadhyay 1988) and Bajcsy (Bajcsy 1988) introduced the general framework of active perception. With similar intentions, Denzler et al. demonstrated how to deal with field-of-view limitations and occlusions in a scene by using an information theoretic approach to actively select optimal sensor configurations (Denzler and Brown 2000). Inspired by these approaches, Ruiken *et al.* proposed an *Active Belief Planner* (ABP) (Ruiken et al. 2016b) that relies on models called Aspect Transition Graphs (ATGs) (Sen 2013) to approximate belief transition dynamics. Aspect Graphs were originally introduced in the 70's to represent an object's appearance using multiple viewpoints (Koenderink and Doorn 1979) inspired by the human vision system (Van Effelterre 1994). ATGs can be constructed from extensive, cumulative experience with an object under controlled conditions. Probabilistic constellations of low-level features detected from a single sensor configuration are used to define aspect-nodes. ATGs describe how an agent can interact with an object using stochastic actions to transition between these aspect-nodes.

Models such as these support efficient methods for predicting future states using histories of actions and observations. Consider the Active Belief Planner shown in Algorithm 1. Given a transition model T (such as an ATG), the algorithm takes actions $a \in A$, to optimize a reward function r based on the distribution of belief b over the underlying POMDP states S. Every execution cycle the planner computes the next action a^* that maximizes expected r to a fixed search depth with

$$a^* = \arg\max_{a \in A} \mathbb{E}[r(b, a)].$$

To compute a^*, all actions are considered given the models and b_k, the distribution of belief over states at time k. This computation considers how each action a_k will impact b_{k+1}. The posterior after the control update $\overline{b}_{k+1}$ is computed given the belief of the current state b_k and the transition probability $\Pr(s_{k+1}|a_k, s_k)$ from T (Lines 5-6). Given an expected observation from a forward model through SIMULATEOBSERVATION(s_{k+1}) and an observation function $\Pr(z_{k+1}|s_{k+1})$, the expected posteriors are computed (Lines 8-10). These posteriors are then used to evaluate the metric r (Line 12). The action a_k that maximizes r is then chosen to be executed (Line 13). Algorithm 1 shows a search depth of one. To plan multiple plies in the future, the planner is simply called recursively using the previous posterior belief as the new prior.

Algorithm 1 ACTIVE BELIEF PLANNER: A planner that optimizes reward r based on the expected belief state b_{k+1} over S given $a_k \in A$ using a transition function T to select actions. An optional target distribution G can specify a goal distribution toward which to drive the system. η is a normalization constant, O is an observation function, and z is a simulated observation from a forward model.

1: **function** ABP(S, A, T, O, b_k, r, G)
2: $\quad$ **scores** $\leftarrow \emptyset$
3: $\quad$ **for** *all* $a_k \in A$ **do**
4: $\quad\quad u(a_k) \leftarrow 0$
5: $\quad\quad$ **for** *all* s_{k+1} **do**
6: $\quad\quad\quad \overline{b}(s_{k+1}) = \sum_{s_k} \Pr(s_{k+1}|a_k, s_k) b(s_k)$
7: $\quad\quad$ **for** *all* s_{k+1} **do**
8: $\quad\quad\quad z_{k+1} \leftarrow$ SIMULATEOBSERVATION(s_{k+1})
9: $\quad\quad\quad$ **for** *all* s_{k+1} **do**
10: $\quad\quad\quad\quad b_{k+1}(s_{k+1}) = \eta \Pr(z_{k+1}|s_{k+1}) \overline{b}(s_{k+1})$
11: $\quad\quad\quad u(a_k) \leftarrow u(a_k) + \overline{b}(s_{k+1}) r(b_{k+1}, a_k, G)$
12: $\quad\quad$ **scores**.*append*$(u(a_k))$
13: $\quad$ **return** $\arg\max_{a_k}$(**scores**)

Hierarchical approaches to address POMDPs have been previously investigated (He, Brunskill, and Roy 2011; Kaelbling and Lozano-Pérez 2011), but generally reason over hierarchies of actions within a single planner to reduce planning time. Foka *et al.* investigated using hierarchies of action and state in a navigation domain (Foka and Trahanias 2007) where the robot does not actively alter the environment. In this work, we leverage hierarchies of planners to reduce uncertainty at many levels of abstraction in a general mobile manipulation context using multi-modal feedback. Sridharan *et al.* use a hierarchical formulation but only rely on actions that expose new information (Sridharan, Wyatt, and Dearden 2008). In contrast, the proposed approach considers actions that are both informational and functional—that is they expose new information and accomplish task objectives simultaneously.

Hierarchical Active Belief Planner

We introduce a hierarchical form of the ABP, *Hierarchical-ABP* (HABP). The i^{th} planner in a hierarchy of ABPs of depth d is defined using: S_i, the set of world states; T_i, the conditional transition probability between states; O_i, an observation function; A_i, the set of available actions; b_k, the belief distribution over states $\in S$ at time k; $r_i(b_i, A_i) \rightarrow \mathbb{R}$ a reward function parameterized by the belief distribution and actions; and Z_i, a state abstraction function $Z_i(b_{i-1}) \rightarrow z_i$.

For $i = 0$, a distribution of feature positions and covariances f_0 is computed from raw percepts (in place of b_0). The state abstraction function allows each successive layer of the hierarchy to form observations based on the belief state of the preceding layer. This creates high-level belief distributions that have been stabilized by the lower levels of the hierarchy. Each time step, the hierarchy is updated from the bottom-up and new observations are fused with the existing state using a Bayesian update (Algorithm 3). This fused state is used to plan n-ply forward into the future using T_i and A_i. Future observations $z_{k+[1\cdots n]}$ are estimated using the forward model. This process is outlined in Algorithm 2 and shown graphically for an arbitrary layer of a hierarchy in Figure 1.

Algorithm 2 HABP : Algorithm for updating a hierarchy of depth d at time k and selecting the next action to execute.

function HABP($f_{0,k}$)
  $b_{0,k} \leftarrow f_{0,k}$
  for i in range 1 to d **do**
    $b_{i,k} \leftarrow$ BAYES$(b_{i,k-1}, a_{i,k-1}, Z_i(b_{i-1,k}))$
  for i in range d to 1 **do**
    $a_i^* \leftarrow$ABP$(S_i, A_i, T_i, O_i, b_{i,k}, r_i, G_i)$
    if ARBITER (a_i^*) **then**
      return a_i^*

This approach relies on implementing layers of active belief to cause useful, multi-level transitions to shore up belief that a task specification has been achieved. By managing belief distributions over multi-level abstractions, we investigate how much a particular robot-object interaction will contribute to the task. Each planner in the hierarchy selects actions a^* that maximize the expected reward r_i at level i given actions in set A_i,

$$a^* = \underset{a \in A_i}{\arg\max}\ \mathbb{E}[r_i(b_i, a)].$$

Task-level reward r_d depends on the confidence in lower-level abstractions. If the entropy of the distribution over belief b_{d-1} is high, it will support many different possible observations z_d and, therefore, provide little new information or guidance to the planner at level d. Actions $a \in A_{d-1}$ can improve the precision of the state and, thus, enhance the performance on the task.

Many different strategies exist for coordinating interactions between multiple ABP layers and the external environment that respect the hierarchical description of the task. For example, directing actions into the lowest levels where minimum confidence levels have not been established and then advancing is a reasonable, conservative strategy. This is the approach investigated in this work. In general, these strategies are determined by an ARBITER. How this is achieved

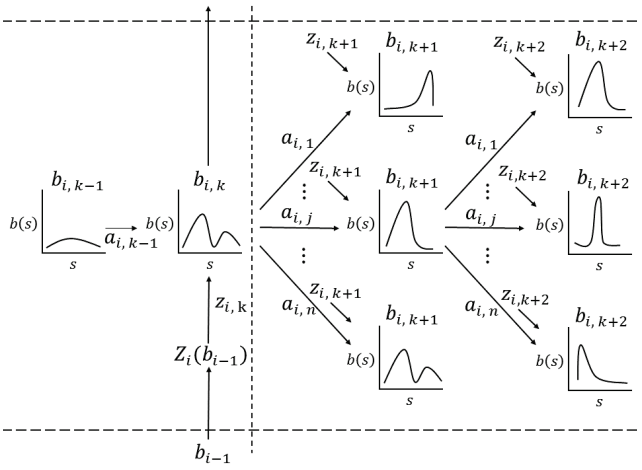


Figure 1: Graphical representation of a layer in the hierarchy. Belief from the lower level and the previous time step state of this level are fused into a current state estimate (left side). This state estimate is used to plan forward into the future using T_i and A_i (right side). Leaf nodes are evaluated using r_i to select a sequence of actions to execute. The first action in this sequence is executed, after which the hierarchy replans with updated state.

Algorithm 3 BAYES FILTER : Algorithm for updating the state of a planner using the previous state b_{k-1}, action a_{k-1}, and new observation z_k. η is a normalization constant.

function BAYES(b_{k-1}, a_{k-1}, z_k)
  for *all* s_k **do**
    $\overline{b_k}(s_k) = \sum_{s_{k-1}} \Pr(s_k|a_{k-1}, s_{k-1}) b(s_{k-1})$
  for *all* s_k **do**
    $b_k(s_k) = \eta \Pr(z_k|s_k)\overline{b_k}(s_k)$
  return b_k

and an example implementation of the hierarchy is described in the following section.

A Two-Layer Assembly Hierarchy

The assembly domain utilizes objects known as ARcubes introduced in (Ruiken et al. 2016a). ARcubes are rigid cubes whose size can be adjusted to meet the requirements of the task. Each of the six faces of the cube is marked with a single ARtag.[1] The natural sparseness of features on any one cube leads to a large degree of ambiguity with respect to a set of ARcubes. We use Kalman Filters to track ARtag positions in $\mathbb{R}^3$.

Recent work has started to utilize ARcubes to form simple assemblies such as towers. In that work, a symbolic planner manipulated symbols grounded in *belief* by a belief space planner to resolve action pre-conditions and resource constraints in unstructured environments (Takahashi, Lanighan, and Grupen 2017). This system demonstrated how symbols grounded in belief lead to more reliable solutions than using maximum likelihood assumptions alone. However, assembly-level actions were not risk compensated. As a result, unexpected outcomes occurred when objects were not placed precisely in the assembly. Without pro-active management of uncertainty or special purpose recovery mechanisms, these outcomes require external resets of the system.

In this example domain, we use a two-layer hierarchy. Belief over the assembly state is managed in the top layer of the hierarchy and belief over objects is managed at the lower level. Each control step, the belief at each level in the hierarchy is updated and each layer plans n-plies into the future after which an ARBITER selects a sequence of actions to execute (as per Algorithm 2). The first action in this n-action sequence is executed, after which the hierarchy re-plans with updated state. Details of the implemented hierarchy and ARBITER are described in the following sections.

Object Level

The bottom of the two-layer hierarchy manages noisy interactions with the environment using ATGs for ARcube as forward models. Given a model-set M, the ATG for object $m \in M$ is a directed multi-graph $G_m = (\mathcal{X}_m, \mathcal{U}_m)$ where $\mathcal{X}_m$ is a set of aspect nodes and $\mathcal{U}_m$ is a set of actions that represent edges in the graph. Edges encode the transition probability T_1 between states. An aspect node $s \in \mathcal{X}$ consists of a set of features, $s = (f_1^{obj}, f_2^{obj}, \cdots)$, that can be observed from a particular sensor configuration. A feature f^{obj} is a tuple of feature id, ξ, and its location in the object frame described using a Gaussian distribution, $f^{obj} = (\xi, \mathcal{N}(\mu, \Sigma))$ where $\mu \in \mathbb{R}^3$ and $\Sigma \in \mathbb{R}^{3\times3}$. During the task execution, the robot updates belief distributions over aspect nodes s for each hypothesis. A "hypothesis" is a spatially constrained volume in $\mathbb{R}^3$ in which distributions of belief over multiple object models are maintained. b_1 are belief distributions over ATG aspect nodes, $b_1 = [b(s_0), b(s_1), \cdots, b(s_{|S_1|})]$ where $S_1 = \bigcup_m \mathcal{X}_m$ and $m = 1, 2, \cdots, |M|$. Each aspect node defines an object frame for its parent ATG.

[1] https://artoolkit.org/

ATGs for ARcubes include parameterized mobility and manipulation actions $\in A_1$ that change the robot's relative position and orientation to features of the object. Manipulation actions can be prehensile or non-prehensile and can create in-hand rotations that re-orient the object (depending on the action parameters and object identity). By discretizing these parameter spaces we can define 4 prehensile and 1 non-prehensile manipulation actions and 7 mobility actions. When selecting actions, the maximum likelihood object coordinate frame is used to parameterize actions. Actions at level 1 accrue belief over ATGs that supports classification and recognition over the history of observations. Observations z_1 are produced with the state abstraction function $Z_0(\{f_1 \cdots f_n\}) \rightarrow z_1$, which turns environmental features into candidate aspects. To perform this transformation a generalized Hough transform is used. This scores candidate aspects from the ATG model-set and forms a belief distribution over candidate aspect-nodes (Ruiken et al. 2016b). Using these observations, the planner can reason about how to manage uncertainty in belief distributions at this level.

We use task partitions similar to Ruiken *et al.* (Ruiken et al. 2016a) to encode *find* tasks for the robot at the object level. The belief that hypothesis h belongs to a target task partition c_{target} with goal state s_j exceeds belief threshold β is defined

$$b(c_{target}) > \beta | c_{target} = \{s_j | \mathbb{1}(s_j) = 1\}. \quad (1)$$

where c_{target} is the target partition—the subset of objects in the model space that satisfy task requirements—and $\mathbb{1}(s_j)$ is an indicator function that evaluates whether $s_j \in c_{target}$. The remainder of the state space populates the non-target partition defined $c_{non-target}$.

Find tasks drive the robot to reduce uncertainty among object models that do and do not support a task. These classes are defined by the goal of an assembly, *e.g.* cubes with a '2' feature and cubes without. By using Information Gain (IG) as r_1, the robot can take actions to condense belief on subsets of the model-set effectively. IG in this setting is defined as:

$$r_1 = IG = H(C_k) - H(C_{k+1}|a_k) \quad (2)$$

where $C = \{c_{target}, c_{non-target}\}$.

Assembly Level

In the top layer of the hierarchy, uncertainty in the spatial precision of the assembly is managed. This is achieved by maintaining belief distributions over positions of goal features in the environment. This defines $s_2 \in \mathbb{R}^3$. In this implementation, Z_2 samples the maximum likelihood state of b_1 to "observe" the expected positions of these features on objects in the environment. This creates observations $z_2 = \{p_1, \cdots, p_h\}$ where $p_i \sim \mathcal{N}(\mu, \Sigma)$, with $\mu \in \mathbb{R}^3$, the mean position and $\Sigma \in \mathbb{R}^{3x3}$, reflecting positional uncertainty of these features on the maximum likelihood object of each hypothesis in the environment. Z_2 only allows object belief to be lifted to this level if they exceed a belief threshold (80%). At the assembly level, actions $\in A_2$ consist of actions that orient and PICK-AND-PLACE objects in the environment based upon the maximum likelihood ATG. This

is achieved by solving a shortest path problem in this ATG using the negative log of the transition probability as cost. Additionally two special actions: a NOP and FIND action are included $\in A_2$. NOP allows the robot to stop if it *believes* the task has been completed. FIND is returned if not enough state exists in b_1 to solve the task. A forward model supports reasoning over the geometric effects of actions $\in A_2$ given b_2 and provides $\Pr(s_{2,k+1}|s_{2,k}, a_{2,k})$, the transition probability T_2 of these actions. Given z_2, we compute the observation probability $\Pr(z_{2,k+1}|s_{2,k+1})$ with empirical models of robot performance of PICK-AND-PLACE actions $\in A_2$. We assume these models are Gaussian $\mathcal{N}(\mu, \Sigma)$. The continuous state introduces minor changes to Algorithm 1, which assumes discrete state. In particular, the belief update (Lines 5-10) is computed with

$$b(s_{2,k+1}) = \eta \Pr(z_{2,k+1}|s_{2,k+1}) \int_{s_{2,k}} \Pr(s_{2,k+1}|s_{2,k}, a_{2,k}) b(s_{2,k}), \quad (3)$$

using the same variables as Algorithm 1.

To quantify performance at the assembly level, we reason over how actions will decrease the Kullback-Leibler divergence D_{KL} (Kullback and Leibler 1951) between the belief distribution $b_{2,k}$ and a goal distribution G. G defines the goal positions of target features in the assembly $(f_1^{goal}, f_2^{goal}, \cdots)$ and specifies the amount of acceptable uncertainty at the goal given empirical models of the robot's performance $\mathcal{N}(\mu, \Sigma)$. To measure this divergence, we use error distributions E computed using the ℓ_2 distance between goal feature positions and feature positions given the belief distribution $b_{2,n}$, given action $a_{2,k}$, and G. This yields the following reward function r_2

$$r_2 = -D_{KL}(E \parallel G) = -\int_{-\infty}^{\infty} e(x) log(\frac{e(x)}{g(x)}) dx \quad (4)$$

where e and g represent the densities of E and G respectively. When the D_{KL} between the current belief state and the goal is less than the D_{KL} between the expected belief state at time $k + l$ ($l \in [0, n]$) and the goal, the planner selects NOP to indicate that belief has been condensed as much as possible (with regard to the performance models of the robot). To bias the robot toward faster solutions (at the cost of precision), the reward function is penalized by the cost of actions. In this implementation, PICK-AND-PLACE actions have a uniform cost of $\lambda = 1.0$, yielding the following reward function,

$$r_2 = -D_{KL}(E \parallel G) - \sum_{i}^{k+l} \lambda_i. \quad (5)$$

Control Authority

In a hierarchy of planners with limited sensor and effector resources a decision regarding control authority must be made at each control step by an ARBITER to determine which planner's prescribed action will be executed in order to optimize the task. In this implementation, control authority is governed by the assembly-level planner. If this planner returns a plan, it is executed. If the planner returns a NOP, the system stops as the robot has condensed belief to a solution of the task. If the top-level planner returns FIND, then the actions prescribed by the object-level planner are selected.

This control structure enables an intrinsically *lazy* behavior for assembly tasks. If not enough state is present in the assembly-level planner, then the object-level planner is leveraged to uncover the needed state from the world. By relying on the assembly level when enough state exists to find a solution to the task the robot attends to the minimum amount of world state required to solve a task. This helps increase tractability and reduce the dimensionality of problems—as the robot will not actively uncover state regarding objects that are not required by the task.

Example Application: Simple Assemblies

To highlight the approach, we conducted three experiments involving simple assembly tasks. Assemblies are specified by configurations of features afforded by ARcubes. The robot is provided with a set of 20 ARcubes modeled as ATGs. In each experiment, the raw materials (ARcubes) needed to construct the assembly are present, although their poses and identities are initially unknown to the robot. The first experiment compared the overall error at the conclusion of a tower assembly using the proposed method to a baseline system used in (Takahashi, Lanighan, and Grupen 2017). This baseline system lifts objects in the environment to symbolic representations and then executes a plan prescribed by a symbolic planner. The second experiment demonstrates the flexibility of the approach by constructing a pyramid of blocks. This assembly challenges the system as previously placed blocks can be disturbed when placing additional blocks. The third experiment highlights the system's ability to overcome uncertainty induced by an external source. This is demonstrated by having a researcher intentionally alter the outcome of an action during a pyramid assembly.

Experiments were conducted using the uBot-6 mobile manipulator (Ruiken, Lanighan, and Grupen 2013) shown in Figure 2. uBot-6 is a toddler-sized mobile manipulator that balances dynamically on two wheels. It has 13 total degrees of freedom (DOF), including two 4 DOF arms and a rotatable trunk which provide a large bimanual workspace. Visual and depth data are provided by an ASUS Xtion RGB-D camera located on the head. In addition to proprioceptive data at each actuated joint, the robot has one six axis force-torque sensor in each hand that provides haptic feedback. Closed loop controllers are used in all actions, based on expectations from the currently observed belief state. These controllers achieve their objectives by following gradients in a convex potential function $\phi(\sigma)$ with respect to changes in the value of the motor resources where σ are sensory inputs (Huber, MacDonald, and Grupen 1996). It should be noted that these expectations will not necessarily exactly match actual action outcomes as this is a real system. Convergence of these controllers does not imply completion of tasks since the goal of the potential field is influenced by noise from sensory inputs (joint angles, feature locations, etc...) and localization errors. For example, if we have 0.03 m in local-

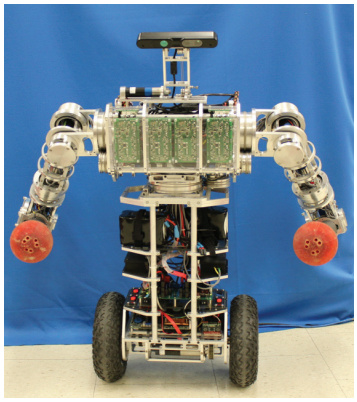


Figure 2: uBot-6: a 13 degree of freedom, dynamically balancing whole body mobile manipulator used in the experiments.

ization error, the controllers may converge to a goal offset 0.03 m from the ground truth goal. It is not possible to know this offset at run-time. In ideal environments (such as in simulation) we may not have these issues. Our system aims to minimize the impact of these unavoidable sources of noise in the real world by considering the belief of states in addition to minimizing errors in controller and perceptual units.

For placing actions, 120 trials were conducted by the robot placing objects from the ARcube model-set yielding a Gaussian distribution with

$$\mu = (0.03345256, -0.04095612)\ (m)$$

$$\Sigma = \begin{bmatrix} 0.00324572 & 0.00073537 \\ 0.00073537 & 0.00235752 \end{bmatrix}\ (m^2)$$

This distribution is used to evaluate sampled future states and to compute $\Pr(z_{2,t+1}|s_{2,t+1})$ in Algorithm 1.

Results

Comparison to Baseline Planner

The first experimental task required the robot to construct a tower consisting of two ARcubes. In this assembly, the bottom cube was required to have a "2" tag in front and a "1" tag on top. The top cube was required to have a "3" tag in front and a "0" tag on top. The features were required to be located in specific positions relative to a fixed frame defined by 3 environmental tags ("A", "B", and "C") as seen in Figure 3. We compared the proposed method with a baseline method (Takahashi, Lanighan, and Grupen 2017). When using the baseline, the system would fail when action outcomes did not match the expectations of the planner.

We performed the experiment five times for both methods. The robot successfully built the assembly one out of five times with the baseline and five out of five times with HABP. The final assemblies using both approaches are shown in Figure 3. The average of the final assembly errors and the results of a T-test are shown in Table 1. The p-value of the error is less than 0.05, which shows that HABP has statistically significant better outcomes than the baseline system.

Approach	Final Error $(m) \pm$ variance (m^2)
Baseline	0.349 ± 0.0157
HABP	$0.071 \pm 4.05\,\mathrm{E}{-5}$
T-test	$p = 0.011 < 0.05$

Table 1: The mean (m) and variance (m^2) of final error ($\sum \ell_2$ distance) in tower experiments (shown in Figure 3) using the baseline and the hierarchical framework (HABP). As $p < 0.05$, the results show statistically better performance compared to the baseline.

Due to stochastic actions, during several trials the robot misplaced an object in the assembly. If this occurred in the baseline, the robot would often fail to successfully place the second object, causing it to fall to the floor. With HABP, the robot successfully suppresses this error by reasoning about the likelihood of achieving the goal by re-placing poorly placed objects. Using the HABP approach, the robot does not take further actions when the risk of damaging the current assembly with continued interaction is sufficiently high and chooses to maintain the current world state. This risk management artifact is intrinsic to the planner and does not need to be directly specified or managed externally.

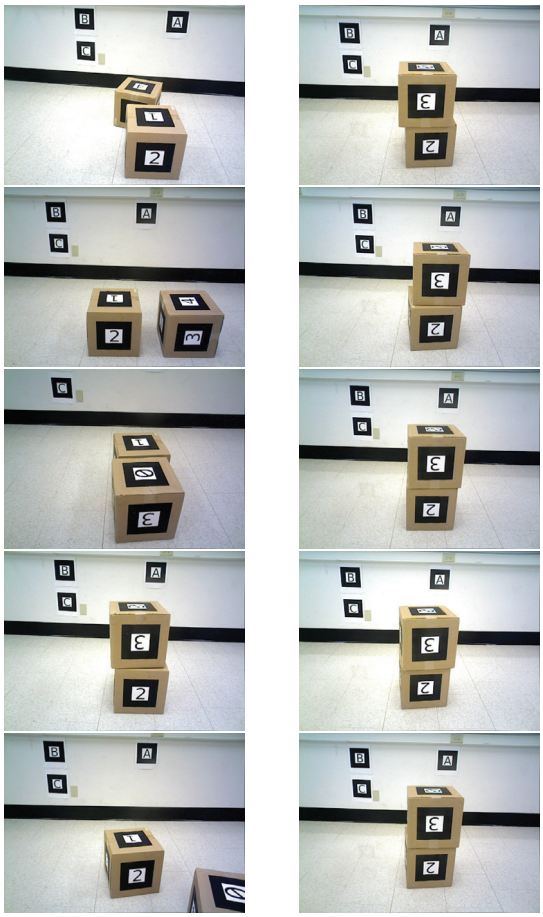
Pyramid Assembly

The second experiment was designed to demonstrate the flexibility of the approach by constructing a pyramid of three blocks. This assembly requires the two bottom objects to be in close contact with each other to provide enough support for the top block. This close proximity can lead to induced errors during assembly as the first block can be unintentionally disturbed when placing the second block. The goal required the lower left block to have a "4" tag on front and a "2" tag on top, the lower right block to have a "7" tag on front and a "1" tag on top, and the top block to have a "3" tag on top and a "0" tag on top. Goal positions were once again specified relative to a fixed frame defined by 3 tags ("A", "B", and "C").

The only change between these runs and the towers in the previous experiment is the specification of the overall goal—the configurations of the ARcube features in specific positions. Using the proposed approach, the robot is able to construct low-error approximations of the goal assembly, while recovering from induced errors during assembly. This error recovery artifact (similar to recovering from topples in the first experiment), does not need to be specified or managed externally and arises naturally from our approach. Snapshots from the robot's perspective of assembly progress for a pyramid assembly are shown in Figure 4.

Rejecting External Disturbances

Another useful artifact that arises from our approach is best highlighted when an external adversarial agent intentionally alters the outcome of the robot's action. With the HABP architecture, the robot recovers autonomously from such unexpected outcomes without additional recovery mechanisms. This process is illustrated in Figure 5. In this demonstration, the robot is constructing a pyramid similar to those in



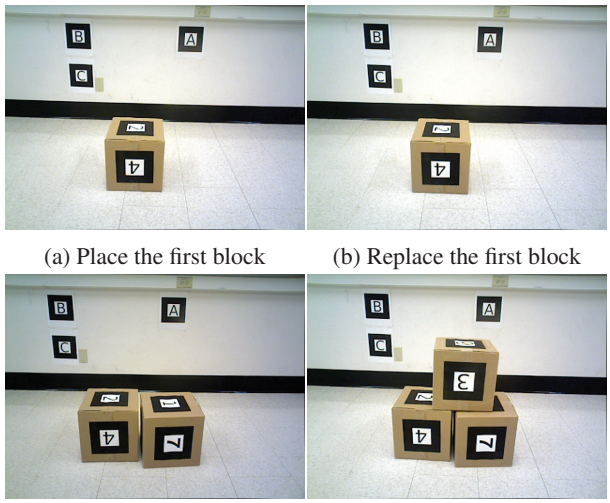
(a) Baseline Outcomes (b) HABP Outcomes

Figure 3: Final tower assemblies produced by the two approaches in the first experiment (comparing to a baseline). On the left are the assemblies using the baseline approach. On the right are the outcomes using the proposed hierarchical belief space framework.

the second experiment. After placing the first object (Figures 5a-5b), a researcher alters the outcome of a placing action by flipping the box the robot just placed (Figures 5c-5d). The robot observes this outcome in Figure 5e. Due to the unexpected transition, the robot is no longer confident it has satisfied the sub-task (placing the object in the correct orientation). As such, the planner (correctly) selects to flip the cube back to the correct orientation (Figures 5f-5g). The robot then replaces the object in the correct position in the assembly (Figures 5h-5k). After observing this satisfactory outcome (Figure 5k), the robot then proceeds to the remaining objects (Figure 5l) and completes the assembly (not shown).

Discussion and Conclusion

Assembling blocks in unstructured environments provides an ideal domain for investigating the sensitivity of performance to undetected or hidden state. Small errors introduced



(a) Place the first block (b) Replace the first block

(c) Place the second block (d) Place the third block

Figure 4: Example pyramid assembly using the proposed approach. The sequence starts in the upper left and progresses left to right, top to bottom.

through stochastic actions early on in an assembly can be magnified as more blocks are placed. By framing the problem in a hierarchical belief space, we can address errors during execution without additional recovery structure. Currently, we only consider actions that place objects at goal locations. This excludes any actions that would disassemble beyond one step. Future work will investigate allowing actions that place objects at arbitrary positions in addition to goals to consider the impact a partial (or possibly full) disassembly would have on the task. This can be important if the robot (or external agent) induces errors in previously placed objects that are inaccessible from the current state.

Online error recovery is an open problem that must be addressed before robots can interact reliably with unstructured environments. In complex tasks that employ tens or hundreds of actions in a sequence, a single fault can disable the robot and/or damage the environment. In order to embed autonomous robots in unstructured worlds, it seems reasonable to specify a reliability in excess of thousands of control decisions between failures, where a "failure" denotes a situation that requires external reset. To meet these goals, breakthroughs are required concerning the assessment of uncertainty—specifically as it puts a task at risk—and in the formulation of risk averse and error recovery behavior.

The hierarchical belief space framework proposed in this paper is a promising direction to meet these goals. We showed that our approach performs statistically significantly better than a baseline approach when constructing simple assemblies. Although we demonstrated that our approach is capable of solving additional assemblies (such as the pyramid) we did not collect sufficient data to draw more sweeping conclusions. Quantifying the performance of our approach in these settings will be a goal of future work. The experiments and demonstrations in this paper highlight that the robot can recover from errors at run-time without ad-

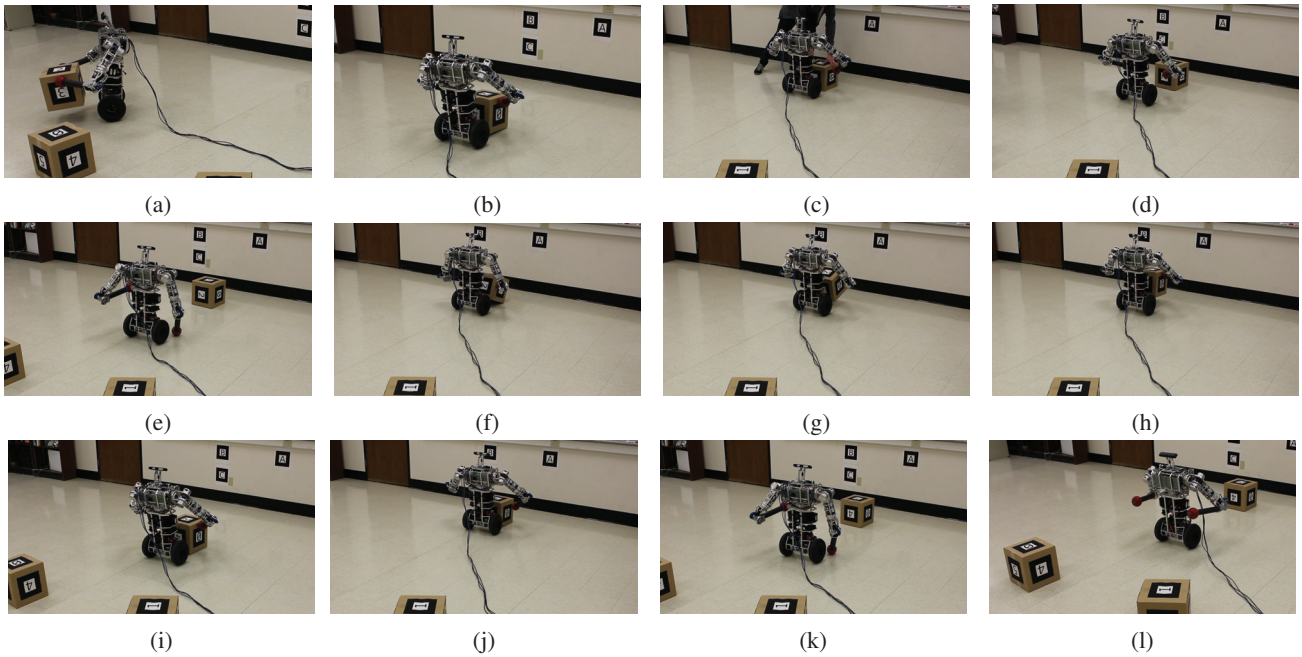


Figure 5: Outcomes of intentional disturbance by an adversarial agent during a pyramid assembly. The robot begins to construct the assembly (a-b). The disturbance is introduced in (c), after which the robot is no longer confident it has placed the object in the correct orientation (d-e). The planner reduces this uncertainty (f-h). After the uncertainty has been addressed, the robot replaces the object in the assembly in the correct orientation (i-k) then proceeds to complete the rest of the assembly (l).

ditional task or recovery structure by simply taking actions that condense belief towards goal distributions. By structuring the task hierarchically, the robot is able to reason and address errors at multiple levels of precision, from independent objects to the whole assembly. These results indicate that it is possible to subsume error recovery, uncertainty and risk, and task planning in a uniform framework with hierarchies of belief space planners, enabling more robust autonomous robots.

Acknowledgments

The authors would like to thank members of the Laboratory for Perceptual Robotics at UMass Amherst for their support. This material is based upon work supported under Grant NASA-GCT-NNX12AR16A. Any opinions, findings, conclusions, or recommendations expressed in this material are solely those of the authors and do not necessarily reflect the views of the National Aeronautics and Space Administration.

References

Aloimonos, J.; Weiss, I.; and Bandyopadhyay, A. 1988. Active vision. *International journal of computer vision* 1(4):333–356.

Araya, M.; Buffet, O.; Thomas, V.; and Charpillet, F. 2010. A pomdp extension with belief-dependent rewards. In *Advances in Neural Information Processing Systems*, 64–72.

Atkeson, C.; Babu, B.; Banerjee, N.; Berenson, D.; Bove, C.; Cui, X.; DeDonato, M.; Du, R.; Feng, S.; Franklin, P.; et al. 2015. No falls, no resets: Reliable humanoid behavior in the DARPA robotics challenge. In *2015 IEEE-RAS Humanoids*, 623–630. IEEE.

Bajcsy, R. 1988. Active perception. *Proceedings of the IEEE* 76(8):966–1005.

Bry, A., and Roy, N. 2011. Rapidly-exploring random belief trees for motion planning under uncertainty. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 723–730. IEEE.

Cassandra, A. R.; Kaelbling, L. P.; and Kurien, J. A. 1996. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *IEEE/RSJ IROS*.

Correll, N.; Bekris, K. E.; Berenson, D.; Brock, O.; Causo, A.; Hauser, K.; Okada, K.; Rodriguez, A.; Romano, J. M.; and Wurman, P. R. 2016. Analysis and observations from the first Amazon Picking Challenge. *IEEE T-ASE*.

Denzler, J., and Brown, C. 2000. Optimal selection of camera parameters for state estimation of static systems: An information theoretic approach. *University of Rochester Tech. Rep* 732.

Di Lello, E.; Klotzbucher, M.; De Laet, T.; and Bruyninckx, H. 2013. Bayesian time-series models for continuous fault detection and recognition in industrial robotic tasks. In *2013 IEEE/RSJ IROS*, 5827–5833. IEEE.

Foka, A., and Trahanias, P. 2007. Real-time hierarchical POMDPs for autonomous robot navigation. *Robotics and Autonomous Systems* 55(7):561–571.

Hauser, K. 2010. Randomized belief-space replanning

in partially-observable continuous spaces. In *Algorithmic Foundations of Robotics IX*. Springer. 193–209.

He, R.; Brunskill, E.; and Roy, N. 2011. Efficient planning under uncertainty with macro-actions. *JAIR* 40:523–570.

Huber, M.; MacDonald, W. S.; and Grupen, R. A. 1996. A control basis for multilegged walking. In *IEEE International Conference on Robotics and Automation*.

Kaelbling, L. P., and Lozano-Pérez, T. 2011. Hierarchical task and motion planning in the now. In *2011 IEEE ICRA*, 1470–1477. IEEE.

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

Koenderink, J. J., and Doorn, A. v. 1979. The internal representation of solid shape with respect to vision. *Biological cybernetics* 32(4):211–216.

Ku, L. Y.; Ruiken, D.; Learned-Miller, E.; and Grupen, R. 2015. Error detection and surprise in stochastic robot actions. In *2015 IEEE-RAS Humanoids*, 1096–1101. IEEE.

Kullback, S., and Leibler, R. A. 1951. On information and sufficiency. *The annals of mathematical statistics* 22(1):79–86.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of markov decision processes. *Mathematics of operations research* 12(3):441–450.

Platt, R.; Tedrake, R.; Kaelbling, L.; and Lozano-Pérez, T. 2010. Belief space planning assuming maximum likelihood observations. In *Proceedings of the Robotics: Science and Systems*.

Rodriguez, A.; Mason, M. T.; Srinivasa, S. S.; Bernstein, M.; and Zirbel, A. 2011. Abort and retry in grasping. In *2011 IEEE/RSJ IROS*, 1804–1810. IEEE.

Roy, N.; Gordon, G. J.; and Thrun, S. 2005. Finding approximate pomdp solutions through belief compression. *JAIR*.

Ruiken, D.; Liu, T. Q.; Takahashi, T.; and Grupen, R. A. 2016a. Reconfigurable tasks in belief-space planning. In *16th IEEE-RAS Humanoids*.

Ruiken, D.; Wong, J. M.; Liu, T. Q.; Hebert, M.; Takahashi, T.; Lanighan, M. W.; and Grupen, R. A. 2016b. Affordance-based active belief recognition using visual and manual actions. In *IEEE/RSJ IROS*.

Ruiken, D.; Lanighan, M. W.; and Grupen, R. A. 2013. Postural modes and control for dexterous mobile manipulation: the UMass uBot concept. In *13th IEEE-RAS Humanoids*, 721–726. IEEE.

Sen, S. 2013. *Bridging the gap between autonomous skill learning and task-specific planning*. Ph.D. Dissertation, University of Massachusetts Amherst.

Smith, T., and Simmons, R. 2004. Heuristic search value iteration for POMDPs. In *20th UAI*, 520–527. AUAI Press.

Sridharan, M.; Wyatt, J. L.; and Dearden, R. 2008. HiPPo: Hierarchical POMDPs for planning information processing and sensing actions on a robot. In *18th ICAPS*. AAAI Press.

Takahashi, T.; Lanighan, M. W.; and Grupen, R. A. 2017. Hybrid task planning grounded in belief: Constructing physical copies of simple structures. In *27th ICAPS*. AAAI Press.

Van Effelterre, T. 1994. Aspect graphs for visual recognition of three-dimensional objects. *Perception* 23(5):563–582.