Robust Simple Assembly via Hierarchical Belief Space Planning

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

***Abstract*—In this paper, we discuss a hierarchical belief space system capable of supporting task planning while simultaneously managing uncertainty in a uniform framework. A demonstration of the approach is conducted with a mobile manipulator performing simple assemblies in unstructured environments.**

I. Introduction

In general, robot systems operate under *uncertainty*. The sensitivity of performance to undetected or hidden state is demonstrated by state of the art systems [1, 3]. 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.

We introduce a Hierarchical Belief Space framework to manage uncertainty over multiple levels of abstraction and demonstrate the approach in an assembly domain. By utilizing object level and assembly level abstractions, the proposed framework can create plans that are capable of both accomplishing the task and managing uncertainty at runtime in a uniform manner.

II. Related Work: Belief Space Planning

The underlying state space in partially observable systems is a Partially Observable Markov Decision Process (POMDP). Papadimitriou and Tsitsiklis proved that finding exact optimal solutions to POMDP problems is PSPACE-complete and thus, intractable [9]. A common approach to approximating solutions to POMDP problems at run-time transforms a POMDP to a Markov Decision Process (MDP) in belief space [7, 13].

Belief space planning techniques generally combine information gathering with belief condensation to states that solve a task. Several belief space planning methods have been used to address POMDPs. Maximum likelihood approaches maintain distributions over state but act greedily on the most likely underlying state [10]. Heuristic techniques have been used to address the task while minimizing the impact of uncertainty [14]. "Dual-control" techniques employ two types of actions: actions that reduce uncertainty and actions that maximize reward [2].

The belief space planning problem is defined $< B, A, \tau, r >$ where B is the set of distributions of belief over underlying states S, A is the set of available actions, τ is the set of conditional transition probabilities between belief states, and $r : B \times A \rightarrow \mathbb{R}$ is a reward function. Generally, these approaches select actions a^* to maximize the reward r,

$$a^* = \underset{a \in A}{\arg\max}\ r(b, a) \tag{1}$$

Ruiken *et al.* introduced an Active Belief Planner [13] that employs sets of models called Aspect Transitions Graphs (ATGs) [8]. ATGs summarize an agent's cumulative interactions with an object to solve object identification tasks. Ruiken *et al.* demonstrated how the same planner can condense belief to target partitions of the state space [12]. By so doing, they were able to encode common tasks such as *recognizing*, *finding*, and *orienting* objects as goal subsets of those partitions.

Hierarchical approaches to address POMDPs have been investigated [5, 6], but generally concern hierarchies of actions—that is they organize actions hierarchically to reduce planning time. Foka *et al.* investigated using hierarchies of action and state in a navigation domain [4]. 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.

III. Hierarchical Belief Space

In this work, we introduce a hierarchical form of the ABP. In a hierarchy of depth n, the i^{th} planner is defined: $< B_i, A_i, \tau_i, r_i, Z_i >$ where B_i is the set of distributions of belief over underlying states S, A_i is the set of available actions, τ_i is the set of conditional transition probabilities between belief states, $r_i(B_i, A_i) \rightarrow \mathbb{R}$ is a reward function parameterized by current belief distributions and actions, and Z_i is a state transformation function $Z_i(b_i) \rightarrow z_{i+1}$ where $b_i \in B_i$. For $i = 0$, f_0, a distribution of feature positions and covariances, is computed by a perceptual front-end in place of b_0. The state transformation function allows each successive layer of the hierarchy to form observations based on the belief state of the preceding layer. This creates higher-level belief distributions that have been stabilized by the lower levels of the hierarchy. A depth two hierarchy is shown in Figure 1.

Each planner in the hierarchy selects actions a^* that maximize

$$a^* = \underset{a \in A_i}{\arg\max}\ r_i(b_i, a),$$

the reward r_i at level i given actions in set A_i. Task-level reward r_n depends on the confidence in lower-level abstractions. If the entropy of the distribution over belief b_{n-1} is high, it will support many different possible observations z_n

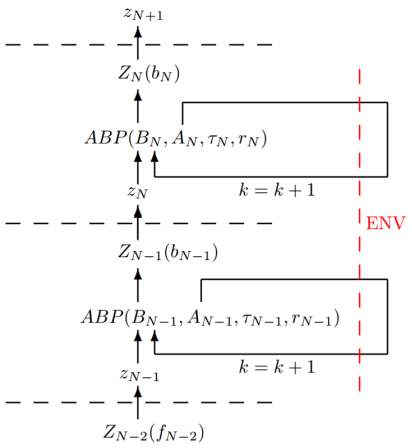

Fig. 1: Graphical representation of the hierarchy of depth two.

and, therefore, provide little new information or guidance to the planner at level n. Actions $a \in A_{n-1}$ can improve the precision of the state and, thus, enhance the performance on the task.

At runtime, the hierarchy decides how actions at each level impact the overall 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 to consolidate belief bottom-up is a reasonable strategy (and will likely result in conservative problem-solving behavior). In this work, instead of computing discrete actions, to compute the reward function at level n we consider receding horizon plans π up to length D consisting of actions and sub-tasks $\in A_2$ that configure lower level reward. We call the set of these plans Π. As such, plan π^* is selected according to

$$\pi^* = \underset{\pi \in \Pi}{\arg\max}\ r_n(b_n, \pi) \tag{2}$$

Plans are evaluated by rolling out the planner to horizon D. However, only the first action $a^* \in \pi^*$ is executed, after which a new plan is computed based on updated beliefs. Actions are chosen at level i until r_i exceeds a threshold specified by the designer.

IV. Demonstrations

Recent work has started to utilize ARcubes [13] to form simple assemblies such as towers [15]. In that work, stochastic actions during assembly were not considered. As a result, occasional failures occurred when objects were not placed precisely in the assembly. Without pro-active management of uncertainty or special purpose recovery mechanisms, such failures require external resets of the system.

We demonstrate the effectiveness of the hierarchical belief space to manage uncertainty at both the object and assembly levels to reduce the need for external resets in three simple assemblies: a simple place with external perturbations, a "tower" of two blocks, and an "pyramid" composed of three blocks. Assemblies are specified by configurations of ARcube features in specific positions. Demonstrations use the uBot-6 mobile manipulator [11].

In the demonstration, a two-layer hierarchy is considered. The bottom layer of the hierarchy is equivalent to the original active belief planner introduced in [13]. This layer of the hierarchy manages noisy interactions with the environment using ATGs as forward models τ_1. B_1 are belief distributions over ARcube ATGs. ARcube ATGs include parameterized mobility and manipulation actions in A_1. The details of these actions are described in [13]. Information gain (with respect to task partitions) is used as reward r_1.

In the top layer of the hierarchy, belief over error in the assembly is considered. Belief B_2 is defined over continuous spatial error. τ_2 and A_2 are provided by a task-planner (such as the output of a symbolic planner) or as a finite state machine by the system designer. A_2 consists of actions that *orient* and *pick-and-place* each object in the assembly. For this continuous state, s_2, belief is updated with

$$b_2(s_{2_{t+1}}) = \eta Pr(z_{2_{t+1}}|s_{2_{t+1}}) \int_{s_{2_t}} Pr(s_{2_{t+1}}|s_{2_t}, a_{2_t}) b_2(s_{2_t}) \tag{3}$$

Observations z_2 are of the errors between constellations of object features with respect to the goal assembly. To form this observation, the maximum likelihood state of B_1 is sampled to "observe" expected feature locations. Missing features are sampled using the forward models of place actions with their respective objects. Given the goal assembly, errors are computed to form the observation z_2. Given z_2, we compute the observation probability $Pr(z_{2_{t+1}}|s_{2_{t+1}})$ with empirical models of robot performance of the place action of the form $\mathcal{N}(\mu, \sigma^2)$. To measure the performance of this layer, the continuous form of Kullback-Leibler divergence D_{KL} between the expected error distribution at the goal, P, and the expected error distribution B_2 given action $a \in A_2$, is considered as the reward r_2. This is defined as

$$r_2 = -D_{KL}(P \parallel B_2) = -\int_{-\infty}^{\infty} p(s_2) log(\frac{p(s_2)}{b_2(s_2)}) ds_2. \tag{4}$$

V. Results

Snapshots of the demonstration with external perturbations are shown in Figure 2. The evolution of the task-level reward r_2 and lower level belief b_1 in terms of actions selected during that demonstration is shown in Figure 3. After the robot places the first object in the assembly, a researcher disturbs the intermediate state by flipping the placed cube (Figures 2a-2c). 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 lower level planner selects lifting, flipping, and orbit actions (Figure 2d-2e) to condense belief back on the target. The robot then picks and places the object (Figure 2f) and observes the outcome (Figure 2g) and determines it has completed the task. The completed assemblies for the "pyramid" and "tower" tasks are shown in Figure 4.