Task-Directed Exploration in Continuous POMDPs for Robotic Manipulation of Articulated Objects

Aidan Curtis[1], Leslie Kaelbling[1], Siddarth Jain[2]

***Abstract*— Representing and reasoning about uncertainty is crucial for autonomous agents acting in partially observable environments with noisy sensors. Partially observable Markov decision processes (POMDPs) serve as a general framework for representing problems in which uncertainty is an important factor. Online sample-based POMDP methods have emerged as efficient approaches to solving large POMDPs and have been shown to extend to continuous domains. However, these solutions struggle to find long-horizon plans in problems with significant uncertainty. Exploration heuristics can help guide planning, but many real-world settings contain significant task-irrelevant uncertainty that might distract from the task objective. In this paper, we propose STRUG, an online POMDP solver capable of handling domains that require long-horizon planning with significant task-relevant and task-irrelevant uncertainty. We demonstrate our solution on several temporally extended versions of toy POMDP problems as well as robotic manipulation of articulated objects using a neural perception frontend to construct a distribution of possible models. Our results show that STRUG outperforms the current sample-based online POMDP solvers on several tasks.**

I. Introduction

Typical model-based approaches to robotics make decisions based on the most likely state of the world. While a point estimate of the world state is tolerable for some applications, it is not sufficient for problems that require actions to improve the model estimate, or in cases where optimal plans under incorrect model assumptions have adverse and irreversible effects. For instance, the robot in Figure 1 has a head-mounted camera that can noisily estimate the state of a cabinet object. The robot also has access to controllers (e.g., `OpenDrawer`) that it can use to affect the world. Executing one of those controllers under an incorrect world model could lead to unintended effects, such as pulling the cabinet off the table or breaking something. An optimal strategy might be to partially execute a controller, observe the effect, and then execute the correct controller under the new estimated model. Finding such an optimal plan requires a representation of model uncertainty and a decision-making strategy that reasons about uncertainty.

Decision-making under uncertainty is a widely studied topic that spans many research areas [1] including motion planning [2], control theory [3], reinforcement learning [4], and task and motion planning [5], [6]. The most general formulation of the problem is as a partially observable Markov decision process (POMDP). Unfortunately, finding optimal solutions to POMDPs can be computationally intractable. Online POMDP solvers have an advantage over optimal solvers because they consider only sections of the belief space reachable from the current state, which enables them to be applied to larger state spaces. Online POMDP solvers such as POMCP [7] and DESPOT [8] that use a particle-based representation of uncertainty have proven to be the most efficient in large state spaces and have even been applied to problems with continuous observations and actions [9].

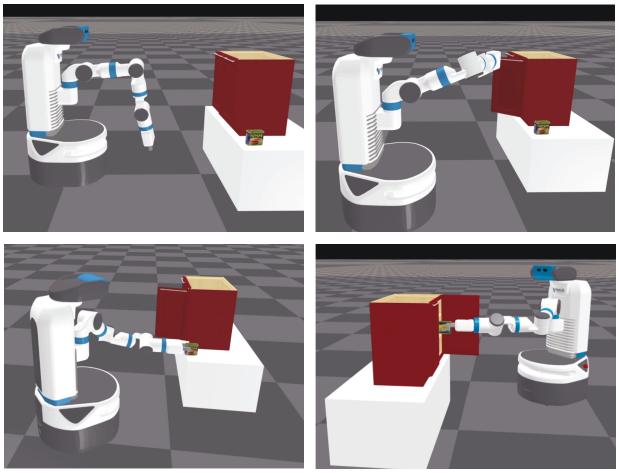

Fig. 1. Visualization of an articulated object robotic manipulation task (OpenDoor) with snapshots of the robot looking at the articulated object (top left), opening the cabinet door (top right), picking up the target object (bottom left), and placing the object inside the cabinet (bottom right).

Although these approaches are capable of working in large and continuous state spaces, they struggle with long-horizon planning when information-gathering is necessary for reducing model uncertainty due to the doubly exponential nature of the search tree. These issues can sometimes be overcome by augmenting the reward or regularizing the value function to explicitly incentivize uncertainty reduction [10]. Other approaches propose active learning, exclusively for uncertainty reduction, followed by maximum likelihood model-based planning [11], which works when all of the uncertainty about the model is relevant to the task. However, for many real-world tasks, there is a significant amount of uncertainty about components of the state that are irrelevant to the task. A robot that tries to reduce all uncertainty in the task shown in Figure 1 will interact with all of the drawers to learn each joint parameter, even when only one drawer is necessary.

In this paper, we propose Search with Task Relevant Uncertainty Guidance (STRUG), a particle-based online POMDP solver capable of finding long-horizon paths to

[1]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. This research was completed during A. Curtis's internship at MERL.

[2]Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. sjain@merl.com

the goal with significant task-relevant and task-irrelevant uncertainty. STRUG uses a domain-independent strategy for identifying task-relevant information-gathering actions. At a high level, STRUG relies on a fast heuristic-based forward search planner to find per-particle plans and then uses these plans to identify subgoals that reduce model uncertainty.

Our work presents two key contributions. First, we develop a novel metric for task-relevant uncertainty and show usefulness in the context of an online belief tree policy search. Second, we demonstrate the effectiveness of our approach by showing how robotic manipulation tasks with complex kinematics can be modeled as a POMDP with information gathering actions and solved using task-relevant uncertainty.

II. Related Work

The topics of planning under uncertainty and estimating articulated models for robot manipulation are widely studied.

Online POMDP solvers: Online POMDP solvers have been shown to be effective in high-dimensional state spaces in comparison to other POMDP solvers [7], [8], [12]. Recent extensions have shown that these approaches also work in continuous action and observation spaces [9] with several applications to robotics [13], [14].

Planning with uncertainty: One approach to planning with uncertainty is to model the uncertainty on various state variables using parameterized continuous distributions and define how certain actions will affect various statistics on those distributions [15]. While this approach works well with structured uncertainty and known action effects, it fails in the presence of unstructured uncertainty. Another way of dealing with uncertainty is to first reduce any model ambiguity through active learning and then plan in the resulting model [16], [17]. Other approaches try to merge the model learning and planning steps using entropy regularized objective functions or reward augmentation to encourage information-gathering [10], [11]. Although active learning and entropy-regularized approaches to planning with uncertainty can effectively reduce general uncertainty, they are overwhelmed when there is significant goal-irrelevant uncertainty.

Kinematic Model Learning: Many approaches from computer vision estimate articulated object models from a sequence of observations [18], [19]. While these methods produce impressive results, it is unclear where these image sequences would come from in autonomous robotic systems. Other approaches have attempted to estimate kinematic models from single images [20], [21], [22], [23]. These approaches are more applicable but are error-prone due to partial visibility and the ambiguity inherent in still images of dynamic objects. Some works have experimented with model-free learning for interaction with articulated objects [24], [25], but these approaches tend to be less efficient in real-world interactions and can be potentially dangerous in safety-critical situations. The most closely related works to ours model kinematic uncertainty and take actions to reduce that uncertainty [26], [16], [27]. To our knowledge, we are the first to attempt to reduce *task-relevant* uncertainty for estimating kinematic models in the context of a task.

III. Background

In this section, we outline the POMDP formulation, describe one class of algorithms used in large and continuous state spaces, and show an extension to the classic formulation that allows for belief-dependent search guidance.

A. POMDPs

A standard infinite-horizon POMDP is defined by the tuple $\langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{Z}, \gamma \rangle$ where $\mathcal{S}, \mathcal{O}, \mathcal{A}$ are the state, observation, and actions spaces respectively, $\mathcal{T}$ defines the transition probabilities that govern state transitions $P(s_{t+1}|s, a)$, $\mathcal{R}$ defines the reward function $\mathcal{R}(s, a)$, $\mathcal{Z}$ defines the observation function $P(o_{t+1}|s_t, a_t)$, and γ is the discount factor.

While this is the standard formulation used for offline solvers, online Monte-Carlo simulation methods use a generative model POMDP definition $\langle G, \mathcal{R}, b_0, \gamma \rangle$ where G is a generative function $G(s_t, a_t)$ that yields (s_{t+1}, o_{t+1}) drawn from $P(s_{t+1}, o_{t+1}|s_t, a_t)$, and b_0 defines the initial state distribution $P(s_0)$. The initial belief b_0 exists in a belief space $\mathcal{B}$, which is the space of probability distributions over the state space. This generative formulation lifts the restrictive assumption that the $\mathcal{T}$ can be represented in closed form. A solution to a POMDP is a policy $\pi : \mathcal{B} \to \mathcal{A}$ that maps beliefs in $\mathcal{B}$ to actions in $\mathcal{A}$ such that the value of the initial belief is maximized, with value defined as follows

$$V_\pi(b) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(b_t))\Big] \,. \tag{1}$$

B. Belief Tree Policy Search

One class of approaches attempts to maximize $V_\pi(b_0)$ by building a belief tree with b_0 as the root node. Nodes in this belief tree alternate between belief nodes and observation nodes. Multiple actions can be taken from a belief node and multiple observations branch from a single observation node due to the stochasticity of the model and uncertainty in the underlying state. At leaf nodes of the belief tree, $V_\pi(b)$ is estimated by simulating action sequences using a random or heuristic-guided policy. At intermediate nodes of the belief tree, $V_\pi(b)$ is estimated from its children by taking an argmax over possible actions and an expectation over resulting observations. The value of an optimal policy π^* at belief b is then defined as follows

$$V_{\pi^*}(b) = \max_{a \in \mathcal{A}} \big(\mathbb{E}_b[R(s, a)] + \gamma \sum_{o \in \mathcal{O}} P(o|b, a) V_{\pi^*}(b')\big) \,. \tag{2}$$

Where b' is the result of performing a belief update using action a, observation o, and prior belief b. In settings with particle belief representations in which a belief is sets of possible states, the belief update is simply $b' = \{s_{t+1} \mid s_{t+1}, o_t = G(s_t, a_t), o = o_t\}$. Sample-based POMDP solvers typically build this belief tree incrementally in an approach similar to MCTS [7], [9], [8]. Given a belief tree, a policy tree can be extracted by selecting the action that maximizes value at each belief node, leaving a tree that only branches on observations.

C. Belief-Dependent Rewards

An important variant of this problem, and the one that we use in this paper, formulates the reward to be a function of the belief $\mathcal{R}(s, a, b)$ where b is computed from a history of action, observation pairs [28]. This formulation has been used to augment the reward to encourage information-gathering actions that reduce entropy in the agent's belief [11], [29].

Active learning approaches directly optimize for reduction in belief state entropy $\mathbb{E}[log(b_s)]$. Other task-directed POMDP solvers augment the reward function with the belief state entropy to encourage actions that lead to low uncertainty along a trajectory [10], [30], [11]:

$$\hat{\mathcal{R}}(s, a, b) = \mathcal{R}(s, a) + \beta \mathbb{E}[log(b)] \ , \tag{3}$$

where β is a hyperparameter that determines the exploration exploitation tradeoff. Our approach formulates a new augmented belief-dependent reward function that includes a measure of task-relevant uncertainty.

IV. Method

This section describes our method for solving POMDPs with belief-dependent reward functions. We call our approach STRUG for *search with task-relevant uncertainty guidance*. At a high level, STRUG samples particles, or possible states, from the initial belief and uses an uncertainty-free heuristic planner to find maximum-reward plans for each particle. The particle-specific plans are then evaluated on the other particles sampled from the initial state distribution yielding a probability of success for each particle's plan on each other particle. We then run a search in belief space with an augmented reward function that incentivizes actions leading to observations that separate particles with mutually incompatible plans. Our key insight is that dissimilar states in the state space often share successful plans, especially when there is a significant degree of task-irrelevant uncertainty.

A. Task-Relevant Uncertainty

We first define task-relevant uncertainty (TRU) of a policy to be the expected variance in value under trajectories sampled from the policy and transition model. We want to incentivize our search to find a policy π with low task-relevant uncertainty (TRU) defined

$$\text{TRU}(b) = \mathbb{E}_{s_1 \sim b}\Big[\text{Var}_{s_2 \sim b}[V^*_{\pi_{s_1}}(s_2)]\Big] \ . \tag{4}$$

This is notably different from energy-based or distance-based objectives that express uncertainty in state. A reasonable approach for estimating this objective during planning is to sample a number of candidate action sequences with a random policy, evaluate the value for each simulated action sequence, and find the sample variance.

However, random actions are unlikely to obtain any reward in sparse-reward environments, resulting in a TRU estimate of zero even when TRU is high. A more accurate TRU estimate in sparse environments requires a denser sampling in the support of the reward function. To obtain this denser sampling, we can identify a set of action sequences with high likelihood of reaching the goal for some particular particle using an uncertainty-free planner. Subsequent sections describe how this metric is calculated and used in the context of an online sample-based POMDP solver for both discrete and continuous action and observation spaces.

B. STRUG Algorithm

We start by sampling a set of M particles $X = \{x_0^0, ..., x_0^p\}$ from the initial belief state b_0. A determinized search is then performed to obtain a maximum-reward plan $\bar{a}^i = ((a_0^i, x_0^i), ..., (a_T^i, x_T^i))$ on each particle $x_0^i \in X$. We then estimate the expected value of each plan on each sampled particle $x_j \in X$ by executing plan $\bar{a}^i$ K times from state x_j, inducing K state trajectory rollouts $(x_0^{i,j}, ..., x_T^{i,j})_k$. This results in an $M \times M$ *plan compatibility matrix*

$$S_b^{i,j} = \frac{1}{K} \sum_{k=0}^{K} \sum_{t=0}^{T} \gamma^t \mathcal{R}(x_t^{i,j}) \ . \tag{5}$$

See Algorithm 1 for details. Element (i, j) in matrix S_b indicates the expected value of executing particle i's optimal plan in particle j. It follows that TRU is calculated from S_b in the following way:

$$\text{TRU}_{S_b}(b) = \sum_{j=0}^{M} b(x_0^j) \sum_{i=0}^{M} \Big(S_b^{i,j} - \sum_{j'=0}^{M} b(x_0^{j'}) S_b^{i,j'} \Big)^2 \ . \tag{6}$$

In order to apply our TRU estimate to an online tree-based POMDP solver, we need a way of calculating the *incremental* TRU, or the TRU gain from taking a certain action in a certain belief state. Incremental TRU is defined recursively as a function of the current belief state and action:

$$\Delta\text{TRU}(b, a) = \mathbb{E}_{o \in \mathcal{O}}\Big[\text{TRU}_{S_{b'}}(b') - \text{TRU}_{S_b}(b)\Big] \ . \tag{7}$$

Unfortunately, calculating TRU at every belief node in the belief tree is computationally burdensome, with the primary bottleneck being per-particle planning time and quadratic plan evaluation time. To resolve this, we make an approximation of ΔTRU that uses a cached version of the plan compatibility matrix S_{b_0} instead of recomputing it at each belief node during search. Approximate ΔTRU is defined as

$$\widehat{\Delta\text{TRU}}(b, a) = \mathbb{E}_{o \in \mathcal{O}}\Big[\text{TRU}_{S_{b_0}}(b') - \text{TRU}_{S_{b_0}}(b)\Big] \ . \tag{8}$$

Intuitively, we want to take actions that result in information that would be useful at the initial belief state. This approximation makes two important assumptions. First is that the uncertainty in the problem can be represented by a possibly infinite set of latent parameters that do not change throughout the problem [11], [31]. The second is that the actions that gather information pertaining to the unchanging hidden variables do not lead to irreversible effects from which the goal cannot be achieved. For example, actions such as dropping a vase on the floor to see if it is made of glass would violate this assumption. The original reward function is augmented with the exploration reward to yield the final belief-dependent reward:

$$\hat{\mathcal{R}}(s_t, a_t, b_t) = \mathcal{R}(s_t, a_t) + \beta \ \widehat{\Delta\text{TRU}}(b_t, a_t) \ , \tag{9}$$

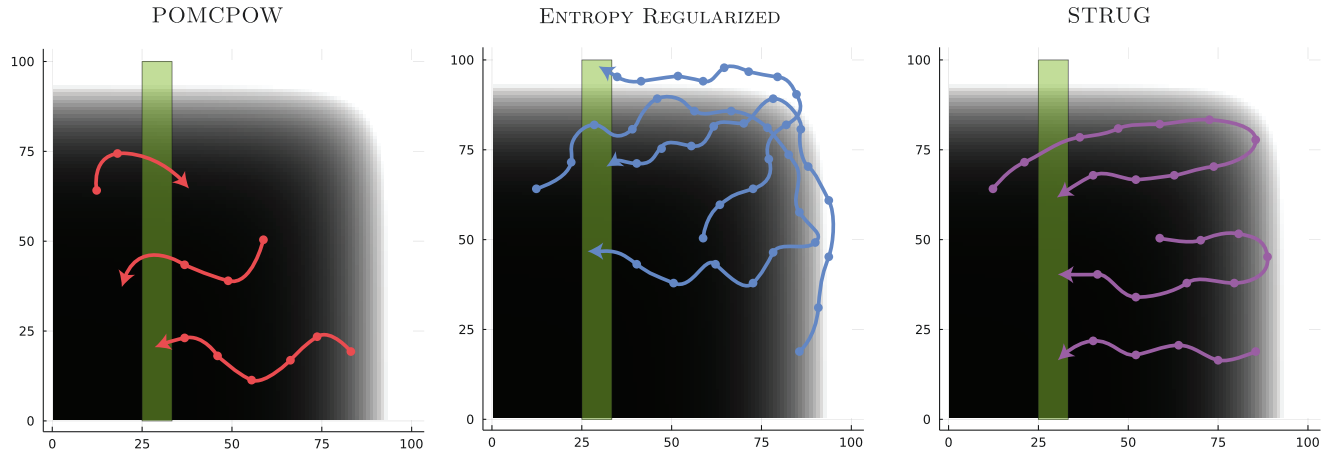

Fig. 2. Visualization of sample trajectories from three POMDP solvers on the LightDark2D Task. The agent starts with uncertainty in its position, and has to navigate to the goal region (vertical green bar). The top wall reduces uncertainty about the y position, and the right wall reduces uncertainty about the x position. POMCPOW fails to take any information-gathering actions, ENTROPY REGULARIZED reduces all state uncertainty before going to the goal. STRUG reduces only task-relevant uncertainty before reaching the goal.

where β is a hyperparameter specifying the importance of gathering task-relevant information. The full algorithm description including TRU augmentation can be seen in 2 In our experiments, we err on the side of selecting large importance values ($\beta = 10$) because the TRU term will approach zero after all task-relevant information is gathered.

Algorithm 1 COMPATIBILITYMATRIX

Require: POMDP model p

$P \leftarrow$ Sample M particles $b \sim b_0$
$F \leftarrow \{\}, D \leftarrow \{\}, S \leftarrow \text{zeros}(M, M)$
for $m \in 1 : M$ **do**
  $D \leftarrow D \cup \text{PLAN}(P[m])$
for $m \in 1 : M$ **do**
  for $d_i \in 1 : |D|$ **do**
    for $k \in 1 : K$ **do**
      $R \leftarrow 0$
      for $a \in \text{reverse}(D[d_i])$ **do**
        $s', o, r \leftarrow G(s, a)$
        $R \leftarrow \gamma R + r$
      $S[m, d_i] \leftarrow S[m, d_i] + R/K$
return S

Algorithm 2 STRUG

Require: POMDP model p, Initial belief distribution b_0

procedure SOLVE(p, b_0)
  $S \leftarrow$ COMPATIBILITYMATRIX(p)
  $t \leftarrow$ EMPTYTREE
  for $i \in$ tree_samples **do**
    SIMULATE(t, p, b_0)
  return $\operatorname{argmax}_a t.Q(b_0, a)$
procedure SIM(t, p, b)
  $s \leftarrow \text{sample}(b)$
  $a \leftarrow \text{HPW}(s, p.A, t.N)$
  $b', o, r \leftarrow G(b, a)$
  $s' \leftarrow \text{sample}(b')$
  if $t.N(b) = 1$ **then**
    $r \leftarrow r + \gamma \text{ROLLOUT}(s') + \beta \Delta \widehat{TRU}(b', a)$
  else
    $r \leftarrow r + \gamma \text{SIM}(t, p, b') + \beta \Delta \widehat{TRU}(b', a)$
  $t.Q(s, a) \leftarrow t.Q(s, a) + (r - t.Q(s', a))/t.N(s', a)$
  return r

C. Hierarchical Progressive Widening

Sampling continuous actions during tree search is typically performed using Progressive widening (PW), double progressive widening (DPW), or Voronoi progressive widening (VPW) [32], [9], [33]. These all work by limiting the number of child nodes of a certain parent node. For example, progressive widening limits the number of child action nodes to kN^α where N is the number of samples reaching the parent node, and k, α are hyperparameters of the search process. This strategy does not easily extend from fixed-dimensional purely continuous action spaces to the hybrid discrete-continuous actions spaces that are typical in robotic planning applications. Typical robotic planning applications define a number of controllers $a \in \mathcal{A}$, each with a unique set of continuous parameters θ_a [34]. A naive extension of double progressive widening might use a fixed-size continuous search space with dimensionality $\max_{a \in \mathcal{A}} |\theta_a|$, ignoring the extraneous continuous parameters for controllers with fewer parameters. Unfortunately, this approach is highly inefficient due to redundancy. We propose a new Hierarchical Progressive Widening (HPW) strategy that splits action selection into a sequence of decisions about discrete action parameters followed by a joint selection of the continuous parameters with progressive widening constraints. HPW uses the standard UCB criterion at each step of the action selection process, giving priority to exploration of the continuous parameters of promising controllers without unnecessary redundancy [35]:

Algorithm 3 HPW

Require: State s, action schemas A, state-action counts N

$a \leftarrow []$
for $\Theta \in A$.discrete **do**
  $a \leftarrow a \oplus \underset{a_i \in \Theta}{\text{argmax}}\ \text{UCB}(s, a_i)$
$a_d \leftarrow a$
for $\Theta \in A$.continuous **do**
  if $|\Theta_s[a_d]| \leq kN(s,a)^\alpha$ **then**
    $\Theta_s[a_d] \leftarrow \Theta_s[a_d] \cup \text{sample}(\Theta)$
  $a \leftarrow a \oplus \underset{a_i \in \Theta_s[a_d]}{\text{argmax}}\ \text{UCB}(s, a_i)$
return a

$$\text{UCB}(s,a) = Q(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}\,. \tag{10}$$

$Q(s,a)$ is the estimated value of an action in a particular state, N(s) is the state sample count, $N(s,a)$ is the state-action sample counts, and c is the exploration parameter. In our experiments we set c to be the maximum reward. See Algorithm 3 for details.

V. Experiments

Our baselines include planning methods that handle uncertainty and aim to trade off information-gathering with reward exploitation. Namely, we evaluate **POMCP** [7] and its continuous variant **POMCPOW** [9]. To apply POMCP to continuous problem domains, we followed the procedure described in [9] to discretize the action and observation spaces. The **Entropy Regularized** uses the augmented reward in Equation 9 to explicitly encourage entropy reduction. Lastly, **Active Learning** separates the problem into two steps: an active learning step that explicitly aims to minimize model entropy and an exploitation step that uses MCTS to find an optimal plan under the most likely model.

We perform experiments on several temporally extended versions of toy POMDP problems and on tasks involving robotic manipulation of articulated objects that relies on perception with a neural network for object-detection to construct a distribution of possible model states. We evaluate on the following tasks that all require some form of information-gathering in addition to goal-directed actions to achieve good performance. For tasks involving robotic manipulation, we use the Fetch robot [36] in the Issac gym simulation [37].

- **Tiger [38]:** The agent stands in front of two doors. Opening one door reveals a tiger (negative reward), and the other reveals treasure (positive reward). The waiting action incurs a slight negative reward but results in a noisy observation of the tiger's location.
- **ExtendedTiger:** A modified version of the Tiger problem that requires several repeated wait actions before any observations are received.
- **LightDark1D/2D [39]:** The agent is uncertain about its current position, and has to navigate to a goal region. It can gain information about its position by moving toward the light region(s). In the 2D environment, moving to particular walls reduce uncertainty about specific dimensions of the agent's location (see Fig. 2).
- **Shelf Manipulation:** The robot is placed in front of a piece of storage furniture (shelf) from the Partnet Mobility dataset [40] with both hinge and door joints and a spam object is placed on a table from the YCB object set. A reward is received when the object is placed inside the storage furniture. Although the agent is uncertain about many aspects of the shelf model, it does not need to reduce uncertainty to achieve the goal.
- **Open Drawer (Door):** The same setup as the Shelf problem, but the articulated object has changed to only have prismatic (hinge) joints. The agent is uncertain about many aspects but only needs to reduce uncertainty about a single drawer (door) to achieve the goal.

A. Robotics Task Details

1) Dynamics & Reward: Our simulated experiments were performed in the Pybullet [41] physics simulator and visualized in IsaacGym [37]. The physics simulator served as the dynamics function f of our POMDP. As shown in Figure 1, we use a mobile-base Fetch robot model [36] with articulated object models from the Sapien Partnet Mobility dataset [40]. Initial state distributions are generated by sampling multiple camera perspectives with the robot RGBD camera and passing the images through a MaskRCNN object detection model that predicts articulated object link masks, joint types, and joint parameters [20]. If the base of the kinematic model moved more than 0.1 meters, the environment terminated, and a reward of -10 was returned. If the task was completed successfully the environment was terminated, and a reward of 10 was given. A maximum of 10 steps was allowed.

2) Initial State Distribution: The MaskRCNN was fine-tuned from the Detectron2 model [42] using simulated data captured in IsaacGym on objects from the Partnet Mobility dataset not used during evaluation. Because the trained model had a tendency to be confidently incorrect, we additionally added noise to the input images, class predictions, and output joint parameters. Example bounding boxes, and predicted masks can be seen in Figure 3 (top).

3) Observation Function: The observation function is defined using the Hausdorff distance, H between the expected and received pointcloud observation:

$$\mathcal{Z}(o|s,a) = \mathcal{N}\big(\text{H}(o, \text{PCD}(f(s,a)), \sigma^2\big)\,, \tag{11}$$

where PCD is the simulated observation that results from execution of action a on state s. We used $\sigma = 10^{-3}$.

4) Action Space: The actions consisted of parameterized controllers `OpenDrawer(?g)`, `OpenDoor(?g)`, `Pick(?o)`,`Place(?o, ?p)`,`Push(?g)`,`PullV(?g)`, `PullH(?g)` where `?o` is the discrete object to manipulate, `?p` is a 6 DoF object pose, and `?g` is a 6 DoF gripper pose. To increase the potential for the pull controllers to impact the kinematic model state, we biased the controller parameter sampling using the Where2Act model [25]. The Where2Act

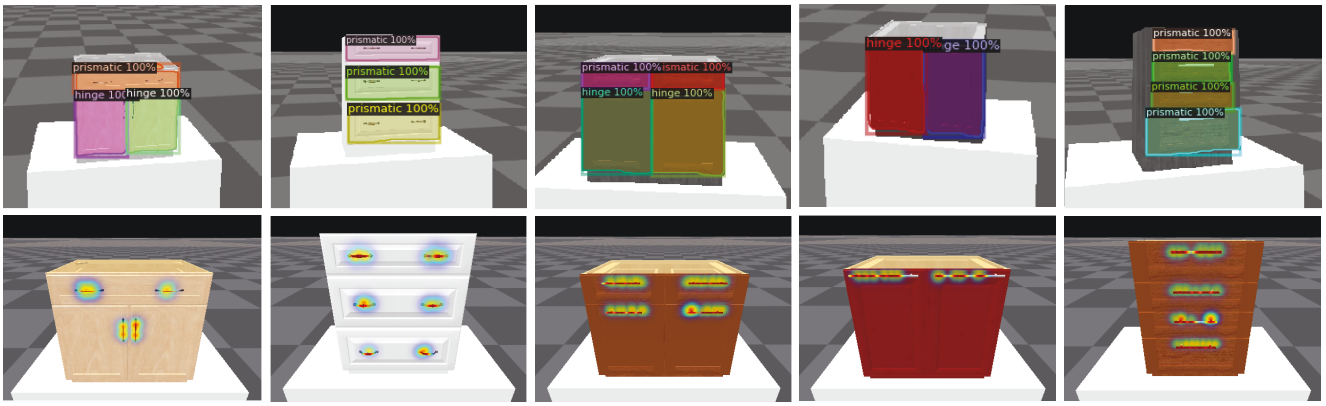

Fig. 3. A visualization of the perceptual model results used to construct the POMDP problem. Top: Results of the MaskRCNN detection module for link mask prediction. Bottom: The combined, thresholded, and filtered Where2act heatmap for the Pull controller

TABLE I

EXPERIMENTAL RESULTS ON THE TASKS DESCRIBED IN SECTION V. THE MEAN DISCOUNTED REWARD AND STANDARD ERROR ACROSS 100 SEEDS. REWARDS ARE BETWEEN -10 AND 10. SCORES WITHIN 2 POINTS OF THE HIGHEST-SCORING ALGORITHM ARE BOLDED

	Tiger	ExtendedTiger	LightDark 1D	LightDark 2D	Shelf	Open Drawer	Open Door
Random	-1.62 ± 0.98	-0.21 ± 0.89	-5.43 ± 0.65	-3.13 ± 0.14	-9.34 ± 0.20	-9.47 ± 0.05	-9.54 ± 0.04
POMCP (Disc)	$\mathbf{5.67 \pm 0.75}$	-0.02 ± 0.98	1.30 ± 0.98	-0.57 ± 0.12	$\mathbf{9.02 \pm 0.00}$	-1.00 ± 0.33	-1.50 ± 0.41
POMCPOW	$\mathbf{6.49 \pm 0.66}$	-0.37 ± 0.81	2.60 ± 0.97	-1.20 ± 0.89	$\mathbf{8.62 \pm 0.19}$	1.30 ± 0.71	0.20 ± 0.71
Entropy Regularized	$\mathbf{6.25 \pm 0.31}$	$\mathbf{3.95 \pm 0.34}$	0.12 ± 0.94	1.26 ± 0.41	3.91 ± 0.11	0.34 ± 1.18	-0.15 ± 0.97
Active Learning	$\mathbf{5.59 \pm 0.28}$	$\mathbf{3.96 \pm 0.33}$	$\mathbf{5.50 \pm 0.82}$	$\mathbf{2.13 \pm 1.29}$	1.62 ± 0.92	1.49 ± 1.91	1.03 ± 0.84
STRUG	$\mathbf{6.34 \pm 0.31}$	$\mathbf{2.31 \pm 0.25}$	$\mathbf{5.80 \pm 0.62}$	$\mathbf{3.55 \pm 0.91}$	$\mathbf{9.02 \pm 0.00}$	$\mathbf{4.22 \pm 1.60}$	$\mathbf{5.21 \pm 1.39}$

models generate heatmaps of pull locations for horizontal and vertical orientations that highlight handles, buttons, and other likely object interfaces on unseen articulated objects. The Where2Act models are also trained from a simulated dataset using Partnet Mobility models. An example combined horizontal and vertical pull heatmap is shown in Fig. 3 (bottom).

B. Results & Discussion

Our results show all POMDP methods work on the simplest Tiger problem with relatively equal performance. In the more difficult ExtendedTiger problem, we observe that only approaches that perform explicit uncertainty reduction (Entropy Regularized, Active Learning, and STRUG) perform well. A similar trend is apparent in the LightDark 1D and LightDark 2D domains. We see the Entropy Regularized baseline struggle in the 1D version of this domain, and both Entropy Reduction and Active Learning perform worse in the 2D version. A few qualitative results shown in Figure 2 help elucidate the problem. Methods that reduce general uncertainty perform better than standard POMDP solvers but are less efficient than STRUG, which reduces only task-relevant uncertainty. For the shelf robotics task, we can see that the standard POMDP methods solve it effortlessly since no additional information is needed to solve the goal. However, the methods that reduce general uncertainty are less efficient because they try to reduce uncertainty anyway. Lastly, in the Open Drawer and Open Door robotics tasks, we observe that only STRUG is capable of solving them. Standard POMDP methods fail to consider information-gathering actions due to the temporally extended nature of the problem, and general information-gathering approaches spend more time gathering unnecessary information.

Although our task selection does not highlight them, STRUG does come with a set of assumptions and general limitations. First, it cannot be used in environments with irreversible actions that gain task-relevant information. Second, using an approximation of TRU restricts STRUG to environments in which the uncertain variables do not change during the task. In future work, we hope to address some of these shortcomings and move to larger task and motion planning robotics domains with more controllers, objects, and temporal dependencies.

VI. CONCLUSIONS

In this paper, we present STRUG, an approach for planning with model uncertainty that prioritizes information-gathering actions the reduce task-relevant uncertainty. While directly reducing task-relevant uncertainty is intractable, we found a practical approximation under some assumptions. We evaluated this approach on a number of POMDP problems, including robotic manipulation of articulated objects, a task that contained model uncertainty arising naturally from limitations in the robot's perception. Our findings show that augmenting reward with task-relevant uncertainty improves performance over existing POMDP solvers in temporally extended domains and outperforms approaches that reduce general uncertainty instead of task-relevant uncertainty.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[2] N. Dadkhah and B. Mettler, "Survey of motion planning literature in the presence of uncertainty: Considerations for UAV guidance," *J. Intell. Robotics Syst.*, vol. 65, no. 1–4, p. 233–246, jan 2012. [Online]. Available: https://doi.org/10.1007/s10846-011-9642-9

[3] H. La, A. Potschka, J. Schlöder, and H. Bock, "Dual control and information gain in controlling uncertain processes," *IFAC-PapersOnLine*, vol. 49, no. 7, pp. 139–144, 2016, 11th IFAC Symposium on Dynamics and Control of Process SystemsIncluding Biosystems DYCOPS-CAB 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896316304372

[4] M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar, "Bayesian reinforcement learning: A survey," *CoRR*, vol. abs/1609.04436, 2016. [Online]. Available: http://arxiv.org/abs/1609.04436

[5] L. Kaelbling and T. Lozano-Perez, "Integrated task and motion planning in belief space," *The International Journal of Robotics Research*, vol. 32, pp. 1194–1227, 08 2013.

[6] C. R. Garrett, C. Paxton, T. Lozano-Pérez, L. P. Kaelbling, and D. Fox, "Online replanning in belief space for partially observable task and motion problems," *CoRR*, vol. abs/1911.04577, 2019. [Online]. Available: http://arxiv.org/abs/1911.04577

[7] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23. Curran Associates, Inc., 2010.

[8] N. Ye, A. Somani, D. Hsu, and W. S. Lee, "DESPOT: online POMDP planning with regularization," *CoRR*, vol. abs/1609.03250, 2016. [Online]. Available: http://arxiv.org/abs/1609.03250

[9] Z. Sunberg and M. J. Kochenderfer, "POMCPOW: an online algorithm for pomdps with continuous state, action, and observation spaces," *CoRR*, vol. abs/1709.06196, 2017. [Online]. Available: http://arxiv.org/abs/1709.06196

[10] T. L. Molloy and G. N. Nair, "Entropy-regularized partially observed markov decision processes," *ArXiv*, vol. abs/2112.12255, 2021.

[11] M. Chen, E. Frazzoli, D. Hsu, and W. S. Lee, "POMDP-Lite for robust robot planning under uncertainty," *CoRR*, vol. abs/1602.04875, 2016. [Online]. Available: http://arxiv.org/abs/1602.04875

[12] K. M. Seiler, H. Kurniawati, and S. P. N. Singh, "An online and approximate solver for POMDPs with continuous action space," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2290–2297.

[13] M. C. Koval, N. S. Pollard, and S. S. Srinivasa, "Pre- and post-contact policy decomposition for planar contact manipulation under uncertainty," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 244–264, 2016. [Online]. Available: https://doi.org/10.1177/0278364915594474

[14] M. Chen, E. Frazzoli, D. Hsu, and W. S. Lee, "POMDP-Lite for robust robot planning under uncertainty," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5427–5433, 2016.

[15] L. P. Kaelbling and T. Lozano-Pérez, "Implicit belief-space pre-images for hierarchical planning and execution," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 5455–5462.

[16] N. Houlsby, F. Huszár, Z. Ghahramani, and M. Lengyel, "Bayesian active learning for classification and preference learning," 12 2011.

[17] M. Noseworthy, C. Moses, I. Brand, S. Castro, L. P. Kaelbling, T. Lozano-Pérez, and N. Roy, "Active learning of abstract plan feasibility," *CoRR*, vol. abs/2107.00683, 2021. [Online]. Available: https://arxiv.org/abs/2107.00683

[18] Z. Yan, R. Hu, X. Yan, L. Chen, O. van Kaick, H. Zhang, and H. Huang, "RPM-Net: Recurrent prediction of motion and parts from point cloud," *CoRR*, vol. abs/2006.14865, 2020. [Online]. Available: https://arxiv.org/abs/2006.14865

[19] L. Yi, H. Huang, D. Liu, E. Kalogerakis, H. Su, and L. Guibas, "Deep part induction from articulated object pairs," *ACM Trans. Graph.*, vol. 37, no. 6, dec 2018. [Online]. Available: https://doi.org/10.1145/3272127.3275027

[20] H. Jiang, Y. Mao, M. Savva, and A. X. Chang, "Opd: Single-view 3d openable part detection," 2022. [Online]. Available: https://arxiv.org/abs/2203.16421

[21] B. Abbatematteo, S. Tellex, and G. Konidaris, "Learning to generalize kinematic models to novel objects," in *Proceedings of the Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds., vol. 100. PMLR, 30 Oct–01 Nov 2020, pp. 1289–1299. [Online]. Available: https://proceedings.mlr.press/v100/abbatematteo20a.html

[22] X. Li, H. Wang, L. Yi, L. J. Guibas, A. L. Abbott, and S. Song, "Category-level articulated object pose estimation," *CoRR*, vol. abs/1912.11913, 2019. [Online]. Available: http://arxiv.org/abs/1912.11913

[23] J. Lv, Q. Yu, L. Shao, W. Liu, W. Xu, and C. Lu, "Sagci-system: Towards sample-efficient, generalizable, compositional, and incremental robot learning," 11 2021.

[24] O. Brock, J. Trinkle, and F. Ramos, *Learning to Manipulate Articulated Objects in Unstructured Environments Using a Grounded Relational Representation*, 2009, pp. 254–261.

[25] K. Mo, L. J. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani, "Where2act: From pixels to actions for articulated 3d objects," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 6813–6823.

[26] D. Katz and O. Brock, "Manipulating articulated objects with interactive perception," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 272–277.

[27] M. Noseworthy, C. Moses, I. Brand, S. Castro, L. P. Kaelbling, T. Lozano-Pérez, and N. Roy, "Active learning of abstract plan feasibility," *CoRR*, vol. abs/2107.00683, 2021. [Online]. Available: https://arxiv.org/abs/2107.00683

[28] M. Araya, O. Buffet, V. Thomas, and F. Charpillet, "A POMDP extension with belief-dependent rewards," in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23. Curran Associates, Inc., 2010.

[29] L. Dressel and M. Kochenderfer, "Efficient decision-theoretic target localization," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 27, no. 1, pp. 70–78, Jun. 2017. [Online]. Available: https://ojs.aaai.org/index.php/ICAPS/article/view/13832

[30] T. L. Molloy and G. N. Nair, "Smoother entropy for active state trajectory estimation and obfuscation in POMDPs," *CoRR*, vol. abs/2108.10227, 2021. [Online]. Available: https://arxiv.org/abs/2108.10227

[31] F. Doshi-Velez and G. D. Konidaris, "Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations," *CoRR*, vol. abs/1308.3513, 2013. [Online]. Available: http://arxiv.org/abs/1308.3513

[32] A. Couëtoux, J.-B. Hoock, N. Sokolovska, O. Teytaud, and N. Bonnard, "Continuous upper confidence trees," vol. 6683, 01 2011, pp. 433–445.

[33] M. H. Lim, C. J. Tomlin, and Z. N. Sunberg, "Voronoi progressive widening: Efficient online solvers for continuous space MDPs and POMDPs with provably optimal components," *CoRR*, vol. abs/2012.10140, 2020. [Online]. Available: https://arxiv.org/abs/2012.10140

[34] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *CoRR*, vol. abs/2010.01083, 2020. [Online]. Available: https://arxiv.org/abs/2010.01083

[35] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling, P. Rohlfshagen, S. Tavener, D. Perez Liebana, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4:1, pp. 1–43, 03 2012.

[36] M. Wise, M. Ferguson, D. King, E. Diehr, and D. Dymesich, "Fetch & freight : Standard platforms for service robot applications," 2016.

[37] J. Liang, V. Makoviychuk, A. Handa, N. Chentanez, M. Macklin, and D. Fox, "GPU-accelerated robotic simulation for distributed reinforcement learning," *CoRR*, vol. abs/1810.05762, 2018. [Online]. Available: http://arxiv.org/abs/1810.05762

[38] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1, pp. 99–134, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000437029800023X

[39] R. Platt, L. Kaelbling, T. Lozano-Perez, , and R. Tedrake, "Efficient planning in non-Gaussian belief spaces and its application to robot

grasping," in *Proc. of International Symposium of Robotics Research, (ISRR)*, Flagstaff, AZ, 2011.

[40] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. Chang, L. Guibas, and H. Su, "Sapien: A simulated part-based interactive environment," 06 2020, pp. 11 094–11 104.

[41] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[42] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.