POMDP Planning Under Object Composition Uncertainty: Application to Robotic Manipulation

Joni Pajarinen, Jens Lundell, and Ville Kyrki, *Senior Member, IEEE*

***Abstract*—Manipulating unknown objects in a cluttered environment is difficult because segmentation of the scene into objects, that is, object composition, is uncertain. Due to the uncertainty, prior work has either identified the "best" object composition and decided on manipulation actions accordingly or tried to greedily gather information about the "best" object composition. We instead, first, use different possible object compositions in planning, second, utilize object composition information provided by robot actions, third, consider the effect of competing object hypotheses on the desired task. We cast the manipulation planning problem as a partially observable Markov decision process (POMDP) that plans over possible object composition hypotheses. The POMDP chooses the action that maximizes long-term expected task-specific utility, and while doing so, considers informative actions and the effect of different object hypotheses on succeeding in the task. In simulation and physical robotic experiments, a probabilistic approach outperforms using the most likely object composition, and long term planning outperforms greedy decision making.**

***Index Terms*—Grasping, image segmentation, partially observable Markov decision process (POMDP), robotic manipulation, task planning.**

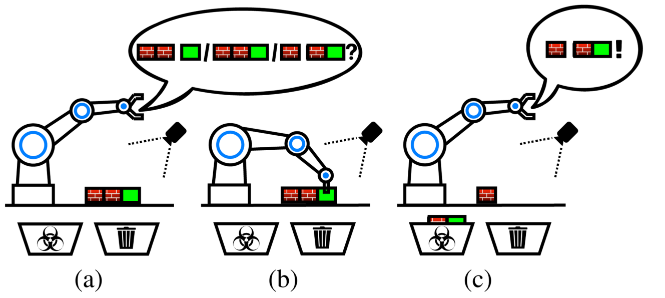


Fig. 1. In the automated waste recycling example, the robot is tasked with finding and putting objects that do not contain toxic green material into the rightmost bin. (a) Due to uncertain segmentation, three different object hypotheses exist. (b) Best option is to pick a block with green color and move it to the leftmost bin. (c) By removing the block with green, the robot resolves the segmentation uncertainty and finds recyclable material.

I. Introduction

SERVICE robots in domestic environments need the ability to manipulate objects without good prior models in order to cope with the variability of such environments. This need is usually approached by modeling the objects online using sensors based on stereopsis or structured light. When multiple measurements can be acquired around an isolated object, this approach works quite satisfactorily as the generated 3-D models can often be used for successful manipulation. On the other hand, in cluttered scenes with multiple unknown objects, the segmentation of objects, also known as object discovery in perception research, becomes a significant problem.

In clutter, the challenge is to decide which segments in an over-segmented scene belong to the same object, as shown in Fig. 1. This challenge is notoriously difficult as objects can partially occlude each other. A promising approach toward solving object discovery in clutter is interactive perception [1], where the object configuration is examined actively, typically by manipulating the objects and observing the results. Another line of work is to use learned priors to find the most likely object composition. Despite the recent advances, manipulation of unknown objects in cluttered environments is still an open problem.

This article proposes another solution to manipulation planning in clutter, where the idea is to plan over hypotheses of possible object compositions (segmentations) instead of determining a single best hypothesis. For estimating the distribution of object compositions, we propose a Markov chain Monte Carlo (MCMC) procedure that produces approximately exact, independent samples from the distribution. The approach combines earlier ideas of interactive perception and learned composition priors in a planning under uncertainty framework. The manipulation planning problem is cast as a partially observable Markov decision process (POMDP), which integrates active exploration and planning. In particular, a POMDP model enables the use of information-gathering actions which may not yield immediate benefit but allows collecting crucial information, for example, about object compositions, that is required to solve the actual

Manuscript received 15 October 2021; revised 6 April 2022; accepted 13 June 2022. Date of publication 20 July 2022; date of current version 8 February 2023. This work was supported in part by the Academy of Finland, decision 271394 and in part by the German Research Foundation (DFG) Project PA 3179/1-1 (ROBOLEAP). This article was recommended for publication by Associate Editor R. Tron and Editor W. Burgard upon evaluation of the reviewers' comments. *(Corresponding author: Joni Pajarinen.)*

Joni Pajarinen is with the Department of Electrical Engineering and Automation, Aalto University, 00076 Aalto, Finland, and also with the Intelligent Autonomous Systems, Technische Universität Darmstadt, 64289 Darmstadt, Germany (e-mail: joni.pajarinen@aalto.fi).

Jens Lundell and Ville Kyrki are with the Department of Electrical Engineering and Automation, Aalto University, 00076 Aalto, Finland (e-mail: jelundel@kth.se; ville.kyrki@aalto.fi).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TRO.2022.3188168.

Digital Object Identifier 10.1109/TRO.2022.3188168

task in the end. That is, the POMDP formulation allows us to utilize the information provided by robot actions and plan actions under object composition uncertainty into the future, which is beneficial for manipulating objects in challenging environments such as the one presented in Fig. 1.

In contrast to earlier work, our approach:

1) utilizes different possible object compositions in decision making;
2) considers the effect of competing hypotheses on the goal task;
3) actively explores the hypothesis space if that benefits the task.

This journal article extends our earlier conference papers [3] and [4] by

1) combining the MCMC procedure for belief estimation in [3] with the POMDP planning in [4];
2) by including entirely hidden objects into the POMDP model by taking advantage of hidden volume information;
3) by a high volume of simulations for statistically significant evaluations;
4) new experiments using a Franka Panda robotic arm verifying the general applicability of the approach.

This article begins by surveying work related to object discovery, interactive perception, and planning under uncertainty in Section II. Our framework of manipulation planning over object compositions is then introduced in Section III. Section IV proposes our approach for estimating the distribution over object hypotheses. The state space of hypotheses is then used for manipulation planning as described in Section V. Experiments with two different physical robot arms and an RGB-D sensor presented in Section VI demonstrate that the proposed approach can integrate perception to the manipulation task and that the use of multiple hypotheses improves the system performance when compared to considering only the most likely hypotheses. Finally, Section VII concludes this article.

II. Related Work

Our work focuses on manipulation under object composition uncertainty. In the following, we discuss related work w.r.t. image segmentation, object discovery, active and interactive perception, grasping unknown objects, and manipulation under uncertainty.

Image segmentation: Image segmentation is a widely and actively studied research field [2], [5]–[8]. Most of the earlier work concentrates on segmenting grey and color 2-D images [5], [6]. However, with the introduction of new cheap 3D-sensors, such as Microsoft's Kinect or Intel's RealSense, research on segmenting 3-D images has gained in popularity [2], [9]–[13]. The 3-D information from these sensors are especially useful for robotics tasks including grasping and manipulation [13]–[17]. The main limitation with the aforementioned methods is that they only consider point estimates of the segmentation, which, in the case of poor segments, can lead to unsuccessful grasps [13].

To mitigate the effect of wrong segmentations, related works have focused on utilizing a probability distribution oversegmentations [18], [19] or considered many segmentation hypotheses [20]. By treating the problem as such it is possible to use robots to gather information about the most likely segmentation through exploratory actions, such as poking [19], [20]. In this work, we also treat the problem probabilistically but instead of considering only distributions oversegmentations, we model the complete probability distribution over object compositions and use this information for downstream decision making tasks.

A common technique for finding the best object composition is through graph cuts [2], [7], [21]. This article, instead, estimates the probability distribution over object compositions using an MCMC procedure. Prior works have also applied MCMC for segmentation, but focused on finding a single best segmentation [22] or a fixed amount of distinct 2-D segmentations [23].

Object discovery: Scene segmentation is a classic problem in computer vision tightly coupled to object recognition [24], [25] so that it can be argued that the segmentation problem does not have unique solutions if the objects are not known. Nevertheless, there is a need to discover objects from scenes even when the objects are *a priori* unknown. Recent works in the area are based on supervised learning for detecting objects from synthetic training data [12], deep-learning for unsupervised object discovery [26]–[29], learning general models for recognizing object classes from segments (e.g., segment labeling [30]), detecting segments based on their "objectness" [31], or choosing which segments belong to a single object [2], [11]. Our work follows the line of work of [2] and [11], but instead of trying to find a single optimal composition, it considers a distribution of possible compositions.

Active and interactive perception: Instead of passively discovering objects, other approaches have addressed the object discovery problem from the point of view of active or interactive perception [32]. Some examples of active perception are Gaze control and foveation [33]. Interactive perception [1] has been proposed for object discovery with the goal of singulating [34], clearing [20], or segmenting [19] a pile of objects. These approaches use poking or pushing actions to estimate the object composition. This article follows the interactive perception paradigm but, in contrast to the works above, integrates the perception with goal-directed planning so that perceptual actions are only used when they are expected to support the task goal.

Grasping unknown objects: In recent decades, grasping unknown objects has gotten significant attention in the research community, especially after Saxena's seminal work [35] on using machine learning for detecting good grasps. Since then, the deep-learning-based grasping methods have revolutionized the area of unknown object grasping [14]–[16], [36]–[39]. However, to reach high grasp success rates, many of these methods [14], [16], [36], [37], [39] have constrained grasps to top-down and 4 degrees of freedom. In this work, we also consider top-down grasps but, instead of using a data-driven methodology, we align the robot hand according to the principal axes of the point-cloud. In addition, we consider grasping as a component for both informative and goal-directed actions; hence, even failed grasps give valuable information about object composition that is used for planning future grasps.

Manipulation planning under uncertainty: In planning manipulation actions under uncertainty, classical deterministic planning can be used to reduce uncertainty. For example, Dogar *et al.* [40] plan pregrasp pushing actions that collapse pose uncertainty of a target before executing a grasping action. This type of approach is usually only available for completely known objects.

When faced with limited knowledge, POMDP-like approaches can be used to plan over a distribution of states. Hsiao *et al.* [41] proposed the partitioning of a one-dimensional configuration space to yield a discrete POMDP, which can be solved for an optimal policy. For planning grasp locations under uncertainty, prior state-of-the-art probabilistic approaches [42]–[44] have focused on positioning the robot as accurately as possible [42] or maximizing the probability of a successful grasp [43]. The planning can also be extended to include information gathering actions [44].

In POMDP-based approaches for multiobject manipulation, Monso *et al.* [45] use a POMDP definition designed specifically for clothes separation, which assumed that each object is uniform in color. In contrast to [45], this article does not make that assumption as it addresses object discovery of multicolored and textured objects. Moreover, the state-space model of [45] is clothes separation specific, which, for instance, focuses on modeling the number of clothes in different areas. Our approach reasons about objects directly.

Other example of multiobject manipulation tasks that have been formulated as POMDPs include emptying a refrigerator [46] and object search [47]. Similarly to our work, Xiao *et al.* [47] also consider fully occluded objects. The main difference between the approach in [46] and ours is that [46] do not explicitly model a probability distribution over object hypotheses but assume *a priori* six different objects. For the approach in [47], the main differences to ours is that it assume perfect object composition segmentation with some uncertainty in object locations and that the number and models of objects are known in advance. Our approach, on the other hand, does not assume any information on object models or number of objects.

III. Manipulating Object Compositions

We consider the scenario of a robot manipulating unknown objects based on RGB-D data. The manipulation goal is defined in terms of simple features that can be observed incompletely from the point clouds. For example, the goal could be to move all objects with a certain color to a particular location. Manipulating unknown objects is difficult because even if RGB-D data is available, the robot does not know in advance the shape or color of the objects. Thus, the robot has to guess, which parts of the point cloud belong to the same object. Occlusion and noisy sensor readings make this task challenging. Attempting to segment individual objects from the point cloud typically results in oversegmentation, which leads to the problem of deciding which segment belongs to which object (object composition), forming *object hypotheses*.

In previous works such as [48], the choice of an action is based on the most probable hypothesis of object composition. The shortcoming of this approach is that it does not account for the long term effects of uncertainty or the value of information gathering actions. Instead, we propose to choose the action that maximizes long term reward over the current and future distribution of possible object compositions. By considering a temporally evolving system, the robot can infer from past grasp attempts the likelihood of object hypotheses.

Fig. 2 shows an overview of the proposed approach. At each time step, we capture an RGB-D image by a vision sensor, such as a Microsoft Kinect and oversegment the RGB-D image. Furthermore, using MCMC and information about previous manipulation outcomes, we generate a probability distribution over possible object compositions where each composition consists of segment patches. We perform long-term planning over possible future object composition distributions using a POMDP model. Finally, the best POMDP action is executed on the robot. Before going into belief estimation and manipulation planning details, we first describe how to model robotic manipulation as a POMDP.

A. Robotic Manipulation as a POMDP

A POMDP defines optimal behavior for an agent in an uncertain world with noisy, partial measurements, when the stochastic world model is accurate and when the agent's goal has been defined precisely. Previously, POMDPs have yielded good results in robotic applications, such as navigation [49], autonomous driving [50], human–robot interaction [51], and manipulation [41], [45]–[47], [52]. We utilize a POMDP because it accounts for uncertainty in action effects and observations. Moreover, a POMDP assigns the correct long-term value to informative actions, which are needed when exploring object hypotheses.

The temporal model of a POMDP is defined by the transition probability $P(s'|s,a)$ from state s to the next time step state s', when action a is executed, and the probability $P(o|s',a)$ of observing o, when action a was executed and the world moved to the state s'. A real-valued reward $R(s,a)$ for executing action a in state s encodes the objective. An optimal policy π maximizes the expected reward $E[\sum_{t=0}^{T-1} R(s(t),a(t))|\pi,b_0]$ over T time steps, where b_0 denotes the initial *belief*, which represents a probability distribution over world states. At each time step, the agent decides on an action a based on its current belief $b(s)$.

In principle, the belief can be kept up-to-date given an accurate temporal model. However, because an accurate model is, in practice, unavailable, we instead estimate the belief at each time step from current visual sensor data and past history, and use an *online* POMDP method to compute a new policy at each time step. That is, we take a receding-horizon online approximate planning approach to action selection. We use the particle policy graph improvement (PPGI) algorithm introduced in [52] (the technical report [53] provides further details) to compute a new policy. To cope with the large state space, PPGI uses a state particle representation for the belief $b(s)$ and to cope

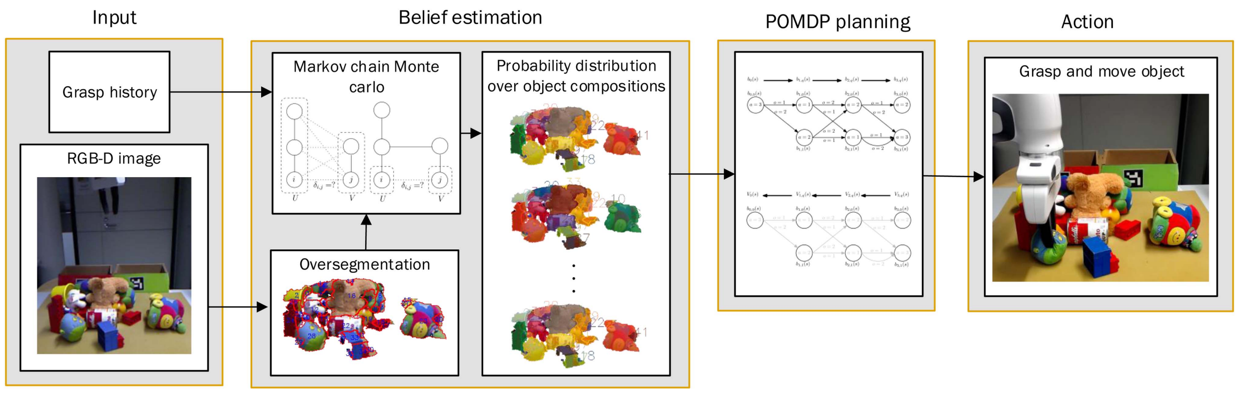


Fig. 2. Overview of the proposed approach. At each time instant the robot obtains and oversegments RGB-D scene data. From pair-wise segment probabilities of belonging to the same object, we generate a probability distribution over possible object compositions. We condition the probability distribution on past grasp successes and failures (see Section IV). The robot uses a POMDP to select the best long-term manipulation action for the current object distribution (see Sections III-A and V) and executes the action. For segmenting RGB-D data and estimating probabilities for segment pairs, we use [2].

with a large action space a graph-based policy representation. Other sample-based POMDP methods, such as POMCP [54] or DESPOT [55], would also be applicable.

The POMDP model used in this article is based on the POMDP model in our previous work on multiobject manipulation under partial observability [52]. In [52], perfect segmentation was assumed, meaning that we knew exactly which parts of the scene belong to which object. Here, our POMDP model uses a probability distribution over possible object hypotheses and grasping actions occur in the space of object hypotheses. Shortly, the scene is oversegmented into a set of segments (point clouds). Possible objects, also known as object hypotheses, are formed from a combination of segments, which is further described in Section IV. Following [52], the probability of successfully grasping a hypothetical object, and observing its attributes (such as color), depend on how occluded the object is. Moreover, we assume that a previously failed grasp cannot succeed unless the occlusion on the grasped object changes.

Formally, the POMDP state $s = (s_1, s_2, \ldots, s_N)$ is a combination of (hypothetical) object states $s_i = (s_i^{\text{loc}}, s_i^{\text{attr}}, s_i^{\text{point cloud}}, s_i^{\text{hist}})$ where s_i^{loc} is the semantic location of the object, such as "on top of the table" or "in a box," s_i^{attr} is the attribute of the object, $s_i^{\text{point cloud}}$ is the point cloud associated with this object (empty for hidden objects), and s_i^{hist} is the historical information for object i. The historical information in s_i^{hist} contain all necessary information for decision making, including the grasp success probability (see Section V-A3), which depends on prior grasp failures and successes, and the historical observation information used for computing observation probabilities (see Section V-B). $s_i^{\text{point cloud}}$ is composed of a set of point clouds that correspond to the image segments of the object hypothesis, which is further discussed in Section IV. Attributes of the object, such as color, shape, and pose, can be derived from $s_i^{\text{point cloud}}$. In practice, $s_i^{\text{point cloud}}$ is stored as a set of pointers to point clouds (segments) to limit storage space. Since each state in a belief consist of object hypotheses with different sets of segments, the belief corresponds to a distribution over different object compositions. The number of objects, N, and object states, s_i, differ between different POMDP states since we model a variety of object hypotheses. This procedure is further detailed in the following section.

Rewards, actions, observations, transition, and observation probabilities. In order to perform POMDP planning, in addition to the state space, we need to define rewards, actions, observations, transition, and observation probabilities. *Rewards*, $R(s, a)$, are application specific. For example, in this work, the agent is rewarded for moving objects into the correct basket. Consequently, we define an *action*, a, as an attempt to grasp and move an object hypothesis. The action specifies a discrete object hypothesis id and a discrete semantic location to move the object hypothesis to, such as a the target basket. For more details on actions and grasping see Section V.

Observations, o, consist of a discrete combination of attributes, such as color, for a limited set of object hypotheses and a boolean indicating whether a grasp succeeded or not (see Section V-B2 for more details). Observations yield limited information about the current state due to (i) occlusions and sensor noise preventing observing objects and object compositions fully, and (ii) for efficient POMDP planning the number of different observations needs to be limited.

State transition probabilities, $P(s'|s, a)$, are based on whether an object hypothesis is grasped and moved successfully or not, which influences the probability of both the grasped and other object hypotheses (see Section V-B1 for more details). Similarly to [52], *observation probabilities* $P(o|s,' a)$ depend on how object (hypotheses) occlude each other: heavy occlusion, for instance, makes observations noisier (see Section V-B3 for more details).

IV. Belief Estimation

The belief consists of state particles and their probabilities. Here, instead of objects, each state, which we refer to as an *object composition*, consists of a set of *object hypotheses*. An object hypothesis consists of a set of segments where each segment

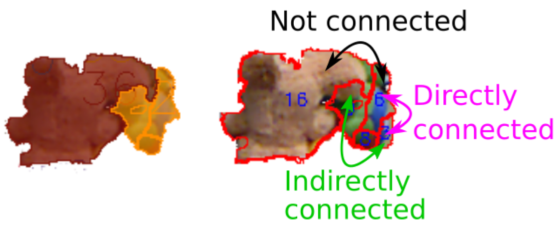


Fig. 3. Illustration of the segment connection types. The left image displays two object hypotheses that are constructed from the oversegmented image on the right. The topmost (black) arrow indicates that if two segments in the oversegmented image do not belong together, they are not connected. On the other hand, the rightmost (pink) arrow indicates that if two segments share a border and belong to the same hypothesis, they are directly connected. Finally, the bottom-most (green) arrow indicates that if the segments belong to the same hypothesis but do not share a border, they are indirectly connected.

consists of a set of points that together form a point cloud. Each point, in turn, is represented by an x, y, z coordinate and a color.

For performing inference on object hypotheses, we need to define how segments relate to each other. We defined three such relationships (all shown in Fig. 3): "directly connected," "in-directly connected," and "not connected". If two segments belong to the same hypothesis they are either "directly connected" or "in-directly connected." A direct connection exist if the segments occlude each other or if another segment occludes both, which would allow for a direct connection behind the occluding object. Segments not part of the same object hypothesis are referred to as "not connected."

The probability of a sampled state is proportional to the probability of the related object composition existing which, in turn, depends on the history of successful and failed grasps. The key insight is that if previously performed grasps on an object hypothesis did not move the object the grasps must have failed. Similarly, a grasp can only succeed for an incorrect hypothesis when that hypothesis is part of a hypothesis for which the grasp succeeds. Due to these insights, we assume that failed grasps do not displace objects. This assumption proved to hold very well in the experiments, even though objects were close to each other.

We always estimate the current belief from the current image observation conditioned on the action/observation history. We use a computer vision algorithm to segment the observed point cloud and compute the connectedness probability for each point cloud segment pair. Based on these probabilities, and whether segments can be directly connected, we define a Markov chain which converges to a distribution over object compositions. We sample object compositions from this Markov chain after a burn-in period. A belief state corresponds to a sampled object composition with sampled object attributes. Our framework also supports sampling object locations for a belief state. However, in the experiments, object locations are deterministic.

The probability of the belief state is set proportional to the probability of the sampled object composition, which is computed (for uniform priors) as the probability of the observation/action history conditional on the object composition. This means that the belief over object compositions is shaped by past events. For example, if the robot fails to grasp an object hypothesis, which should be easy to grasp when the object

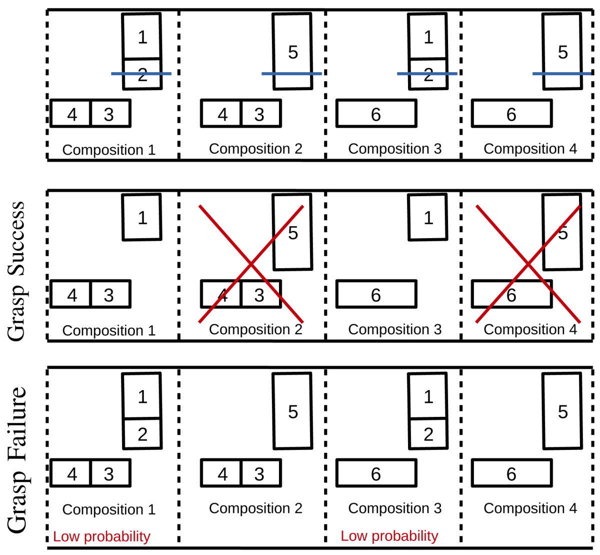


Fig. 4. Illustration of conditioning on grasp outcome using a top-down view on rectangular objects. Here the probability distribution over object compositions consists of four possible compositions with six possible objects. (Top) The robot executes blue grasp that grasps object 2. (Middle) The grasp succeeds and object 2 is removed. Compositions 2 and 4 which did not contain object 2 conflict with reality and are eliminated. (Bottom) The grasp fails either due to object 2 not existing or a bad grasp. Therefore, probability of object compositions that contain object 2 decreases.

hypothesis is correct, then the hypothesis is likely incorrect, and the belief will reflect this. Fig. 4 illustrates how grasp successes and failures influence the probability distribution over object compositions. Next, we will discuss how to sample object compositions and then show how to estimate the conditional probability of an object composition given past events.

A. *Markov Chain Sampling of Object Compositions*

To generate object compositions computationally efficiently, we first segment the image into pixel patches or segments, as shown in Fig. 2 and then combine the segments into object compositions consisting of object hypotheses. Other approaches for segmenting an image and then combining the segments into a single object composition immediately [2], or through interaction [18], [19], exist. We instead maintain a probability distribution over object compositions and make decisions based on the probability distribution.

More formally, denote with $\delta_{i,j}$ whether segments i and j are directly connected. $\delta_{i,j} = 1$ and $\delta_{i,j} = 0$ represent, respectively, direct or no direct connection. Direct connection means segments are assumed physically connected. Denote with $\boldsymbol{\delta}$ all possible direct connections. Denote with $\mathbf{h} = (h_1, \dots, h_N) \in \mathscr{H}$ an object composition, where h_k is an object hypothesis and $\mathscr{H}$ the space of object compositions. An object hypothesis h_k is a set of segment indices where all index pairs $i \in h_k$, $j \in h_k$ are connected either directly or indirectly through other segments, denoted with $c_{i,j}(h_k) = 1$ for all index pairs $i \in h_k$, $j \in h_k$ and $c_{i,j}(h_k) = 0$ otherwise. We estimate the probability distribution $P(\mathbf{h})$ over object compositions by (Gibbs) sampling individual segment pair connections $\delta_{i,j}$. Note that our sampling procedure in Algorithm 1 in Section IV-B uses direct and indirect connections to estimate the probability of a segment pair connection.

In the worst case, the dimensionality of $|\mathscr{H}|$ is $2^{N^2/2}$ w.r.t. the number of segments N. In practice, however, $|\mathscr{H}|$ is much lower since segments with only "air" between them cannot be directly connected. Often, in real-world scenes, $|\mathscr{H}|$ is a product of the dimensionality of disconnected groups of segments $\prod_i 2^{N_i^2/2}$, where N_i is the number of segments in a segment group. For exact computation this is still intractable, and therefore, we use an approximate particle representation for the probability distribution over object compositions: $P(\mathbf{h}) = \sum_i w_i \mathbf{h}_i$, where $\sum_i w_i = 1$ and $w_i \geq 0\ \forall i$.

B. Markov Chain Monte Carlo

In order to generate the particle-based probability distribution over object compositions, which can be used as a basis for decision making, we utilize Gibbs sampling (also known as Glauber dynamics) [56]–[58]. We randomly sample direct connections one at a time. Next, we will first discuss how a new Markov chain state is sampled, and then show that the proposed Markov chain is ergodic and converges to a unique distribution for nondeterministic connection probabilities. Finally, we will present a sampling procedure that aims to generate exact, independent samples.

Our sampling technique for generating a new Markov chain state exploits the fact that evaluating the probability for a single segment connection is fast as it only needs to consider local segment connections. The sampling technique consists of the following two steps: 1) randomly select two segments i and j, which may be directly connected, and 2) sample the direct connection from the probability distribution, by assuming the direct connection is disabled and by keeping other direct connections fixed to their current values. When i and j are indirectly connected, that is, part of the same object through some other connections, the probability for the direct connection between i and j depends only on the prior probability of them being part of the same object. The reason for this is due to the fact that connecting i and j would not change, which object hypothesis other segments would belong to. When i and j are not already part of the same object, the probability for the direct connection depends on the probabilities between the segment sets U and V, which connecting i and j would connect into the same object hypothesis. Fig. 5 illustrates this.

Algorithm 1 formally defines how to sample a new object composition $\mathbf{h}^*$, when given the current object composition $\mathbf{h}$, a uniformly random selected possible direct connection index w, and finally the sampled value $q \sim \text{Uniform}(0,1)$. The algorithm first chooses the direct connection candidate using w, then on lines 4 and 5 determines the segment sets U and V, which the direct connection would connect. On lines 6 and 7, the algorithm computes the probability for the segment sets U and V to belong to the same object hypothesis when i and j are connected and when not. Assuming a uniform direct connection prior, line 8 computes the direct connection probability, and line 9 finally sets the direct connection on or off using q.

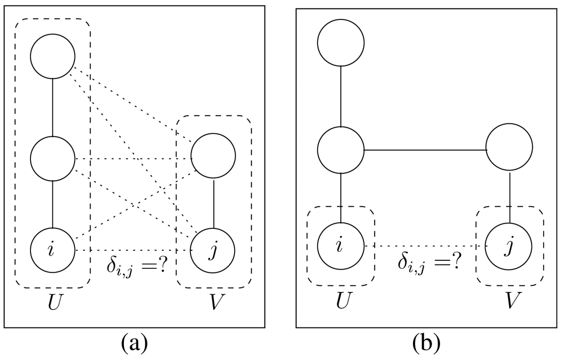


Fig. 5. Effect of indirect connections on the connection probability between segments i and j. A circle denotes a segment and a solid line denotes a connection between segments. Dotted lines denote which connection probabilities are used for sampling the connection between i and j. U and V denote the sets of segments directly or indirectly connected to i and j, respectively. (a) Because i and j are not indirectly connected we have to consider the connection probabilities between segments that will become part of the same object, that is, we have to take into account the connection probabilities between all segments in the sets U and V. (b) Because i and j are already indirectly connected we consider only the probability of the direct connection between i and j.

Algorithm 1: Sample New Object Composition.

```
1: $\mathbf{h}^* = \texttt{Sample}\ (h, w, q)$
     Input: Composition $\mathbf{h}$, random values $w$ and $q$
     Output: New composition $\mathbf{h}^*$
2:   $\delta_{i,j} \leftarrow$ The $w$th direct connection;
3:   $\mathbf{h}^* \leftarrow \mathbf{h}$ so that $\delta_{i,j} = 0$;
4:   $U \leftarrow \begin{cases} i & \text{if } c_{i,j}(h^*) = 1 \\ i \cup \{u|c_{i,u}(h^*) = 1\} & \text{if } c_{i,j}(h^*) = 0 \end{cases}$;
5:   $V \leftarrow \begin{cases} j & \text{if } c_{i,j}(h^*) = 1 \\ j \cup \{v|c_{j,v}(h^*) = 1\} & \text{if } c_{i,j}(h^*) = 0 \end{cases}$;
6:   $P(h^*|\delta^*_{i,j} = 1) \leftarrow$
     $\frac{1}{Z(U,V,h^*)} \prod_{u \in U} \prod_{v \in V} P(c_{u,v}(h^*) = 1)$;
7:   $P(h^*|\delta^*_{i,j} = 0) \leftarrow$
     $\frac{1}{Z(U,V,h^*)} \prod_{u \in U} \prod_{v \in V} P(c_{u,v}(h^*) = 0)$;
8:   $P(\delta^*_{i,j} = 1|h^*) \leftarrow \frac{P(h^*|\delta^*_{i,j}=1)}{P(h^*|\delta^*_{i,j}=1) + P(h^*|\delta^*_{i,j}=0)}$;
9:   $\delta^*_{i,j} \leftarrow \begin{cases} 0 & \text{if } P(\delta^*_{i,j} = 1|h^*) \leq q \\ 1 & \text{if } P(\delta^*_{i,j} = 1|h^*) > q \end{cases}$;
```

a) Ergodicity of the Markov chain: When the connection probability $P(c_{i,j})$ for any two segment patches is nondeterministic, i.e., $0 < P(c_{i,j}) < 1$, the Markov chain generated by Algorithm 1 is ergodic and converges to a unique distribution. Because $P(c_{i,j})$ is nondeterministic the probabilities on lines 6–8 are nondeterministic, and since we randomly select the direct connection to consider, Algorithm 1 enables or disables any direct connection with nonzero probability. Therefore, the Markov chain is ergodic and converges to a unique distribution in the limit. Note that due to the inherent uncertainty in segmentation, the condition $0 < P(c_{i,j}) < 1$ usually applies, which is also the case for the experiments in Section VI. Note also that since our sampling process is similar to Gibbs sampling,

Algorithm 2: Sample a Set of Object Compositions.

```
 1: $\mathbf{H}$ = Compositions ($N_{\text{ESS}}, N_{\text{START}}, |\mathbf{H}|$)
      Input: ESS target $N_{\text{ESS}}$
      Output: Compositions $\mathbf{H}$
 2:   $\{\mathbf{h_1}, T\} \leftarrow$ CFTP ($N_{\text{START}}$);
 3:   $t \leftarrow 1$, $\mathbf{H} \leftarrow \mathbf{h_1}$;
 4:   while (($\text{ESS}_{min}(\mathbf{H}) < N_{\text{ESS}}$) AND
 5:       ($T < T_{\text{MAX}}$)) do
 6:     while $t < T$ do
 7:       $\mathbf{h_{t+1}} \leftarrow$ Sample ($\mathbf{h_t}, w, u$);
 8:       $\mathbf{H} \leftarrow \{\mathbf{H}, \mathbf{h_{t+1}}\}$;
 9:       $t \leftarrow t + 1$;
10:     end
11:     $T \leftarrow 2T$;
12:   end
13:   $\mathbf{H} \leftarrow$ Prune $\mathbf{H}$ evenly to size $|\mathbf{H}|$
```

Algorithm 3: Coupling from the Past (CFTP).

```
 1: $\{\mathbf{H_T}, T\}$ = CFTP ($N_{\text{START}}$)
      Input: # of start compositions $N_{\text{START}}$
      Output: Composition $\mathbf{H_T}$ at time $T$
 2:   $\mathbf{H}_{\text{INIT}} \leftarrow \{\{\mathbf{h_1}|\boldsymbol{\delta} = 1\}, \{\mathbf{h_2}|\boldsymbol{\delta} = 0\}$,
 3:       $\{\mathbf{h_3}, \ldots, \mathbf{h}_{\mathbf{N}_{\text{START}}}|\boldsymbol{\delta} = \text{random}\}\}$;
 4:   $T \leftarrow 1$;
 5:   repeat
 6:     $T \leftarrow 2T$, $\mathbf{H_T} \leftarrow \mathbf{H}_{\text{INIT}}$;
 7:     $w_T, \ldots, w_{T/2} \leftarrow$ Random;
 8:     $u_T, \ldots, u_{T/2} \leftarrow$ Random;
 9:     for $t \leftarrow T$ to 1 do
10:       $\mathbf{H_{T-t+1}} \leftarrow \emptyset$;
11:       foreach $\mathbf{h_{T-t}} \in \mathbf{H_{T-t}}$ do
12:         $\mathbf{H_{T-t+1}} \leftarrow \mathbf{H_{T-t+1}} \cup$
13:           Sample $\mathbf{h_{T-t}}, w_t, u_t$;
14:       end
15:     end
16:   until $|\mathbf{H_T}| = 1$;
```

convergence of the algorithm could be derived based on the sufficient conditions for convergence of Gibbs sampling [59].

b) MCMC procedure: We want our MCMC approach to produce independent samples from the correct distribution. Therefore, our MCMC approach first aims to get an exact sample, that is, a sample from the correct probability distribution, then continue sampling until having enough independent samples (**H** in Algorithm 2). Algorithm 2 shows the proposed MCMC approach. Since the Markov chain state is a discrete combination of binary variables, each variable denoting whether two segments are directly connected, we base our approach on the coupling from the past (CFTP) [60] technique, which aims at providing exact samples. The basic idea of CFTP is to run Markov chains from each possible state with the same random numbers, starting further back in time until the chains collapse. Here, collapsed means that all the chains arrive at the same state even though they start from different states, which removes the initial state bias (see [60] for more details).

For monotone [60] and antimonotone [61] Markov chains, only two starting states are needed. However, our chains are neither monotone nor antimonotone. Due to the large number of states, we start CFTP from a limited dispersed set of states: the all connected, all disconnected, and from a fixed number of randomly selected states. We can make the collapsed sample more likely to be exact by increasing the number of starting states: when the starting states cover the whole state space the collapsed sample will be exact [60]. Algorithm 3 shows the CFTP procedure we use. In the experiments, we used 100 starting states (N_{START} in Algorithms 2 and 3).

After CFTP, we start the actual sampling from the collapsed sample produced by CFTP, and doubling the sampling horizon until having enough independent samples. We use the minimum of the effective sample size (ESS) [62] over all possible direct connections (N_{ESS} in Algorithm 2) as a lower bound estimate for the number of independent samples. Sampling stops when the estimate for independent samples is large enough. We also use a hard limit on the number of generated samples (T_{MAX} in Algorithm 2).

C. Probability of an Object Composition Given Past Events

Object composition sampling requires assessing the probability of an object composition conditioned on past grasp events. We start with a more general definition and then proceed on to conditioning on past grasp events. In general, the probability of an object composition $h = (h_1, \ldots, h_N)$, where h_i is a single object hypothesis, depends on the sequence of past actions and observations $\theta_t = (a(0), o(1), a(1), o(2), \ldots, a(t-1), o(t))$, where t denotes the current time step. Formally, the conditional probability of the object composition defined above, assuming uniform priors, independent object hypotheses, and independent history events is:

$$P(h|\theta_t) = \prod_{i=1}^{N} P(h_i|\theta_t) = \prod_{i=1}^{N} \prod_{k=1}^{t} P(a(k-1), o(k)|h_i). \quad (1)$$

For object hypothesis manipulation, we maintain a history of unique executed grasps. In our model, there is no need to remember multiple identical grasps as a previously failed grasp cannot succeed again unless the occlusion of the object, for which the grasp is optimized, changes. The composition probability, conditional on past independent grasps $(\text{grasp}_1, \ldots, \text{grasp}_M)$, is

$$P(h|\theta_t) = \prod_{i=1}^{N} \prod_{k=1}^{M} P(\text{grasp}_k|h_i). \quad (2)$$

D. Hallucinating Hidden Objects

Due to occlusions, the robot cannot see the complete scene as objects can occlude each other which is illustrated in Figs. 6 and 17. To allow a robot to reason about hidden objects, we enable it to hallucinate hidden objects into the object composition distribution. For hallucinating hidden objects, we utilize the key insight that the probability of hidden objects depend on the amount of hidden space, and the object occupancy density of that

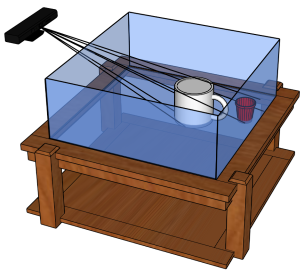


Fig. 6. Illustration of hidden objects. The transparent "glass" box denotes the workspace inside which the robot currently operates. A white cup fully occludes a red cup. To reason about hidden objects, we use the insight that only a limited amount of objects can be hidden behind other objects. We assume that the probability for an object to be hidden behind another object depends on the volume hidden behind the occluding object, and, the object occupancy density.

space. Using these assumptions, we transform the probability distribution over visible objects into a probability distribution over both visible and hidden objects.

Let us denote the visible volume with V_{visible} and the hidden volume with V_{hidden}. These quantities represent how much workspace the robot can and cannot see. To compute the hidden volume V_{hidden}, we project each visible voxel onto the workspace boundary from the origin (camera location) and estimate the volume between the rectangular visible voxel and the voxel projected onto the workspace boundary. For each segment patch, we get the hidden volume by summing over the segment voxels. Summing over all segment patches yields then the hidden volume V_{hidden}. Let us denote the average height of the visible segment patches with d, the cube floor area with A_C, and the average number of objects with n_{objects}. The minimum for n_{objects} is set to 1. We estimate the total volume of interest V_{total} for a cubical workspace as $V_{\text{total}} = 2dA_C$. The visible volume, V_{visible}, is then total volume minus hidden volume: $V_{\text{visible}} = V_{\text{total}} - V_{\text{hidden}}$.

The object occupancy density is $p_{\text{objects}} = n_{\text{objects}}/V_{\text{visible}}$. For each segment patch i, we compute the expected number of hidden objects n^i_{objects} behind the patch as $n^i_{\text{objects}} = p_{\text{objects}} V^i_{\text{hidden}}$, where p_{objects} represent the object occupancy density and V^i_{hidden} the patch's hidden space volume. In the experiments, although the model allows for multiple hidden objects behind an object, we assume for simplicity that there is at most one object behind an occluding object.

The sampling of hidden object hypotheses is performed after sampling nonhidden object hypotheses described in Algorithm 2. For each nonhidden object hypothesis in each object composition $\mathbf{h_t}$ in $\mathbf{H}$, we sample whether a hidden object hypothesis is behind it. A hallucinated object is sampled according to the probability $1 - \Phi((1 - n^i_{\text{objects}})/\sqrt{n^i_{\text{objects}}/4})$, which corresponds to the Gaussian probability of having at least one hidden object when the mean of the Gaussian is n^i_{objects} and the standard deviation is $\sqrt{n^i_{\text{objects}}/4}$.

Depending on the task, we can estimate probabilities for hidden object properties based on visible objects and a priori information. For example, in the object search experiments, we estimate the probability for hidden objects to be red based on the average probability for redness. Now that we have discussed how to estimate the belief over object compositions, we will next discuss how to plan manipulation actions using the belief.

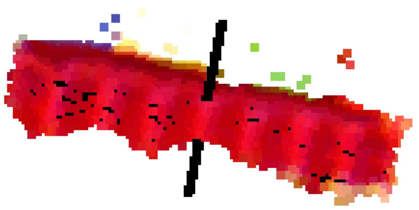


Fig. 7. Example of a grasp on a red Lego block. The black line shows where the robot finger tips are positioned when grasping (the black line goes through finger tip positions). The robot tries to grasp a narrow part of the point cloud. In detail, the grasp is based on the second PCA eigenvector of the point cloud projected on the 2-D plane going through the wrist of the robot (not shown).

V. Manipulation Planning

After the robot has estimated the current belief, it uses that belief to decides on its next action. As discussed earlier, we use a POMDP for decision making. The POMDP chooses a high level action such as the object to grasp and the location for placing it. The POMDP enables the robot to take actions that support long term behaviors, including information gathering actions, such as removing obstacles for finding hidden objects. Next, we discuss parts of our system that the POMDP requires for planning. We will discuss the actions available to the robot, how to sample a new state and observation, and how to compute the observation probability.

A. Actions

In our problem setting, the robot may grasp an object and move it. We employ top-down grasping. For selecting the finger distance and rotation of the robot hand, we use a simple approach based on computing a vector at a narrow part of the unknown object with principal component analysis (PCA). An example grasp is shown in Fig. 7. Next, we discuss details on how a grasp is computed for an object hypothesis, how to limit the total number of possible grasps for all object hypotheses, and finally how to compute grasp success probabilities.

1) Details on Grasp Computation: The approach:

1) projects the point cloud PC_1 of the object hypothesis onto a plane aligned with the wrist of the down-pointing robot hand;
2) makes the point density of the projected point cloud uniform, resulting in point cloud PC_2;
3) projects the centroid of PC_2 toward PC_1 along the wrist-plane normal, by the distance between the centroids of PC_1 and PC_2, to get the grasp centroid.

Next, the approach computes the PCA decomposition of PC_2. For getting the final two grasp contact points, we project two points along the second eigenvector in opposite directions from the grasp centroid and select the two points from PC_1 closest to the projected points. The reason for choosing the second eigenvector is because, in PCA, the second eigenvector aligns with the second-largest variance in the data, which is the width for a long and narrow object. Finally, to prevent collisions with objects, we check whether some part of another object hypothesis blocks the direct path up from the two grasp contact points, and if so, set the grasp success probability to zero.

2) Restricted Action Set: In order to restrict the computational load, we bound the number of possible grasps, and thus actions, by a predefined maximum number (in the experiments 20). Instead of restricting the number of possible object hypotheses, we select a subset of all hypotheses to use for grasping.

Optimally, we want a set of grasps that yields the best policy among all possible grasp sets. However, because the best policy is unknown, we settle for computing an action set A that maximizes the expected grasp success probability

$$A = \arg\max_{A} \sum_{h_i} P_{\text{grasp prob}}(\text{SUCCESS}|h_i, A)P(h_i) \quad (3)$$

where $|A|$ is the number of actions, $P(h_i)$ is the probability of hypothesis h_i, and $P_{\text{grasp prob}}(\text{SUCCESS}|h_i, A)$ is the grasp success probability. The grasp success probability refers to the probability of successfully grasping and moving an object hypothesis h_i when the robot chooses the best action from the action set A.

One option to find the action set A is to solve an integer linear program. Unfortunately, such a program is in the worst case NP-hard. As an approximation, we use a greedy approach that incrementally selects the object hypothesis, which increases the expected grasp success probability the most. In the experiments, the expected grasp success probability using a restricted action set usually remained close to the probability with the complete set of possible actions.

3) Grasp Success Probability: We parameterize A grasp by the distance between the finger tips, rotation of the hand, and the location of the robot wrist. The grasp success probability is the product of a *grasp quality* and an occlusion specific grasp probability. To calculate the occlusion-specific grasp probability, we use the grasp probability model proposed in [52], and to calculate the grasp quality, we use heuristics. Due to using heuristics, the grasp success probability should perhaps be called a pseudoprobability instead of an actual probability. Nevertheless, as the grasp quality is between 0 and 1, the grasp success probability acts like a probability. Henceforth, for convenience, we use the short hand "probability" when referring to the grasp success probability.

When computing grasp probabilities, we take previously executed grasps into account. For instance, a failed grasp cannot succeed again, unless the occlusion of the object changes for which the grasp is optimized (when the occlusion changes the grasp usually changes also). Grasp quality is intended to evaluate the quality of a grasp on another hypothesis than that it was optimized for. Grasp quality is equal to 1 when using a grasp, which was computed for the same object hypothesis that the robot tries to grasp. The grasp quality decreases when the grasp centroid moves away from the optimized grasp centroid, and becomes zero when it is outside the object.

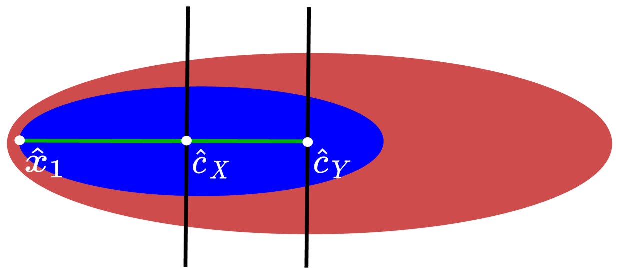


Fig. 8. Visualization of grasp quality computation when the grasp optimized for the red ellipsoid object grasps a blue ellipsoid object. Black lines denote grasps (see Fig. 7 for an example grasp). Grasp quality is based on projecting point clouds onto the robot's 2-D wrist plane. We compute grasp quality as the distance from the blue object's projected grasp centroid $\hat{c}_X$ to the blue object edge $\hat{x}_1$ divided by the distance from the red object's projected grasp centroid $\hat{c}_Y$ to $\hat{x}_1$.

How we compute the grasp quality is illustrated in Fig. 8. More formally, we compute a grasp quality for grasping object hypothesis X with a grasp optimized for object hypothesis Y by first checking whether the grasp of Y intersects the point cloud of X. If not, grasp probability is zero. Otherwise, we first compute the centroids c_Y and c_X that represents the intersecting grasp on Y and the centroid of the best grasp for X, respectively. Then, we project the point cloud of X, the point cloud of Y, c_x, and c_Y onto the 2-D robot wrist plane. We denote these projections with $\hat{X}$, $\hat{Y}$, $\hat{c}_x$, and $\hat{c}_Y$, respectively. The grasp quality is finally computed as follows.

1) Find the point $\hat{x}_1$ by projecting a point starting from $\hat{c}_Y$ through $\hat{c}_X$ until the edge of $\hat{X}$.
2) Compute the grasp quality as $\frac{||\hat{c}_X - \hat{x}_1||_2}{||\hat{c}_Y - \hat{x}_1||_2}$.

In other words, the grasp quality decreases when the grasp centroid c_Y moves further away from the grasp centroid c_X.

B. Temporal Model of the World

In order to use the POMDP method in [52] for planning, we need to model the evolution of the world state over time, which requires state transition and observation probabilities. Because our probability distributions use a state particle representation, we need, in particular, an approach for sampling states and observations and estimating the likelihood of a state particle given an observation. Next, we discuss how to accomplish these tasks.

1) State Sampling: As discussed earlier, a world state consists of an object composition, $h = (h_1, \ldots, h_N)$, where each h_i consists of a semantic object location, attributes, and a history. To sample a new state for a grasp action a, first select the object hypothesis h_i that has the highest grasp probability for a. Then, sample grasp success of a on h_i according to the grasp success probability defined in Section V-A3. Finally, execute the grasp, and if it fails, add it to the grasp failure history of h_i. If, instead, the grasp successfully moves an object, change the semantic location of the object to the destination location.

2) Observation Sampling: After executing the grasp action, the robot observes which object moved. In the case of a successful move, the robot observes the attributes (color in the experiments) of a limited number of objects behind the moved object. Assuming independence between attribute observations, the observation probability is $\prod_i P(o_i|h_i)$, where o_i is the observation of h_i. As in [52], $P(o_i|h_i)$ is computed from the occlusion of h_i and the attribute instances of h_i.

3) Observation Probability: The probability of making an observation o in state s is zero if the moved object hypothesis differs from the observed one, or if the move fails and the attribute observations do not match with previous attribute observations. Otherwise, the probability is defined by $\prod_i P(o_i|h_i)$ discussed earlier.

VI. Experiments

The experiments are designed to test whether taking object composition uncertainty into account improves performance in robotic manipulation. Our hypothesis is that modeling the uncertainty explicitly is beneficial. Furthermore, we hypothesize that planning actions to accomplish a task under object composition uncertainty improves the performance. The two main questions we want to answer are as follows:

1) Does taking segmentation uncertainty in decision making into account increase performance?
2) Does multistep planning based on a distribution over object compositions increase performance compared to greedily choosing actions?

In order to provide justified answers to these questions, we compare the baseline and proposed methods in two different tasks with two different robot arms to show the generality of the methods. In the first task, the robot needs to remove objects from a table using a Kinova Jaco robotic arm. In the second task, the robot needs to search for red objects from a pile of objects in simulation and using a real Franka Panda robot arm. In the first task, we focus purely on comparing decision making with and without taking segmentation uncertainty into account. In the second task, multistep planning is needed to utilize both information gathering and task directed actions under object composition uncertainty. Therefore, we evaluate also our POMDP-based planning method in the second task.

A. Segmentation and Estimating Grasp Probabilities

In the experiments, segmentation and segment-pair probability computation is performed using the approach presented in [2]. In short, the approach assigns probabilities to segment pairs using support vector machines (SVMs) trained on RGB-D data of household items, which are not in all ways similar to the toys we use in the experiments.

At each time step, the sensor captures an RGB-D image of the scene, and the system segments the image into patches and computes a prior probability for each patch pair to belong to the same object using the approach in [2]. In more detail, Richtsfeld *et al.* [2] group neighboring pixels into clusters and fits planes and B-splines onto the patches to get parametric models (see Fig. 2, for examples, of segmented patches). The approach in [2] computes, for each patch pair, a set of features based on the texture, distance of the patches from each other, and other properties. Finally, the approach feeds the computed features into a trained SVM and scales the output into a probability indicating whether the patches belong to the same object or not. We use the approach in [2] to compute prior probabilities $P(c_{i,j})$ for all segment/patch pairs, where $P(c_{i,j}=1)$ defines the prior probability for i and j to be part of the same object. Note that, our approach can also use other methods to oversegment and estimate patch pair probabilities.

We use $P(c_{i,j})$ in the MCMC procedure in Algorithm 2 to compute a probability distribution over object compositions (see Fig. 2, for examples). The number of MCMC samples should be chosen according to the computational budget. In the experiments, we used $|\mathbf{H}| = 2000$ samples. Because the minimum lower bound estimate ESS underestimates the real ESS it should be lower than the number of samples: we used a target ESS of $N_{\text{ESS}} = 200$, which is a tenth of the number of samples. The number of CFTP starting states influences the independence of the first sample w.r.t. the starting state. We used $N_{\text{START}} = 100$ CFTP starting states in the experiments. The hard limit for the number of samples generated was $T_{\text{MAX}} = 131,072$.

Grasp probabilities. For grasp probability, we used the parameters estimated for the coffee cups in [52] and set the first color observation parameter to -0.5 and the second to -0.02 for both red and nonred observations (see [52] for details). We emphasize that the models were not optimized for our particular objects as a robot acting in the real world should be able to generalize to new objects.

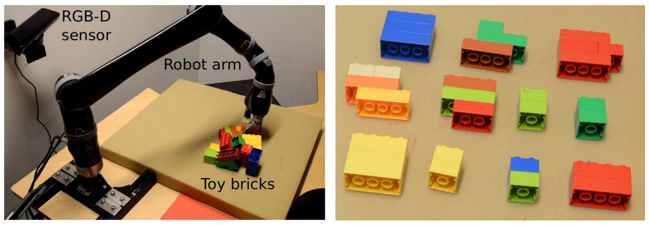
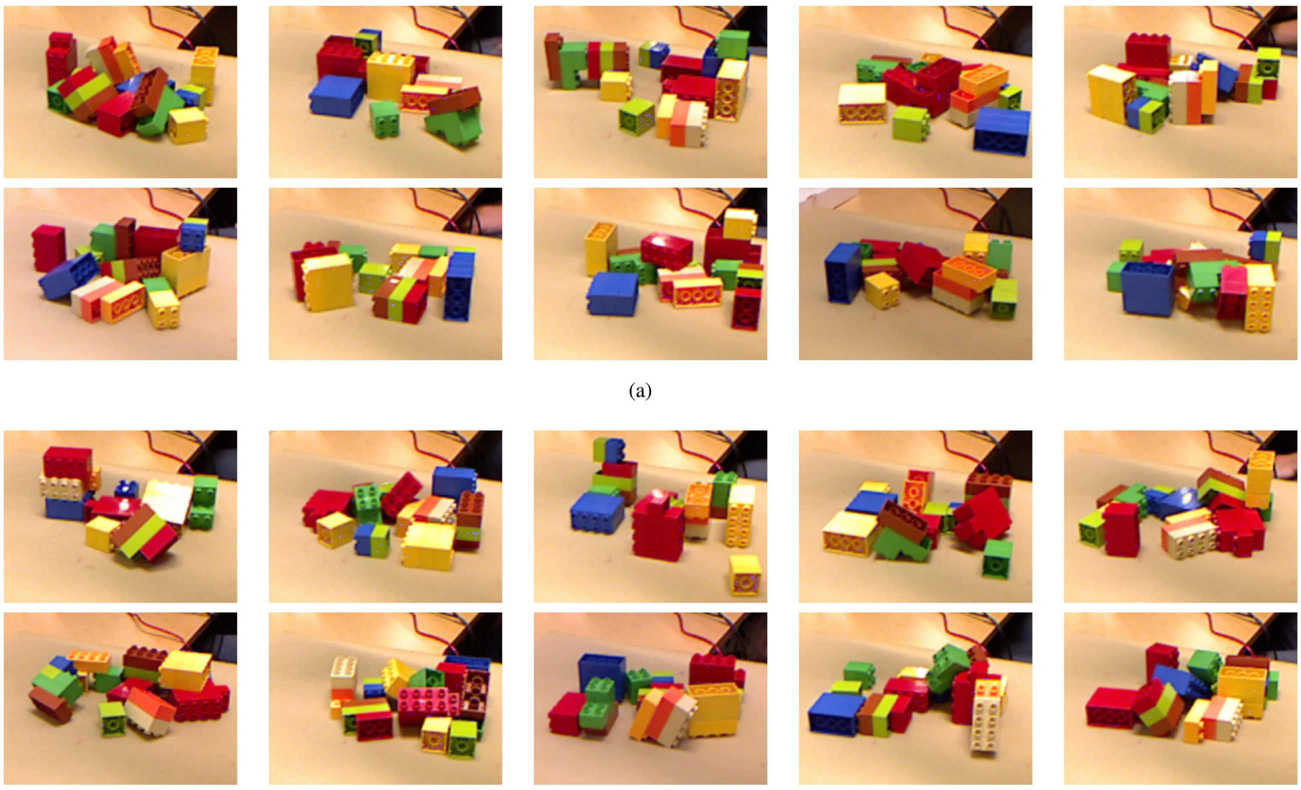


Fig. 9. In table clearing, we use an RGB-D sensor for visual input and a 6-DOF Kinova Jaco arm for grasping randomly placed toy bricks. (Left) Overall experimental setup. (Right) Toy bricks used in the experiment.

B. Experiments: Clearing a Table

Fig. 9 shows the experimental setup for table clearing. In the setup, a Kinect RGB-D sensor captures images of the scene and a 6-DOF robotic Kinova Jaco arm tries to move as many toy bricks away from the table as possible. Since we do not assume any prior information in advance, such as geometric or colour information, and because the bricks are in a pile, segmenting the bricks correctly is difficult. For clearly separated known objects one could possibly use standard segmentation methods. To generate random scenes of toy brick, we shook a box containing toy bricks and emptied it into a specific area on a table, as shown in Fig. 9. Fig. 10 presents the ten random scenes for each method ordered so that the first scene produced highest utility and the last scene the lowest utility. Fig. 11 shows an example segmentation for one of the scenes. The goal was to remove a toy brick from the table at each time step. In the application, action a specifies the object hypothesis to grasp and



(a)

(b)

Fig. 10. Cropped kinect RGB images of the 20 randomly generated scenes in table clearing. (a) Random scenes in "best segmentation" evaluations, ordered according to experimental success from best to worst. (b) Random scenes in "maximum utility" evaluations, ordered according to experimental success from best to worst.

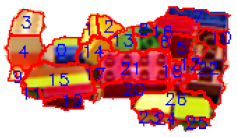


Fig. 11. Example segmentation of Lego bricks.

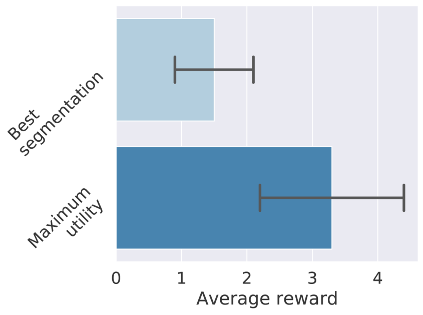


Fig. 12. Results in table clearing. The robot grasps and moves toy bricks away from the table. The bar plot shows the average number of successful moves (average reward) and the 95% confidence interval in ten experimental runs for each method. The "maximum utility" method performed significantly better than "best segmentation" ($p = 0.014$ in the Mann–Whitney U one-sided test [63]).

move away. The Kinova Jaco arm could only grasp an object hypothesis if it fitted inside the maximum grasp width of the gripper, which we set to 4 cm. The robot is rewarded 1 for a successfully grasped and moved object and 0 otherwise.

In this experiment, we do not perform POMDP planning. Instead, we only optimize the expected immediate reward to evaluate whether utilizing a probability distribution over object compositions increases performance compared to utilizing only the best object composition found. We compare two methods. The first method, called "best segmentation," finds the most likely object composition, $\mathbf{h}^*$, and then computes the action a that maximizes the utility or immediate reward function $R(\mathbf{h}^*, a)$. In table clearing, "best segmentation" tries to grasp the object which has the highest grasp success probability in the most likely object composition. The second method, called "maximum utility," corresponds to maximizing the expected immediate reward where the expectation is taken over possible object compositions. In table clearing, "maximum utility" tries to grasp the object, which has the highest grasp success probability weighted by the probability of the object to exist in an object composition.

a) Table Clearing: Results and Discussion: Fig. 12 presents, for each method, the number of successful moves (a maximum of six moves per scene) in ten experimental runs. The "maximum utility" method performed significantly better than "best segmentation" (the p-value was 0.014 in the Mann–Whitney U one-sided test [63]). To qualitatively compare the

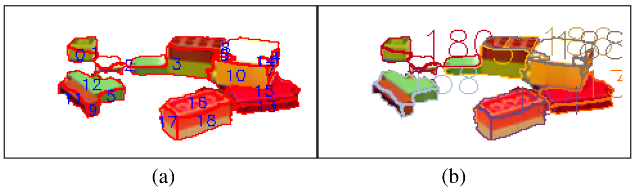


Fig. 13. “Maximum utility” is able to move an object when “best segmentation” fails. Time step 6 in the sixth scene in Fig. 10(b): (a) Segmented patches, (b) the most likely object composition (probability 0.271). “Maximum utility” successfully grasps the object hypothesis consisting of patches 0 and 1. However, “best segmentation” stops because the hypothesis formed by patches 0, 1, and 2 in (b) exceeds the robot’s maximum grasp width which was set to 4 cm.

two methods, we recorded decisions by both, although only one method operated the robot arm in each scene. That is, we ran one method and at the same time output the decisions, which the other method would have made for the same object compositions. In the scenes in Fig. 10(a), even though graspable object hypotheses were still available, “maximum utility” would have finished execution early 3 times and “best segmentation” finished early 10 times. Similarly, in the scenes in Fig. 10(b), “maximum utility” finished execution early 3 times and “best segmentation” would have finished early 23 times. A typical failure case for the “best segmentation” method was that the most likely composition it produced consisted of hypotheses that all exceeded the robot’s maximum grasp width of 4 cm. Fig. 13 shows an example of such a situation.

The time to complete one object movement, which includes image processing, segmentation, generating the particle-based probability distribution, and moving an object (hypothesis), took, on average, 79.9 s. Of this time, our MCMC approach took, on average, 8.8 s (11%). However, the time needed for MCMC depends on the number of particles and CFTP starting states and can be adjusted if necessary.

Regarding segmentation, Fig. 13(a) shows that undersegmentation happens sometimes. In general, it is better to oversegment too heavily than undersegment but this applies to all oversegmentation approaches including the oversegmentation approach utilized by the two comparison methods. Interestingly, grasps were sometimes successful even when the segmentation of the grasped object did not correspond to a real object. This happened, for example, when the robot grasped the segmented top of an object and managed to successfully move the complete object.

Overall, one of the main reasons the robot achieved such a high grasping performance was because the utility function did not include unnecessary constraints. However, for other applications, such as moving fragile objects, the utility function could, for instance, penalize grasping incorrectly segmented objects if this would decrease the chance of dropping the object.

C. Experiments: Object Search

In the object search experiments shown in Fig. 14, a Franka Emika Panda robot arm equipped with a custom parallel gripper was tasked to find and move an unknown number of fully red objects into a red box. For every red object, the robot places in the red box, we increase the score (utility) by 1 and for every nonred object placed in the red box, we decrease the score by 1. To remove occlusions that hinder color detection and grasping, the robot may also move objects into a green box without any direct effect on the total score, i.e., we neither add nor subtract points for such an action.

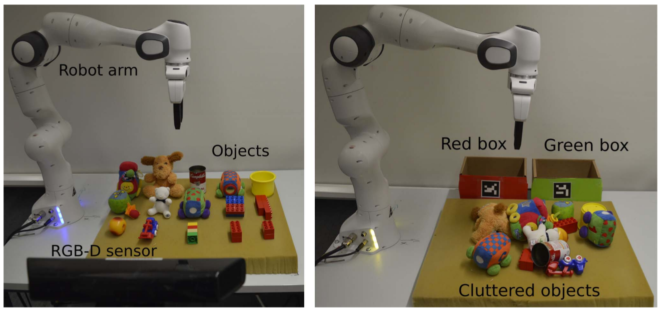


Fig. 14. Experimental setup for object search. Based on RGB-D data from a Microsoft Kinect, a 7-DOF Franka Panda robotic arm tries to move red objects into a red box. (Left) Robotic setup with all objects arranged on the table. (Right) In actual experiments, objects are dumped into the workspace resulting in a cluttered scene. The robot tries to move fully red objects into the red box and may put objects into the green box to reduce clutter.

The complete setup along with the objects used are shown in Fig. 14. We chose these objects as they differed in size, rigidness, color, and texture. A Kinect RGB-D sensor was used to observe the objects on the table. In order to get diverse and unbiased scenes for each trial, we placed the objects inside a box, which was shaken and emptied into the workspace of the robot. If an object ended up outside the workspace the process was repeated.

Object search [47], [64] is a well known task in robotics but existing approaches do not explicitly account for composition uncertainty when planning actions. Of recent work, Danielczuk *et al.* [64] performed object search for a prespecified target by segmenting an image in each time step and combining the segmented image with an “occupancy distribution” that describes where the object could be hidden. We consider a more general case without a specific target object and without knowing the sizes of the objects we are searching for. Xiao *et al.* [47] used a POMDP model for object search but assumes perfect object composition segmentation with some uncertainty in object locations and utilizes *a priori* knowledge of number and models of objects for finding hidden objects. We evaluated the following four different methods.

1) Best grasp for objects observed red in the most likely segmentation (“best segmentation”).
2) Grasp for the highest expected immediate reward (“maximum utility”).
3) POMDP multistep planning without hallucinating hidden objects (“POMDP”).
4) POMDP multistep planning with hallucinating hidden objects (“POMDP with hallucination”) .

The first two methods are similar to the ones used in the table clearing experiment in Section VI-B. The “POMDP” method plans actions based on a probability distribution over object compositions using the POMDP model defined in Sections III

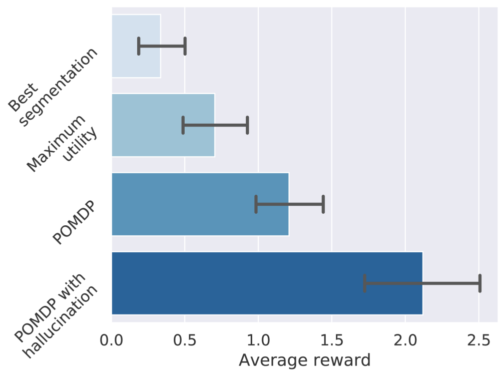


Fig. 15. Simulated object search. The bar plot shows the average reward and the 95% confidence interval for each method. The differences between the method performances is statistically significant. Please, see the main text for further discussion.

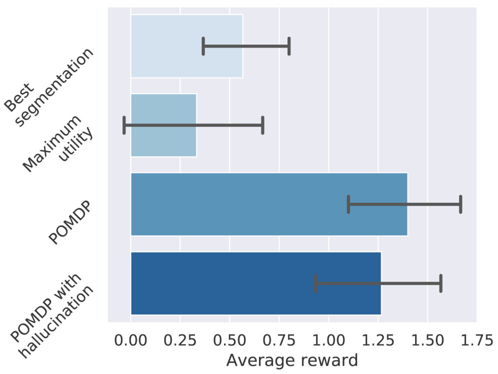


Fig. 16. Object search with a Franka Panda robot arm. The bar plot shows the average reward and the 95% confidence interval for each method. The performance difference between the POMDP methods and the greedy methods is statistically significant. Please, see the main text for further discussion.

and V. The "POMDP with hallucination" method extends the third method with a probability model for hidden objects as defined in Section IV-D. The probability for hidden objects to be red was estimated as the average fraction of visible red objects linearly scaled into $[0.2, 0.8]$ to always allow for both nonred and red hidden objects.

1) Object Search With Simulated Dynamics: In the object search simulation experiments, we used the initial RGB-D scenes captured in the robot experiments discussed in Section VI-C2 but simulated dynamics and observation probabilities. Fig. 15 shows the results. For statistical analysis we ran a Kruskal–Wallis test and then Posthoc Conover tests revealing that all methods' average reward differed statistically significantly: "POMDP with hallucination" from "POMDP" ($p = 0.002$), "POMDP" from "maximum utility" ($p = 0.003$), and "maximum utility" from "best segmentation" ($p = 0.021$).

2) Object Search on Real Hardware: Fig. 14 shows the experimental setup and one of the scenes we used for object search with a Franka Panda robot. As in previous experiments, we randomly generated scenes by adding all objects into a box, shook the box, and emptied the content into the robot's workspace.

Fig. 16 summarizes the results: POMDP methods significantly outperformed the greedy approaches. The main reasons the POMDP methods outperform the greedy approaches is that they utilize information gathering actions and remove occluding nonred objects as shown in Fig. 17 and plan actions over distributions of compositions. Planning over a distribution of compositions discourages our approach from stopping prematurely, contrary to approaches that may stop when no red object is seen in the most probable distribution, which is the main failure case for the greedy baselines. Fig. 18 shows the running times for all comparison methods for the different computational components at different phases of the manipulation task.

Planning using a distribution of compositions also enables complex reasoning, which the example in Fig. 19 illustrates. This example shows that the robot can verify whether a nonred object

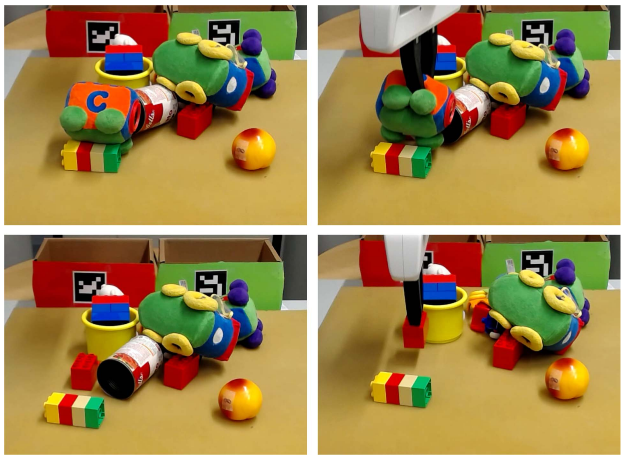


Fig. 17. "POMDP with hallucination" method removes objects occluding a large space to reveal a fully red object. (Upper Left) Initial scene. (Upper Right) Picking up nonred object occluding a large space. (Bottom Left) A fully red hidden object is revealed. (Bottom Right) The robot now picks up the red object and moves it into the red box.

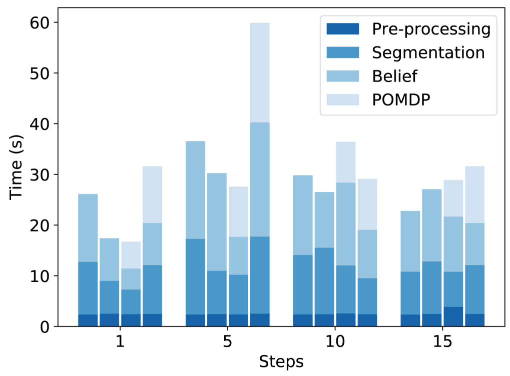


Fig. 18. Average running time for each comparison method at different phases of the manipulation task, that is, the time step. The running time is split into preprocessing, segmentation, belief generation, and the POMDP solver components. The four comparison methods are, from left to right, "best segmentation," "maximum utility," "POMDP," and "POMDP with hallucination." Segmentation, belief generation, and the POMDP solver take roughly the same amount of processing time. This is by design since the running time of belief generation and the POMDP solver can be controlled while increasing or decreasing the exactness of the computations.

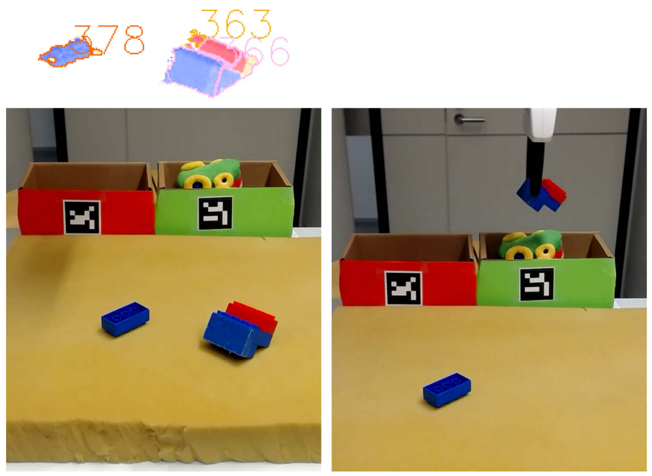


Fig. 19. (Left) The object composition on the top indicates that the red lego block is either separated or part of the larger blue object. A reasonable approach here is to grasp the larger blue part and move it to the green box. Then if the composition is separable the red lego will be left in the scene while if the red is part of the larger blue object the red-blue object will disappear which is also fine. (Right) Robot picked up the larger blue object. In this case the red segment was part of the blue object.

hypothesis forms an object together with a red object hypothesis by grasping the blue object and removing it. If the hypotheses do not belong together, the robot has removed a nonred object and can collect the red object next. If, however, the red and nonred hypotheses are part of the same object, the robot has still only removed a nonred object and clarified the situation.

Regarding removing occluding objects in the scene, “POMDP with hallucination” removed 21 occluding nonred objects to reveal a fully red object and “POMDP” 18 occluding nonred objects. Fig. 17 shows one successful example where “POMDP with hallucination” removes an obstacle to reveal a fully red object which is subsequently moved into the red box. We hypothesize that with bigger and/or more occluding nonred objects or fewer red objects in total the performance of “POMDP with hallucination” may outperform the regular “POMDP” method as it has an inherent advantage of removing occluding objects in the scene.

D. Limiting Assumptions

In grasping, one limitation of using the second eigenvector of PCA for computing the grasp, which is discussed in Section V-A1, is that the grasp may not work on nonconvex object hypotheses. However, due to the large variety of object hypotheses computed for an individual object, it is often possible for the robot to find and grasp a convex part of a nonconvex object, which was the case in the experiments, where the robot successfully grasped various object shapes.

Another limitation is that the current model also assumes that failed grasp attempts do not move objects. In future work, this kind of dynamics could be incorporated into the model by adding pose information to each object hypothesis and simulating movement for failed grasps.

A third limitation of the action definition is that actions do not take object self-occlusion into account which may lead to collisions when moving an object. In the experiments, we did not observe this problem. Nevertheless, future work could account for self-occlusion by using object shape-completion [65] or estimating safety margins for object movement.

VII. Conclusion

Enabling robots to operate in cluttered unknown environments is essential for many tasks, including waste segregation, agile manufacturing, service robotics, and rescue robotics. However, a lack of object models and noisy partial camera views make manipulation in such environments difficult due to uncertain object compositions. Therefore, for robots to successfully perform such tasks, they should consider multiple possible object compositions and, in particular, plan actions that take all possible object compositions now and in future time steps into account. Based on this insight, instead of utilizing only the most likely object composition, we proposed a POMDP planning framework for planning manipulation actions in the space of object compositions. In object search and table clearing experiments, our approach outperforms an approach based on the most likely object composition. Moreover, we empirically showed that long-term planning outperforms greedy approaches when planning over a distribution of object compositions.

Despite the promising results, we only considered settings with static objects. Future work should, therefore, includes manipulation of dynamic moving objects under occlusion. Another avenue of future work is to include prior knowledge from other sensor modalities, such as tactile, into our probabilistic world model.

Overall, we expect the main idea of planning based on a distribution of object compositions to transfer well to other application domains. One such example is autonomous driving, where high uncertainties due to occlusions and weather conditions cause difficulties determining which parts of the scene are pedestrians, cars, or buildings? In this case, planning based on the distribution of object compositions is a safer alternative than relying on the most likely, but possibly incorrect, object composition.

References

[1] J. Bohg et al., “Interactive perception: Leveraging action in perception and perception in action,” *IEEE Trans. Robot.*, vol. 33, no. 6, pp. 1273–1291, Dec. 2017.
[2] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze, “Learning of perceptual grouping for object segmentation on RGB-D data,” *J. Vis. Commun. Image Representation*, vol. 25, no. 1, pp. 64–73, 2014.
[3] J. Pajarinen and V. Kyrki, “Decision making under uncertain segmentations,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 1303–1309.
[4] J. Pajarinen and V. Kyrki, “Robotic manipulation in object composition space,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 1–6.
[5] R. M. Haralick and L. G. Shapiro, “Image segmentation techniques,” *Comput. Vis., Graph., Image Process.*, vol. 29, no. 1, pp. 100–132, 1985.
[6] N. R. Pal and S. K. Pal, “A review on image segmentation techniques,” *Pattern Recognit.*, vol. 26, no. 9, pp. 1277–1294, 1993.
[7] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[8] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, 2004.

[9] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *Proc. Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 1817–1824.

[10] A. K. Mishra, A. Shrivastava, and Y. Aloimonos, "Segmenting "simple" objects using RGB-D," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 4406–4413.

[11] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze, "Segmentation of unknown objects in indoor environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 4791–4796.

[12] M. Danielczuk et al., "Segmenting unknown 3D objects from real depth images using mask R-CNN trained on synthetic data," in *Proc. Int. Conf. Robot. Autom.*, 2019, pp. 7283–7290.

[13] C. Xie, Y. Xiang, A. Mousavian, and D. Fox, "The best of both modes: Separately leveraging RGB and depth for unseen object instance segmentation," in *Proc. Conf. Robot Learn.*, 2020, pp. 1369–1378.

[14] D. Morrison, J. Leitner, and P. Corke, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," in *Proc. Robot.: Sci. Syst.*, 2018.

[15] A. Mousavian, C. Eppner, and D. Fox, "6-DOF graspnet: Variational grasp generation for object manipulation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 2901–2910.

[16] J. Mahler et al., "DEX-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," in *Proc. Robot: Sci. Syst.*, 2017, pp. 1–10.

[17] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 2442–2447.

[18] D. Beale, P. Iravani, and P. Hall, "Probabilistic models for robot-based object segmentation," *Robot. Auton. Syst.*, vol. 59, no. 12, pp. 1080–1089, 2011.

[19] H. van Hoof, O. Kroemer, and J. Peters, "Probabilistic segmentation and targeted exploration of objects in cluttered environments," *IEEE Trans. Robot.*, vol. 30, no. 5, pp. 1198–1209, Oct. 2014.

[20] D. Katz, M. Kazemi, J. Bagnell, and A. Stentz, "Clearing a pile of unknown objects using interactive perception," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 154–161.

[21] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images," in *Proc. Proc. IEEE Int. Conf. Comput. Vis.*, 2001, vol. 1, pp. 105–112.

[22] T. Zhao and R. Nevatia, "Bayesian human segmentation in crowded situations," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2003, vol. 2, pp. II–459.

[23] Z. Tu and S. Zhu, "Image segmentation by data-driven Markov chain Monte Carlo," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 657–673, May 2002.

[24] Y. Yakimovsky, "Boundary and object detection in real world images," *J. ACM*, vol. 23, no. 4, pp. 599–618, 1976.

[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.

[26] R. Veerapaneni et al., "Entity abstraction in visual model-based reinforcement learning," in *Proc. Conf. Robot Learn.*, 2020, pp. 1439–1456.

[27] D. Pathak et al., "Learning instance segmentation by interaction," in *Proc. Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 2042–2045.

[28] K. Greff et al., "Multi-object representation learning with iterative variational inference," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2424–2433.

[29] N. Watters, L. Matthey, M. Bosnjak, C. P. Burgess, and A. Lerchner, "Cobra: Data-efficient model-based RL through unsupervised object discovery and curiosity-driven exploration," 2019, *arXiv:1905.09275*.

[30] H. Koppula, A. Anand, T. Joachims, and A. Saxena, "Semantic labeling of 3D point clouds for indoor scenes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 244–252.

[31] A. Karpathy, S. Miller, and L. Fei-Fei, "Object discovery in 3D scenes via shape analysis," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 2088–2095.

[32] R. Bajcsy, "Active perception," *Proc. IEEE*, vol. 76, no. 8, pp. 966–1005, Aug. 1988.

[33] M. Bjorkman and D. Kragic, "Active 3D scene segmentation and detection of unknown objects," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2010, pp. 3114–3120.

[34] L. Chang, J. Smith, and D. Fox, "Interactive singulation of objects from a pile," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 3875–3882.

[35] A. Saxena, J. Driemeyer, J. Kearns, and A. Ng, "Robotic grasping of novel objects," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1209–1216.

[36] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," in *Proc. Robo.: Sci. Syst.*, 2013.

[37] V. Satish, J. Mahler, and K. Goldberg, "On-policy dataset synthesis for learning robot grasping policies using fully convolutional deep networks," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1357–1364, Apr. 2019.

[38] Y. Qin, R. Chen, H. Zhu, M. Song, J. Xu, and H. Su, "S4G: Amodal single-view single-shot se (3) grasp detection in cluttered scenes," in *Proc. Conf. Robot Learn.*, 2020, pp. 53–65.

[39] J. Mahler et al., "Learning ambidextrous robot grasping policies," *Sci. Robot.*, vol. 4, no. 26, 2019, Art. no. eaau4984.

[40] M. Dogar and S. Srinivasa, "A planning framework for non-prehensile manipulation under clutter and uncertainty," *Auton. Robots*, vol. 33, no. 3, pp. 217–236, 2012.

[41] K. Hsiao, L. P. Kaelbling, and T. Lozano-Perez, "Grasping POMDPs," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 4685–4692.

[42] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Robust grasping under object pose uncertainty," *Auton. Robots*, vol. 31, no. 2/3, pp. 253–268, 2011.

[43] J. Laaksonen, E. Nikandrova, and V. Kyrki, "Probabilistic sensor-based grasping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 2019–2026.

[44] E. Nikandrova, J. Laaksonen, and V. Kyrki, "Towards informative sensor-based grasp planning," *Robot. Auton. Syst.*, vol. 62, no. 3, pp. 340–354, 2014.

[45] P. Monso, G. Alenya, and C. Torras, "POMDP approach to robotized clothes separation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 1324–1329.

[46] J. K. Li, D. Hsu, and W. S. Lee, "Act to see and see to act: POMDP planning for objects search in clutter," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 5701–5707.

[47] Y. Xiao, S. Katt, A. ten Pas, S. Chen, and C. Amato, "Online planning for target object search in clutter under partial observability," in *Proc. Int. Conf. Robot. Autom.*, 2019, pp. 8241–8247.

[48] D. Fischinger, M. Vincze, and Y. Jiang, "Learning grasps for unknown objects in cluttered scenes," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 609–616.

[49] S. Koenig and R. Simmons, *Xavier: A. Robot Navigation Architecture Based on Partially Observable Markov Decision Process Models*. Cambridge, MA, USA: MIT Press, 1998, pp. 91–122.

[50] H. Bai, S. Cai, N. Ye, D. Hsu, and W. S. Lee, "Intention-aware online POMDP planning for autonomous driving in a crowd," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 454–460.

[51] T. Taha, J. V. Miró, and G. Dissanayake, "A POMDP framework for modelling human interaction with assistive robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 544–549.

[52] J. Pajarinen and V. Kyrki, "Robotic manipulation of multiple objects as a POMDP," *Artif. Intell.*, vol. 247, pp. 213–228, 2017.

[53] J. Pajarinen, "Technical report: The policy graph improvement algorithm," Aalto Univ., Espoo, Finland, Tech. Rep.1, 2020. [Online]. Available: https://arxiv.org/abs/2009.02164

[54] D. Silver and J. Veness, "Monte-carlo planning in large POMDPs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 2164–2172.

[55] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "Despot: Online POMDP planning with regularization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 1772–1780.

[56] G. Casella and E. I. George, "Explaining the Gibbs sampler," *Amer. Statistician*, vol. 46, no. 3, pp. 167–174, 1992.

[57] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[58] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times*. Providence, RI, USA: American Mathematical Society, 2009.

[59] G. O. Roberts and A. F. Smith, "Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms," *Stochastic Processes Their Appl.*, vol. 49, no. 2, pp. 207–216, 1994.

[60] J. G. Propp and D. B. Wilson, "Exact sampling with coupled Markov chains and applications to statistical mechanics," *Random Struct. Algorithms*, vol. 9, no. 1/2, pp. 223–252, 1996.

[61] O. Häggström and K. Nelander, "Exact sampling from anti-monotone systems," *Statistica Neerlandica*, vol. 52, no. 3, pp. 360–380, 1998.

[62] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*. Boca Raton, FL, USA: CRC Press, 2013.

[63] H. B. Mann and D. R. Whitney, “On a test of whether one of two random variables is stochastically larger than the other,” *Ann. Math. Statist.*, vol. 18, no. 1, pp. 50–60, 1947.

[64] M. Danielczuk, A. Angelova, V. Vanhoucke, and K. Goldberg, “X-ray: Mechanical search for an occluded object by minimizing support of learned occupancy distributions,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2020, pp. 9577–9584.

[65] J. Lundell, F. Verdoja, and V. Kyrki, “Robust grasp planning over uncertain shape completions,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 1526–1532.

Joni Pajarinen received the doctoral degree in computer science from Aalto University, Helsinki, Finland, in 2013. He is currently an Assistant Professor since 2020 with Aalto University, where he is leading the Aalto Robot Learning research group that develops new decision making and control methods and applies them to novel robotic tasks. From 2013 to 2016, he was a Postdoctoral Researcher focusing on robotics with Aalto University. Starting from 2016, he has been a Postdoctoral Researcher/Research Group Leader with the Intelligent Autonomous Systems institute, Technische Universität Darmstadt, Darmstadt, Germany. In 2019–2020, he was an Assistant Professor with Tampere University, Tampere, Finland. His research interests include reinforcement learning, robotics, planning under uncertainty, multiagent decision making, and mobile manipulation.

Dr. Pajarinen was an Associate Editor for the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) in 2017, 2020, 2021, and 2022 and for the Conference on Robot Learning (CoRL) in 2018, 2021, and 2022.

Jens Lundell received the B.Sc. degree in automation and systems technology and the M.Sc. degree in space science and robotics with Aalto University, Helsinki, Finland, in 2013 and 2016, respectively, and the doctoral degree in electrical engineering and automation from Aalto University, Espoo, Finland, in 2022.

He is currently a Postdoctoral Researcher with the Royal Institute of Technology, Stockholm, Sweden. His primary research interests include robotic manipulation, robot learning, and grasp planning.

Ville Kyrki (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in computer science from the Lappeenranta University of Technology, Lappeenranta, Finland, in 1999 and 2002, respectively.

In 2003–2004, he was a Postdoctoral Fellow with the Royal Institute of Technology, Stockholm, Sweden, after which he returned to the Lappeenranta University of Technology, holding various positions in 2003–2009. During 2009–2012, he was a Professor of Computer Science with the Lappeenranta University of Technology. Since 2012, he has been an Associate Professor of Intelligent Mobile Machines with Aalto University, Helsinki, Finland. His primary research interests include robotic perception, decision making, and learning.

Dr. Kyrki is a Fellow of Academy of Engineering Sciences (Finland), and a member of Finnish Robotics Society and Finnish Society of Automation. He was a Chair and Vice Chair of the IEEE Finland Section Jt. Chapter of CS, RA, and SMC Societies in 2012–2015 and 2015–2016, respectively, Treasurer of IEEE Finland Section in 2012–2013, and Co-Chair of IEEE RAS TC in Computer and Robot Vision in 2009–2013. He was an Associate Editor for the IEEE TRANSACTIONS ON ROBOTICS in 2014–2017.