Manipulation-based Active Search for Occluded Objects

Lawson L.S. Wong, Leslie Pack Kaelbling, and Tomás Lozano-Pérez

Abstract— **Object search is an integral part of daily life, and in the quest for competent mobile manipulation robots it is an unavoidable problem. Previous approaches focus on cases where objects are in unknown rooms but lying out in the open, which transforms object search into active visual search. However, in real life, objects may be in the back of cupboards occluded by other objects, instead of conveniently on a table by themselves. Extending search to occluded objects requires a more precise model and tighter integration with manipulation. We present a novel generative model for representing container contents by using object co-occurrence information and spatial constraints. Given a target object, a planner uses the model to guide an agent to explore containers where the target is likely, potentially needing to move occluding objects to enable further perception. We demonstrate the model on simulated domains and a detailed simulation involving a PR2 robot.**

I. Introduction

Consider searching for a large mixing bowl in the kitchen when preparing a meal. There are three cupboard shelves that have been partially viewed, two containing stacks of plates, the third some dish detergent. Other objects appear to be in the back, but they are occluded by the plates and detergent, and we need to remove some objects in front to continue searching the shelves. Which object should we move?

Even though the bowl has not been observed yet, it is intuitive to us to keep looking on a shelf with plates because they have a closer function to bowls. Suppose we further observe that of the two shelves with plates, one shelf is small, the other large. Assuming that a bowl, if present, is equally likely to be anywhere in the cupboard, and that both regions incur the same exploration cost, then the large region is more desirable to look at since more objects can be expected to be found there. Moreover, given that mixing bowls are usually large, we may even determine that the bowl cannot fit on the small shelf and eliminate it from consideration.

The above example illustrates two aspects of object search that we wish to capture in our model. First, certain categories of objects, such as plates and bowls, tend to co-occur with each other. Sources of such prior information have been actively explored in previous work (e.g., [1], [2], [3], [4]). However, since our focus is not on the *acquisition* of the prior, but is instead on the orthogonal task of Bayesian *posterior inference* given both prior and observations, we will simply rely on empirically-observed co-occurrences as a prior indicator of similarity. This can be enriched with previously developed priors if necessary. Second, in the latter half of the example, we use geometric information to choose between unobserved spaces. Here we understood that objects have physical extent, and knew that containers have limited capacity into which all objects must fit. In particular, the target query object for which we are searching must fit in the remaining unseen container space. Our model reasons about both object type co-occurrences and spatial constraints.

This work was supported in part by the NSF under Grant No. 1117325. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. We also gratefully acknowledge support from ONR MURI grant N00014-09-1-1051, from AFOSR grant FA2386-10-1-4135 and from the Singapore Ministry of Education under a grant to the Singapore-MIT International Design Center.

Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 {lsw,lpk,tlp}@csail.mit.edu

II. Related work

Significant progress has been made in the context of recognizing and manipulating known objects, but this is traditionally limited to cases where the agent knows the initial positions of the objects. Ye and Tsotsos [5] first relaxed this assumption by formulating object search as an active vision problem, where an efficient trajectory of camera views that localizes the target object is sought. Following this line of work, Sjöö et al. [6] and Aydemir et al. [2] recently considered a similar problem and used spatial relations between objects to more efficiently pinpoint good views of the target object. This was extended to include object-location co-occurrence information ([7], [8]), reflecting, for example, that `cereal` is typically found in the `kitchen`.

The idea of using object contextual information has been repeatedly identified as an important facet of object search. Prior to any work in active object search, Wixson and Ballard [9] recognized that searching indirectly for 'intermediate' objects can ultimately make visual search more efficient, as some objects related to the target may be easier or more reliable to detect. More recently, Kollar and Roy [1] showed that object-object and object-location co-occurrence statistics can be used to predict target object locations and decrease expected plan length. Schuster et al. [4] also explored object-object similarity from a semantic perspective, showing that a similarity measure in ontology space is informative in predicting likely storage locations of objects.

Applied to object search, Kollar and Roy [1] used co-occurrence statistics gathered from the Web to train a Markov random field model indicating how likely objects are at given locations. Extensions to this approach involving information retrieval ([10]) and hierarchical spatial models ([11]) have also been explored. Kunze et al. [12] applied the semantic similarity measure mentioned above for object search by using a Web-trained ontology. Finally, Joho et al. [3] illustrated one way of combining both types of information by extracting features to train a reactive search heuristics.

All previous work addresses only situations in which target objects are lying out in the open. Success in these scenarios relies mostly on identifying rooms that likely contain the object (by place classification), and placing the camera at appropriate viewpoints to identify objects. In other words, the *object search* problem is reduced to *active visual search*, perhaps explaining the popularity of this approach in previous work. In real life, however, such convenient scenarios do not always occur. Objects are often stored in cupboards and drawers, possibly in the back, behind layers of other objects. Occluding objects in the front typically need to be moved away to enable further perception and eventual discovery of such occluded objects. To our knowledge, no previous work has attempted to directly tackle this problem.

In this work, our goal is to focus on *occluded* objects inside containers such as cupboards with known dimensions. These objects are occluded in such a way that no possible viewcone (from the outside) can directly perceive the object. The occluding objects must therefore be moved away first, so manipulation is inherently necessary. Since manipulation is still a relatively expensive and error-prone action on mobile robots, we want to minimize the number of such actions. In particular, we want to model container contents more precisely, taking all object observations and container spatial constraints into account, so that a planner can make manipulation decisions using more accurate estimates. As mentioned earlier, to accomplish this we must perform careful Bayesian posterior inference to reason about potential object types within the remaining unobserved space of containers, given the objects and space already observed in each container.

Following the motivating example given in the beginning, our container contents model will have two major components. First, to model object-object type similarity, we introduce the notion of a container's *composition*, a latent distribution over object types, with a prior based on co-occurrence statistics to enforce the known type similarities. Second, we enforce container spatial constraints by specifying a generative model for putting objects into containers, and then using it to sample contents of unobserved container regions. This generative process results in samples of container contents and configurations, which can be used to answer our fundamental query of object search: how likely is the target object to be found in a certain container?

III. A Generative Model of Container Contents

We now formalize the problem and list some assumptions. The domain is partitioned into a finite set of disjoint containers $\{c_l\}$, each with known location and geometry. The domain is also partitioned into observed and unobserved regions, such that each container may be unseen, fully explored, or partially seen. The known regions are persistent (we cannot 'unsee' a part), and are assumed to be static apart from explicit manipulation by the single agent in the environment. The contents of containers are independent from each other. We treat each container as a homogeneous region of space that we think consists of roughly similar content, so containers serve as a fundamental entity in the object search problem. On the object level, each object in the domain belongs to one member of a finite universe of types $\{t_i\}$. An object must also always be in exactly one container; it cannot belong to multiple containers. It is not essential for the general approach, but for simplicity, we assume that objects are recognizable, so that given a sufficiently clear view, an object's type can be resolved without error.

A query object type is given to the agent as input for the search problem. The agent can move around in the domain and collect information by moving objects and observing (converting unknown regions to known regions), yielding observations of objects and their types. Since objects are often occluded, the agent may need to remove objects in order to allow further observation. Updating the object type distributions based on observed objects and known regions changes the agent's belief about where the query object type is likely to be. The updated belief can be used by a planner to determine the agent's next actions in searching for the query object. One essential piece of information for planning is the probability that an object type is in the unseen region of a particular container. Since containers are modeled independently, we focus on a single container c for the remainder of this section. Suppose we have already observed a set of objects $\{o_j\}$ with corresponding object types $\{t_{o_j}\}$. If we are interested in finding an object of the query type q, then we would like to compute $\mathbb{P}(q \text{ in } c \,|\, \{t_{o_j}\})$, the probability that there is at least one object of type q in container c given the observed object types.

To compute this probability, we make use of the notion of the *composition* of a container. This is the normalized vector of object type counts in a container. Hence if there are T different object types, the composition θ belongs to the $(T-1)$-dimensional simplex $\Delta^{(T-1)}$. The composition and total object count of a container are together a sufficient statistic of its contents, since we can use them to reconstruct the container's empirical object type counts, and we will only reason about container contents using the types of objects they contain. Although this appears to be a trivial algebraic manipulation, the resulting entities simplify the problem because we can now construct a generative model of the container's contents and geometry. In particular, we view the composition as a discrete distribution over object types, and the container's object types (both observed and unobserved) as independent random samples from this distribution. The contents then induce certain geometric characteristics (e.g., occupied volume), which are subject to observed spatial constraints (e.g., remaining volume).

This generative view is summarized in Fig. 1. The shaded nodes correspond to observed object types (multiple objects are represented by a single plate) and the unobserved space in the container. The remaining unobserved space is represented by an octree that is updated after every perception action. The observed object types $\{t_{o_j}\}$ provide information about the latent θ, in turn influencing the distribution on unobserved object types $\{t_i\}$. The nodes left of θ represent the prior on compositions, which enforces the object type similarity structure. The prior will be discussed in section III-A. The

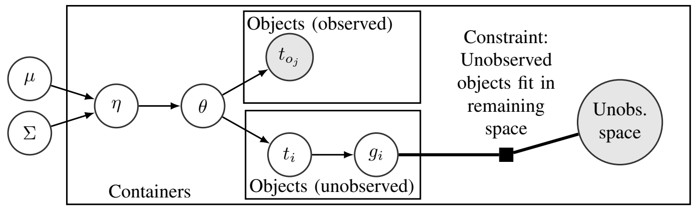


Fig. 1. Graphical representation of our probabilistic model over container contents. Object types t are drawn independently from the composition θ. The prior on θ enforces object type similarities (see section III-A). During object search, parts of a container have been observed, with object types $\{t_{o_j}\}$ found. Unobserved objects with types $\{t_i\}$ may exist in the unobserved space; if so, they must all fit within. This spatial constraint is represented as a factor (see section III-B). See text in section III for details.

downstream factor (solid black square) represents a hard spatial constraint, which is an identity function over whether the object types' geometries satisfy the constraint. That is, it returns 1 if the unobserved object types $\{t_i\}$ fit in the unobserved container space (given the arrangement of observed objects), and 0 otherwise. The spatial constraint will be discussed in section III-B.

The generative model yields a distribution over the multisets of unobserved object types $\{t_i\}$ in a container's unobserved region. Unobserved types that are similar to the observed ones are more likely, and the number of objects present is limited by the size and shape of the unobserved space. We seek the probability that our target query type q is one of these t_i, i.e., that it may be found in the container.

A. *Modeling co-occurrences with logistic-normal priors*

We represent co-occurrence structure in our model by imposing a prior on the composition θ, in the form of a distribution on the simplex. After observing types $\{t_{o_j}\}$ in a container, we perform Bayesian inference to obtain the posterior $\mathbb{P}(\theta \mid \{t_{o_j}\})$ that is used to reason about the unobserved object types $\{t_i\}$. The use of an appropriate prior ensures that the posterior will respect object type similarities.

An immediate candidate distribution on the simplex is the Dirichlet distribution, which essentially tracks pseudo-counts of observed types. However, the Dirichlet distribution can only keep track of individual type counts and not co-occurrences. The covariances between type proportions in θ are also fixed by the pseudo-counts in the prior. The rigidity of this distribution is unsurprising, since it only has a linear number of degrees of freedom, whereas to model general co-occurrence counts, a quadratic number is necessary.

A natural representation of co-occurrences is given by the multivariate normal distribution, where the covariance matrix explicitly encodes such correlations. However, directly generating θ from a normal distribution will violate the simplex positivity and normalization constraints. Instead, we apply a logistic transformation $\sigma(\cdot)$ on some η generated from the normal distribution. This transforms $\eta \in \mathbb{R}^T$ to $\theta \in \mathbb{R}_+^T$ and normalizes θ.[1] The resulting distribution induced on θ is known as the logistic-normal distribution ([13], [14]).

[1]In practice, due to the normalization constraint on θ, θ has one fewer degree of freedom, so it is common to model η such that the first $(T-1)$-element sub-vector is generated from a $(T-1)$-dimensional multivariate normal distribution, and the final element is explicitly set to 0.

The generative model for object types in a container is:

$$\eta \sim \mathcal{N}(\mu, \Sigma) \qquad \mu \in \mathbb{R}^T, \quad \Sigma \in \mathbb{R}^{T \times T}, \Sigma \succ \mathbf{0} \qquad (1)$$

$$\theta_i = \sigma(\eta) = \frac{e^{\eta_i}}{\sum_{k=1}^{T} e^{\eta_k}} \qquad 1 \le i \le T \qquad (2)$$

$$\mathbb{P}(\text{type} = t_i \mid \theta) = \theta_i \qquad 1 \le i \le T \qquad (3)$$

This additional generative component is shown in Fig. 1.

However, the increased representational power comes at a computational price in filtering. In particular, the posterior on η (and θ) is non-conjugate:

$$\mathbb{P}(\eta \mid t_i) \propto \mathbb{P}(t_i \mid \eta)\, \mathbb{P}(\eta) = \frac{e^{\eta_i}}{\sum_{k=1}^{T} e^{\eta_k}} \mathcal{N}(\eta; \mu, \Sigma) \qquad (4)$$

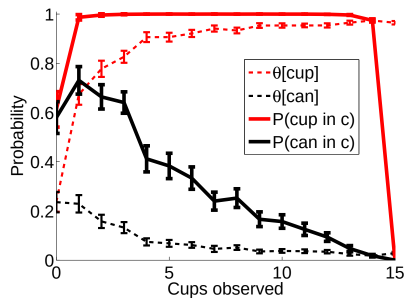
The posterior distribution of η, θ cannot be directly extracted from this form. Instead, we resort to sampling η from the posterior distribution, and apply the logistic transformation on these samples to obtain posterior samples of $\mathbb{P}(\theta \mid \{t_{o_j}\})$.

Details on the Markov chain Monte Carlo sampler for posterior inference are in Hoff [14]. In brief, a Laplace approximation is used to fit a normal distribution to the conditional posterior distribution of η, given the observations so far $\{t_{o_j}\}$. This can be approximately optimized with several iterations of Newton's method. The normal approximation is used as the proposal distribution for posterior η samples, and sample acceptance is determined by the standard Metropolis-Hastings procedure. The logistic transform is applied to accepted samples (after discarding burn-in), giving posterior θ samples that induce a distribution over the unobserved $\{t_i\}$.

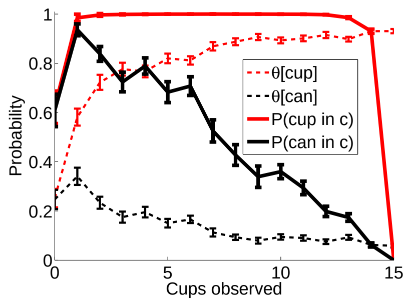
The prior hyperparameters μ, Σ are estimated by maximum likelihood. In particular, training data consists of containers, whose multiset of object types within has been observed. After normalization, this gives empirical observations of θ (and η after an inverse transform). Maximum likelihood estimation of normal distribution parameters from the empirical η then gives the desired μ, Σ. As a robot explores new domains, emptying out containers to find θ can provide domain-specific training data. Because the prior is parametric, it can be instantiated with little *a priori* data; as more training data is available through domain exploration, the prior will become stronger and performance of the model will improve. Alternatively, training data might be acquired from the Web, as has been done in previous work (e.g., [10]).

B. *Enforcing spatial constraints by sampling configurations*

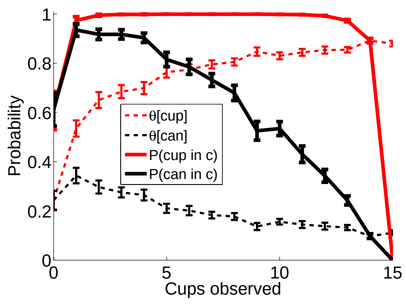
So far we have viewed a container as a partially observed multiset of object types, where the likelihood of the multiset depends on how much it conforms to the type similarity structure. However, since container capacity is limited, most multisets with many objects are unlikely to have a feasible configuration that fits in the container. The spatial arrangements of objects within a container, e.g., their spacing, depends on the process that puts them there. Given a model of that process from which samples can be drawn, we can approximate this process by attempting to pack geometric realizations of the object types into the known container space, and return 1 for the spatial constraint factor if the packing is successful, and 0 otherwise.



(a) Weak prior (5 training examples)



(b) Moderate prior (20 training examples)



(c) Strong prior (100 training examples)

Fig. 2. Demonstration of generative model where cups and cans are known to be similar object types. The progression of posterior θ (no spatial constraint; shown with dashed lines) and probabilities of containing cups and cans (with spatial constraints; shown with solid lines) is shown, as cups are revealed one by one from a container that only contained cups. Probabilities for cup is in red, can is in black. See text in section III-C for details.

Algorithm 1 Sampling algorithm.

```
Input: Obs. types $\{t_{o_j}\}$; unobs. space in $c$; prior params. $\mu, \Sigma$
Output: Estimate of $\mathbb{P}(q \text{ in } c \,|\, \{t_{o_j}\})$ based on sampling
 1: Sample $\eta$ from posterior $\mathbb{P}(\theta \,|\, \{t_{o_j}\})$ (see Section III-A)
 2: count ← 0
 3: for all $\eta$ samples do
 4:     $\theta \leftarrow \sigma(\eta)$, contents ← ⟨⟩, remain_space ← unobs_space
 5:     loop
 6:         Sample type $t$ from $\theta$
 7:         if not CanPack(Mesh($t$), remain_space) then break
 8:         Append $t$ to contents
 9:         remain_space ← Pack(Mesh($t$), remain_space)
10:     if type $q \in$ contents then count ← count + 1
11: return count / num. samples
```

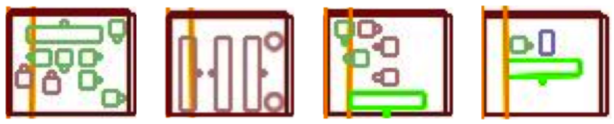


Fig. 3. Container contents are generated by sampling object types one by one from θ, placing each type's template mesh into the remaining space, and terminating when a mesh cannot fit. The two left images show a top view of sample arrangements in an unseen container. For the two right ones, the big bright green object has been observed; only space behind it is unobserved, so fewer things fit and the probability of finding the target decreases.

This generative process is detailed in the bottom right segment of the graphical model in Fig. 1. Object types thought to be in the remaining unobserved container space are generated according to the posterior composition θ. Using template meshes g_i for each posited object type t_i, a greedy packing strategy fits all the meshes into the unobserved space, shown from a bird's-eye view in Fig. 3.

Reasoning exactly about all object type multisets is infeasible. We make inference efficient by using an aggressive sampling strategy, shown in Algorithm 1. Since the posterior on θ is already in the form of samples, we generate an object configuration for each θ sample and return the proportion of configurations that contain the query type q as our estimate of $\mathbb{P}(q \text{ in } c \,|\, \{t_{o_j}\})$. For each θ sample, we draw object types according to θ (line 8) and pack their meshes into the unobserved space as described above (line 11). This continues until the first time a mesh cannot fit in the unobserved space (line 9; the process is not continued, to avoid biasing towards small objects). At this point, one sample configuration of the container's unobserved space is obtained, and whether q exists in this sample can be easily determined (line 13).

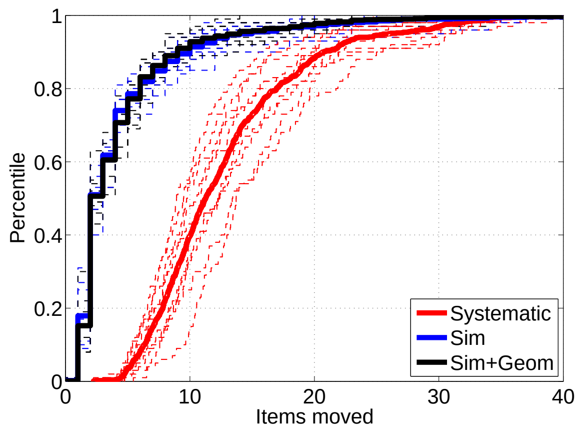
C. *Demonstration of model*

We illustrate some of the basic characteristics of the model with a simple demonstration. Consider a small universe of object types which contains cups, cans, and 2 other irrelevant types. Cups and cans have the same fixed volume. Assume that some containers have been observed for training, where it is found that cups and cans are always either both present or both absent, and in the former case that they occur with similar frequency. Now suppose that we have an unobserved container that contains 15 cups whose capacity is also 15 cups (or cans). As cups are observed one by one, the posterior composition and capacity of the container changes, causing the posterior probability of there being a can in the container to change as well. These probabilities are shown in Fig. 2.

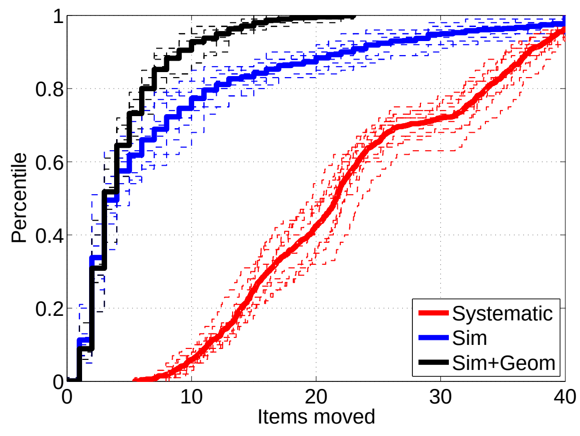
The progression of posterior probabilities shows a number of interesting properties. First, even without observing a single can, the probability of a can being present increased significantly after observing a cup when using moderate and strong priors. This is caused by the type co-occurrence prior increasing θ_{can} after observing cups, since they tended to co-occur in training examples. Second, although the posterior composition of cans is not high, the probability that a can is present is initially much greater because of the large amount of remaining capacity. Third, as more cups are observed, the current container's composition deviates from the training samples. The posterior θ_{can} and $\mathbb{P}(\text{can in } c)$ therefore decrease, at a rate that depends on the prior strength (number of containers in the training set). Finally, as the remaining capacity reaches 0, $\mathbb{P}(\text{can in } c)$ becomes 0 as well.

For comparison, the probabilities for cups are also shown (in red). θ_{cup} changes similarly to that of cans, except in the opposite direction as cups are observed. $\mathbb{P}(\text{cup in } c)$ refers to the probability of cups being present in the unobserved portion of the container (otherwise it would always be 1 after the the first observation). Since θ_{cup} is large, the probability of a cup being present in the remaining space stays close to 1, until the capacity constraint becomes relevant.

The above reasoning would not be possible in a simple non-hierarchical model of object type correlation (e.g., [1]). Although initially seeing a cup made the presence of a can more likely, we were able to update our belief about the make-up of *this particular container*, eventually coming to believe that it was likely to contain only cups.



(a) Searching for a small object



(b) Searching for a large object

Fig. 4. Comparison between three different searching strategies in simulation over 1000 randomly-generated domains. Plots show, for each search strategy, the percentage of simulation trials that found the target object within the number of moves on the horizontal axis. The dashed lines around each solid line show cumulative statistics on 10 bootstrap-resampled datasets, roughly indicating the variance of the cumulative plots. See text in section IV for details.

IV. Experiments

To test our approach, we implemented several object search strategies in simulation. The universe of 9 object types and 10 containers were fixed, along with their default shapes and sizes. 9 example containers were constructed, and the empirical proportions of object type counts in these were used as training examples of θ, from which the maximum likelihood prior hyperparameters μ, Σ were obtained. In each of 1000 simulation trials, the true contents of each container were generated similar to the process in Fig. 1. The contents were all unknown to the agent initially. The visible front layer of each container was returned when views of containers were taken; occluded objects could only be seen after objects in front were removed. A target object type q was also selected for searching. Then, three strategies were tested:

- **Systematic**: Containers were chosen at random and emptied until the target object was found.
- **Sim**: Only object type similarity information was used, i.e., $\mathbb{P}(q \text{ in } c \,|\, \{t_{o_j}\})$ was assumed to be $\mathbb{P}(\theta \,|\, \{t_{o_j}\})$, the posterior θ obtained from the observations and the logistic-normal prior of section III-A.
- **Sim+Geom**: Both object type similarity and spatial constraints were used; $\mathbb{P}(q \text{ in } c \,|\, \{t_{o_j}\})$ is determined using the model of Fig. 1 and algorithm 1.

After each observation, θ and $\mathbb{P}(q \text{ in } c \,|\, \{t_{o_j}\})$ are updated, and the container with the highest probability of containing q is searched (front items moved to reveal occluded objects). For the first two strategies, only non-empty containers were allowed for further search; for **Sim+Geom** this check is in-built due to consideration of unobserved space.

The results of simulating these three strategies are shown in Fig. 4. We measure search performance in terms of the number of items that need to be moved to reveal the occluded target object because (for us) manipulation costs dominate motion costs. The figure shows a cumulative plot of the items moved per search strategy over 1000 trials. For example, for **Systematic** (red) in Fig. 4(a), the median number of items moved is 12, and 90% of simulation trials moved 20 items or fewer. In general, the further left the line is, the better.

Fig. 4(a) shows performance when searching for a small object. Unsurprisingly, strategies using object co-occurrence information are superior, verifying previous findings ([1], [4]). However, reasoning about spatial constraints does not provide any improvements in this case; the lines in the plot for **Sim** (blue) and **Sim+Geom** (black) overlap each other. When searching for larger objects, the benefit of spatial reasoning is more significant; the **Sim+Geom** line in Fig. 4(b) clearly dominates the other two strategies. Because large objects were less prevalent in our domains, the **Systematic** and **Sim** lines have generally shifted lower and to the right. However, **Sim+Geom** is essentially unaffected.

The reasons for these performance gains can be seen more clearly in Fig. 5, which shows the 'manipulation footprint' in a typical simulation trial when searching for a large object. The containers that items were removed from are shown (by color) from left to right in the order of removal, until the large object was found in the crimson container. First, we see again that **Systematic** visits many irrelevant containers (5 in expectation) compared to the other two strategies, wasting moves to empty them. Second, although both **Sim** and **Sim+Geom** started in the same container (dark blue), the latter strategy quickly realized that the large object cannot fit once front items were removed. Finally, **Sim+Geom** visits one fewer container than **Sim** (purple), because that container has empty space in the front, which made it less likely to contain large objects. The additional spatial constraint allows **Sim+Geom** to rule out unlikely containers faster than **Sim**, just as both **Sim** and **Sim+Geom** are faster than **Systematic** because they use prior co-occurrence information.

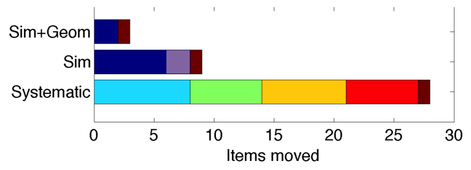


Fig. 5. Comparison between search strategies' 'manipulation footprints' in one simulation run. Different colors represent different containers (of 10 total) in the domain. More sophisticated strategies are able to pinpoint more relevant containers (and not move items in irrelevant ones), and can identify more quickly when a container is unlikely to contain the target object.

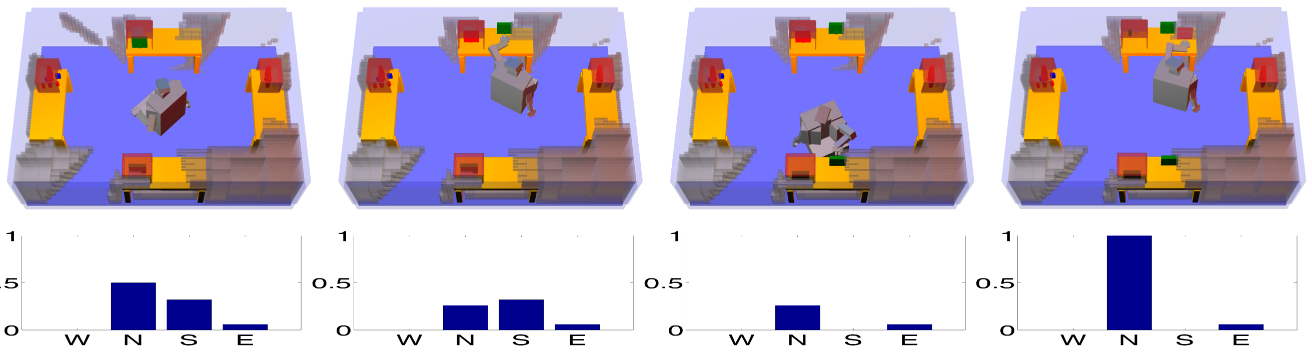


Fig. 6. Simulation trace of a PR2 robot performing object search. See text below for details; the full simulation can be found in the supplementary video.

We have applied our search strategy for a mobile manipulator modeled on a Willow Garage PR2 robot. As shown in Fig. 7, the robot is in an environment with 4 cupboards. Each cupboard has high sides and movable objects in the front that occlude the view of the rest of the contents. The robot's goal is to locate the green cup, which in this example is in the back of cupboard **N**. Object type similarities are indicated by color, with green and brown objects tending to co-occur, and similarly for red and blue. The planning framework described in Kaelbling and Lozano-Pérez [15] is used.

Fig. 6 shows snapshots of the search; please see the supplementary video for the full simulation. The top row is the robot's belief state: gray areas show regions not yet viewed by the robot; colored objects show detected objects. The bottom row shows the estimate of $\mathbb{P}(\text{green cup in } c \,|\, \{t_{o_j}\})$ for each container. From left to right:

(a) After seeing the front of each cupboard, object type similarity indicates only **N** and **S** are likely. Also, **N** is more likely because it has more unobserved space.

(b) When exploring **N**, an unexpected red object is observed. Since red objects tend not to co-occur with green objects, the probability in **N** drops, and **S** becomes more likely.

(c) Removing an object from **S** reveals that there is no more space behind the remaining object for a cup, so the probability becomes 0. Approaches that do not reason about spatial constraints (e.g., **Sim**) would remove the remaining green object in **S** as well, since it is likely to co-occur with the target green cup.

(d) **N** is now the most likely container again. Removing the red object reveals the target green cup in the back of **N**.

V. Conclusion

We have presented an approach for manipulation-based searching of occluded objects. A novel probabilistic model of container contents was introduced that reasons about object-object co-occurrences and spatial constraints, both of which were identified as important aspects of object search. Inference using the model was discussed, and our approach was demonstrated in multiple simulated experiments. The utility of object type similarity was verified, and spatial reasoning was beneficial when searching for large objects. Our approach complements previous vision-only approaches, and we hope to integrate these ideas with the ones presented here on our physical robot in the future.

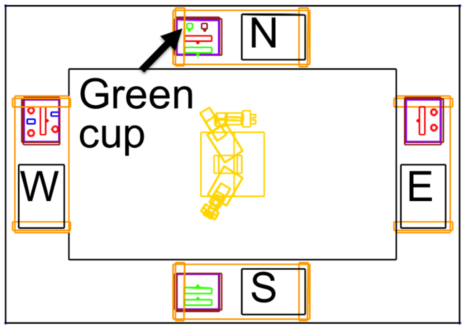


Fig. 7. Bird's-eye view of the initial object configuration in the PR2 robot simulation. The goal is to find the green cup in the back of cupboard **N**.

References

[1] T. Kollar and N. Roy, "Utilizing object-object and object-scene context when planning to find things," in *ICRA*, 2009.

[2] A. Aydemir, K. Sjöö, J. Folkesson, A. Pronobis, and P. Jensfelt, "Search in the real world: Active visual object search based on spatial relations," in *ICRA*, 2011.

[3] D. Joho, M. Senk, and W. Burgard, "Learning search heuristics for finding objects in structured environments," *RAS*, vol. 59, no. 5, pp. 319–328, 2011.

[4] M. Schuster, D. Jain, M. Tenorth, and M. Beetz, "Learning organizational principles in human environments," in *ICRA*, 2012.

[5] Y. Ye and J. K. Tsotsos, "Sensor planning in 3D object search," *CVIU*, vol. 73, pp. 145–168, 1996.

[6] K. Sjöö, D. Gálvez-López, C. Paul, P. Jensfelt, and D. Kragic, "Object search and localization for an indoor mobile robot," *J. Computing and IT*, vol. 17, no. 1, pp. 67–80, 2009.

[7] M. Hanheide, C. Gretton, R. Dearden, N. Hawes, J. L. Wyatt, A. Pronobis, A. Aydemir, M. Göbelbecker, and H. Zender, "Exploiting probabilistic knowledge under uncertain sensing for efficient robot behaviour," in *IJCAI*, 2011.

[8] A. Aydemir, M. Göbelbecker, A. Pronobis, K. Sjöö, and P. Jensfelt, "Plan-based object search and exploration using semantic spatial knowledge in the real world," in *ECMR*, 2011.

[9] L. E. Wixson and D. H. Ballard, "Using intermediate objects to improve the efficiency of visual search," *IJCV*, vol. 12, no. 2–3, pp. 209–230, 1994.

[10] M. Samadi, T. Kollar, and M. M. Veloso, "Using the web to interactively learn to find objects," in *AAAI*, 2012.

[11] P. Viswanathan, D. Meger, T. Southey, J. J. Little, and A. K. Mackworth, "Automated spatial-semantic modeling with applications to place labeling and informed search," in *CRV*, 2009.

[12] L. Kunze, M. Beetz, M. Saito, H. Azuma, K. Okada, and M. Inaba, "Searching objects in large-scale indoor environments: A decision-thereotic approach," in *ICRA*, 2012.

[13] J. Aitchison and S. M. Shen, "Logistic-normal distributions: Some properties and uses," *Biometrika*, vol. 67, no. 2, pp. 261–272, 1980.

[14] P. D. Hoff, "Nonparametric modeling of hierarchically exchangeable data," Dept. of Statistics, U. of Washington, Tech. Rep. 421, 2003.

[15] L. P. Kaelbling and T. Lozano-Pérez, "Unifying perception, estimation and action for mobile manipulation via belief space planning," in *ICRA*, 2012.