2011 IEEE International Conference on Robotics and Automation
Shanghai International Conference Center
May 9-13, 2011, Shanghai, China

Search in the real world: Active visual object search based on spatial relations

A. Aydemir, K. Sjöö, J. Folkesson, A. Pronobis, P. Jensfelt

Abstract— **Objects are integral to a robot's understanding of space. Various tasks such as semantic mapping, pick-and-carry missions or manipulation involve interaction with objects. Previous work in the field largely builds on the assumption that the object in question starts out within the ready sensory reach of the robot. In this work we aim to relax this assumption by providing the means to perform robust and large-scale active visual object search. Presenting spatial relations that describe topological relationships between objects, we then show how to use these to create potential search actions. We introduce a method for efficiently selecting search strategies given probabilities for those relations. Finally we perform experiments to verify the feasibility of our approach.**

I. INTRODUCTION

Service robots – robots that perform everyday tasks in everyday settings, whether domestic, office or other – are an eagerly anticipated goal within autonomous agent research. Compared to industrial robots, progress in service robotics has been relatively slow to date. This discrepancy is largely due to the fact that the environments service robots have to cope with are far more dynamic, unpredictable and "human-oriented" than those encountered by their industrial brethren.

Much work is going into overcoming the problem of making sense of complex environments, especially using vision. Key in this effort is the apprehension of *objects*. Objects hold an important role in human perception of space [1]. Localizing and interacting with them lies at the heart of various robotics research challenges, and while there is no shortage of open questions in dealing with objects, the bulk of previous work relies on the assumption that the particular object in question is already within the sensory reach of the robot. An often stated reason for this is tasks such as object recognition and object manipulation are already challenging enough. Nevertheless, as the field advances in its aim to build versatile service robots, the assumption of objects being readily available in the field of view of robot's sensors is no longer reasonable.

A mobile robot operating in the real world will have to interact with objects of varying size, shape and degree of mobility, to name a few complications. One way around the issue is to let the environment keep track of the objects and report their location when asked; however, this creates a dependency on an intelligent environment. Thus, it is imperative to be able to reliably locate objects visually in the real world in order to perform tasks such as place categorization, fetching and carrying, and manipulation.

The authors are with the Centre for Autonomous Systems at the Royal Institute of Technology (KTH), Stockholm, SE-100 44, Sweden. This work was supported by the SSF through its Centre for Autonomous Systems (CAS), as well as by the EU FP7 project CogX and the Swedish Research Council, contract 621-2006-4520 (K. Sjöö)
[aydemir,krsj,johnf,pronobis,patric]@csc.kth.se

The goal of object search, then, is to produce a set of sensing actions which brings the target object into the sensor's field of view. For efficiency, it should consist of a minimum number of sensing actions with maximal object detection probability. This is an example of active vision [2]. In the context of object search we refer to this as active visual search (AVS).

Considering the case of searching for a 3D object in 3D space, solving the AVS problem is far from trivial. Factors such as occlusion and illumination affect the search outcome significantly; therefore, to construct and execute such a plan, the searcher must actively adjust its sensor parameters to obtain the highest quality data. Search within the context of an agent's current sensory input has been investigated in some detail in [3], [4]. However, the problem of search on a mobile platform, in a real world environment has seen less activity.

Most significantly, the mobile AVS problem has a dimensionality proportional to the number of sensor parameters that can be actively controlled – such as the position and orientation of a sensor, for each action. Uninformed search, i.e. without any prior information on the target object's location, inevitably suffers from the curse of dimensionality. The pioneering work done by Tsotsos [5] showed that the problem of optimal search is NP-hard. [6] provided the probabilistic framework for performing object search which is based on the Bayesian theory. Using the same probabilistic framework [7] performed experiments on a humanoid robots in a real world setting. [8] presented an approach where the robot uses visual cues to further examine a particular part of the search space.

In 1976, Garvey presented *indirect search* as a way of limiting the search space [9]. Indirect search involves first locating an intermediate object in order to facilitate the search for the target object. An example of this is finding the table first and then focusing on top of the table to find a cup in a room. Later on, Wixson [10] showed that indirect search provides a significant increase in search efficiency.

More recently the AVS problem has seen increasing interest. [11] uses spatial relations to guide the search process. [12] presents a system that creates object maps. [13] applies the methodology used in a pursuit-evasion scenario to the AVS problem providing a new insight but with limited experiments. [14] studies the case where a robot simultaneously explores and searches for objects. [15] uses object co-occurrence histograms to locate objects in the environment represented as a SLAM map. [16] focuses on getting the 6DoF pose of the object and uses probability maps to plan the search.

A. Contributions

In this work we consider the case of a mobile robot looking for an object in an indoor environment. Contributions of this work are four-fold. First we provide an application of previously introduced spatial relations in a robotics framework, by basing a strategy for AVS on them. Second, we present several variants of a method for selecting a near-optimal strategy, and compare them in the context of object search. Third, we demonstrate a method for robust execution of a set of strategies, by taking into account failed strategies and re-evaluating possible strategies. Finally, we demonstrate the above ideas by implementing them on a mobile robot.

II. Search in the real world

Imagine a robot tasked with visually locating and fetching a cell phone located somewhere on a floor of a medium sized building. Without any *a priori* information on the object's whereabouts the robot, in the worst case, must cover the entire volume of each of the rooms on that floor. To accomplish this, the robot calculates a search plan which includes a series of sensing actions with the hope that this plan will lead to localizing the said object. Sensors, and in particular cameras, have a limited field of view and a particular object can only be reliably detected within some interval of ranges. Covering the entire environment will be very time consuming and appear quite inefficient to human observers. Therefore a robotic system would greatly benefit from starting its search with some initial information and thereby a non-uniform *a priori* probability distribution function (PDF) defined over the volume of search space.

Assume there is such a PDF defined over the metric space, as is done in [16], [13], [7]. A robot making use of such *a priori* information will perform better than the aforementioned uninformed search, given that its initial PDF is an accurate representation of the real world's state. However this representation of probabilities would suffer from being susceptible to small changes in the environment since no abstraction over the metric space is present. An obvious example would be an object moved from one end of a meeting table to another. Such a system would detect the absence of the object and proceed with a full-fledged search in the entire environment.

Furthermore, several different PDFs for various objects will be harder to maintain and use as the environment grows in size. This has led to experiments in the previous work being done in a very limited search space, as it is computationally expensive to run such a system in larger environments. Finally, a robot interacting with humans is likely to receive information on an abstract level and not on the metric level. Humans describe positions not by exact coordinates but by relations to other entities in the environment. Thus metric level systems will need some mechanism to help them interpret such information. On the other hand, a system that does not take into account lower level aspects might perform poorly through failing to take account of such low level factors as occlusions, limited sensor range, and illumination.

III. Problem Formulation

All of the above points out the necessity of introducing higher level abstraction to the AVS problem, while still meeting the lower-level challenges of a real world scenario. We accomplish this by introducing functional definitions of spatial relations to the AVS problem.

The main issue that this work aims to deal with is: given information – possibly uncertain – about spatial relations between objects in an environment, how is the agent to organize an efficient search aimed at finding a given object?

A. Choosing a next best view

We introduce the next best view selection algorithm following the formulation of [6]. The robot has an initial PDF over the 3D space Ψ. The search region Ψ is discretized by tessellating it into 3D cubes $c_1 ... c_n$. A sensing action s is then defined as taking an image of Ψ and running a recognition algorithm to determine whether the target object o is present in the image or not. In the general case, the parameter set of s consists of camera position (x_c, y_c, z_c), pan-tilt angles (p, t), focal length f and a recognition algorithm a; $s = s(x_c, y_c, z_c, p, t, f, a)$.

An agent starts out with an initial PDF for the target object's location over Ψ. We assume that there is exactly one target object in the environment either inside or outside the search region. Let $p(c_i)$ be the probability of the object's center being in the i^{th} cell.

The next best view selection is then defined as:

$$\underset{j=1..N}{\operatorname{argmax}} \sum_{i=1}^{n} p(c_i) S(c_i, j) \tag{1}$$

Where N is the number of candidate sensing actions and S is defined as:

$$S = \begin{cases} 1, & \text{if } c_i \text{ is covered by the } j^{\text{th}} \text{ sensing action} \\ 0, & \text{otherwise} \end{cases}$$

Finally, the set of candidate view points is determined by randomly sampling the reachable space in Ψ.

B. Search Strategies and strategy steps

We wish to determine the *strategy* that minimizes the expected *cost* to find the target object. A strategy consists of a sequence of steps, each of which is a search procedure in its own right: a "simple" search, looking for an object given some specific prior probability distribution.

For example, a strategy might be composed of the steps

1) Go to room 1
2) Search for the table (which could be anywhere in the room)
3) Search for the box on top of the table
4) Search for the book inside the box

Another strategy might be

1) Go to room 1
2) Search for the box, which could be anywhere in the room but at a height compatible with being on the table
3) Search for the book inside the box

In this case, the robot uses the relational information directly, without the extra step of localizing the table.

The objective, then, is to find the most efficient sequence of steps, out of all sequences that lead to the target object.

Each strategy step, such as "the box on the table" corresponds to a 3D PDF (Figure 2). The step is carried out by greedily generating a set of potential view points, as described in III-A. The process is continued until the object is found, or the remaining probability is lower than a threshold (we set it to 30%), in which case it is deemed that the current strategy step has failed.

The cost of a strategy step is calculated as follows: First a 3D PDF corresponding to a strategy step is calculated. Then the set of next best view points that covers 70% of this PDF is generated in accordance with III-A. Finally the total distance travelled to visit all the view points in the set is calculated. The cost thus is based on the total amount of movement until a certain proportion of the initial probability is covered.

IV. SPATIAL RELATIONS AND SEARCH

Objects in environments that are created and used by human beings do not occur randomly. Rather, people design and organize spaces in ways that serve various functional purposes. This organization is expressible in terms of *spatial relations*.

Spatial relations are abstractions of the configuration in space of objects, such as their distances, directions or topological relationships. These help humans structure and remember aspects of their environment, and are likewise of great potential use when a robot has to search for objects in that same environment.

We make use of two of the most important topological spatial relations: "in" (meaning that an object is contained in the convex hull of another) and "on" (meaning that one object is being physically supported by another). The object that contains or supports is termed the "landmark", while the other is termed the "trajector".

In [11] a detailed computational model for each of these relations is proposed, and a method for computing a probability density, as might be used by an AVS procedure, is presented. These models take the form of functions that take the pose and geometry of two objects and yield a scalar measure of how applicable the relation "on" or "in", respectively, is to the configuration in question. High values mean the trajector's pose corresponds very well to being "on"/"in" the landmark. These functions, when normalized over 3D space, produce probability density functions that can be directly used in AVS.

Given the position of a landmark, this method drastically reduces the search space when looking for a trajector that has a known spatial relation to that landmark. Even when the landmark's position is not known the spatial relation information may help accelerate the search by biasing the distribution.

[11] assumes that the robot has complete knowledge of which relations hold. This is not the typical case, however; rather, what is given will be a probability distribution over possible relations, gleaned from common-sense knowledge databases or learned from experience in real environments. The question then becomes how best to investigate the different possible relational configurations in order to find the sought object at as low a cost as possible.

V. STRATEGY SELECTION

The object search strategy selection is modeled as a Markov decision process, MDP, over the belief state. The target object location is represented by an n-tuple of booleans $\mathbf{s}$. Each element corresponds to a relational description of the object location such as: "book on table in livingroom". We refer to these descriptions as configurations for the object. An element of $\mathbf{s}$ is true if the object has the configuration in question. The configurations are not mutually exclusive. We restrict ourselves to configurations that contain a specific room. The $\mathbf{s}$ is a discrete random variable. Its probability evolves as the robot searches for the object. This probability is the belief state of the MDP.

The actions, a, are specific strategies for searching configurations. A single configuration will always have a direct search strategy which is to search for the object with the *a priori* distribution of object locations dependent on the configuration as a whole. Some configurations will also have indirect search strategies with several steps, as exemplified in Sec. III-B. Each action has a set of possible final states and costs. The state transitions consist of either finding the object or failing at some step. Finding the object changes the belief to certainty and ends the search; failing at some step changes the probabilities of the configurations and the costs of future actions.

The probabilities change by the Bayes update rule:

$$p(\mathbf{s}|z) = p(z|\mathbf{s})p(\mathbf{s})/p(z) \qquad (2)$$

where z is the observation of failing the current step. The probability of successfully finding the object in a step given a specific configuration is the sum of the probability mass covered by all the view cones selected during that step.

Costs of actions are based on an estimate of robot motion needed to carry out the action, including travel to the room and movement between the selected view points. After a failed step, costs for subsequent actions may be decreased, if the search located objects that are intermediate steps in those actions.

The Bellman equation without discount for this system is:

$$V(x) = \max_a (R(x,a) + \sum_{x'} p(x'|x,a)V(x')). \qquad (3)$$

Where x and x' are belief states and V is the value function which here is the negative expected cost. The maximum is taken over all actions (i.e. search strategies) and the sum is over the various possible transitions states x' from x under action a. The optimal action would be the argument of the maximum.

Without a discount on future costs and without any certainty of finding the object, a stopping criterion for the search is required, or the expected cost will be infinite. The choice of stopping criteria will affect the optimal policy. The search terminates when the probability of the object being in a room is below some threshold for every room. One could also use the probability of a configuration or just the posterior probability of the object being in the environment. A stopping criterion at the room level is useful as it allows for exploiting the separation of the search into rooms to make policy computation tractable.

We further simplify the policy calculations by limiting the state transitions to either success or failure at the final step. This means that we need not update the probabilities of the intermediate configurations, e.g., "table in livingroom".

The expected cost for an action selection policy, π is:

$$< cost|\pi >= E_0(\pi) = C_n(\pi) + Q_n(\pi)E_n(\pi) \qquad (4)$$

where $E_n(\pi)$ is the expected cost of the continued search given that the n^{th} policy action failed.

$$Q_n(\pi) = Q_{n-1}(\pi)(1 - p_n(\pi)) \qquad (5)$$

where $p_n(\pi)$ is the probability of success on the n^{th} action given that the previous actions all failed. Q_n is the probability of failing n steps. Then suppressing the dependence on the policy π:

$$C_n = C_{n-1} + Q_{n-1}(p_n c_{sn} + (1 - p_n)c_{fn} \qquad (6)$$

where c_{sn} and c_{fn} are the expected cost of success and failure respectively.

We can now compute the expected cost of any policy if we knew $E_n(\pi)$. It can be approximated based on Q_n. Q_n starts as $Q_0 = 1.0$ and is then reduced as n increases, asymptotically approaching $\bar{Q}$ which is the *a priori* probability that the object is not in the environment. Q_n is therefore a measure of the progress of our search. We will explore two assumptions on E_n: i), it is simply a constant or ii), it is proportional to $(Q_n - \bar{Q})$.

The constant assumption leads to the problem of choosing a specific constant. We use two methods of selecting this constant future cost. First, introducing a parameter $\bar{E}$ chosen based on typical search costs.

$$E_0 = C_n + Q_n\bar{E}. \qquad (7)$$

Second, use Eq.(4) to estimate the constant by setting $E_n = E_0$.

$$E_0 = C_n + Q_n E_0 \rightarrow E_0 = C_n/(1 - Q_n). \qquad (8)$$

Similarly, using the second assumption on E_n we find:

$$E_0 = C_n/(1 - Q_n(Q_n - \bar{Q})/(1 - \bar{Q})), \quad n > 0. \qquad (9)$$

Once an assumption has been made one can find a well-defined optimal action by setting the depth of the policy search, n, or the number of steps to project the MDP. The larger n is the smaller the effect of our approximation; however, the number of belief states grows exponentially with n. This makes simple exhaustive search of all paths impractical for more than a few steps. If the greedy search $n = 1$ is used the search will often be sub-optimal; for example, moving from one room to another carries a large cost and should not be chosen until the current room has been well searched, whereas the cost of returning to the room later is not considered by the greedy search.

Search over multiple rooms is complex. Nevertheless if the search were restricted to one room the sequence of actions would be independent of the state of the search in the other rooms. The magnitude of the problem may thus be greatly reduced by finding the optimal search sequence for each room separately. We then only need to optimize the choice of room at each step, knowing the sequence of actions to take given the room.

Within a room the greedy strategy works well so long as there are no dependencies between the separate configurations. Such independence does not hold in general. In some cases a configuration is made easier to search by having failed on some earlier search; e.g., finding the book on the table after having found the table on a previous failed search. These special configurations can be enumerated. We then do a restricted forward projection of the MDP within a room by choosing at each step to project the lowest expected cost policy from the previous step and the lowest cost policy out of those that contain the special configurations for each of the special configurations. In this way we are not likely to miss an advantage from these dependencies.

During policy evaluation we look at all sequences of room choices out to our search depth, choosing for each room the next action from the list of optimal actions found for that room. We then take the sequence of actions the has the lowest expected cost.

This gives us an efficient way to reduce the search over policies, breaking the problem up first into rooms and then into searches containing a limited number of policies at each step.

VI. Experiments

A. Simulation

We used simulations to verify that our action selection policies could produce lower cost searches on average. The MDP model matches the simulation while the actual implemented object search on the robot is not modeled perfectly in the MDP used for policy selection. Besides not being able to model the uncertainty in object recognition for all viewing angles, we also could not model other relatively random aspects of the real world robot, such as chance recognition of the tables and furniture that might help with later searches.

The simulations were done by varying the *a priori* distribution of the object over configurations. We set these randomly and then computed the action under a policy. We then

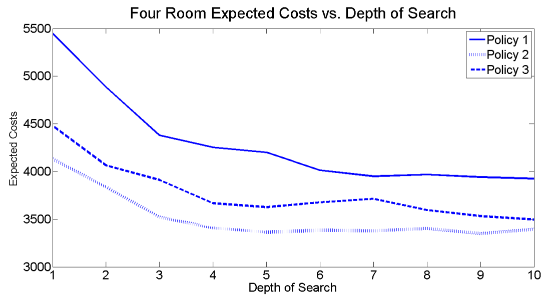


Fig. 1. We show the simulated expected costs for the example of four rooms using the three policies vs. n, the depth of the policy's search over actions.

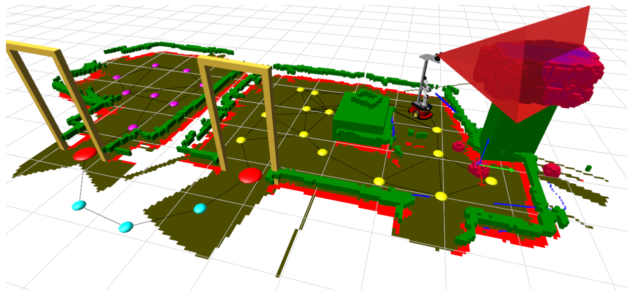


Fig. 2. An example PDF (shown in purple color) that corresponds to the relation "book on bookcase" and selected view point during an actual run. The map containing two rooms is also shown.

found the next action under the policy for the contingency of failure. We continued until the search would have ended due to our stopping criteria. Then we computed the expected cost of executing the entire sequence of actions using Eq. (4) where E_n is zero when we reach the stopping point. All policy variations were checked on this initial distribution. Finally a new random distribution was then selected and the process repeated. We performed 100 distributions and computed the resulting performance for various policies and depth of search. In this way we could see how the three assumptions on E_n, the depth of search, and the size of the environment affect the expected cost of search.

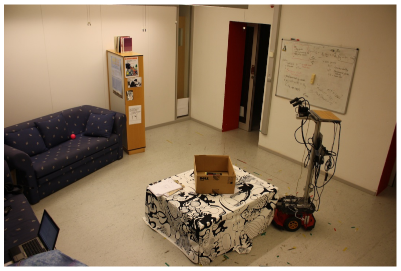
We found that the best expected costs were found by using equation (8). Figure (1) shows the expected costs vs depth of search for our policies 1, 2, and 3 using Eq. (7), Eq. (8) and Eq. (9) respectively. One can see that the cost drops about 20-25% by searching out 3 steps. It then does not change significantly by extending the search to more steps. The standard deviation at these points was about 300-450. This shows that policy 2 is expected to outperform the other two and that there is some gain to looking several steps ahead.

We also compared to exhaustive searching all actions out to three steps. This took on average 450 ms to compute an action and had an expected cost of 3900. This compares to the time 0.4 ms for our policies 1, 2, and 3 which achieved about this same level of cost at 3 steps. Exhaustive searching to two steps took 10 ms and a cost of 4250 vs. about 0.2 ms for policies 1, 2, and 3. So our exploiting the separation into rooms reduced the exponential growth from a factor of 45 to 2 even with only 4 rooms.

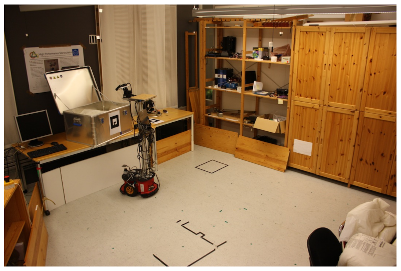
We also looked at much larger environments of up to 10 rooms. For instance for a 10 room environment policy 2 with 5 steps was 12% better than a 1 step policy. For 10 rooms and five steps, policies 1, 2 and 3 all took around 500 ms to compute an action at 5 step depth.

B. Robot experiments

1) Setup: In order to demonstrate its practical workability, we also implemented and tested our approach on a real-world autonomous system. Experiments were carried out on a Pioneer III wheeled robot, equipped with a Hokuyo URG laser scanner, and a camera mounted at 1.4 m above the floor. Experiments took place in two different rooms, connected by a corridor that the robot could traverse whenever it decided to search the other room. A SLAM implementation [17] carried out localization using a previously built map.



(a) Robot in room 1 searching "book in box on table"



(b) Robot in room 2 searching "book in crate"

Fig. 3. The robot used during experiments in two different rooms performing strategies

Room 1 contained the following fixed, identifiable objects: One small and one large bookcase, and one table; room 2 contained another table and a set of shelves. Mobile objects used were a cardboard box, a metal crate and a book (the target object).

The given initial belief state across configurations is presented in Table I. Note that the probabilities do not sum to 1; rather, configurations subsume each other; for example, `book ON table IN room1` contains `book IN box ON table IN room1` as a special case. (In other words, the probability that the book is on the table but *not* in the box is zero.)

2) Selected policy: The policy chosen given the same initial belief state, maps and object geometries is deterministic;

Configuration	Probability
book IN room1	0.55
book ON table1 IN room1	0.05
book ON small_bookcase IN room1	0.30
book IN small_bookcase IN room1	0.05
book IN large_bookcase IN room1	0.05
book IN box ON table1 IN room1	0.05
book IN box IN room1	0.15
book IN room2	0.40
book ON table2 IN room2	0.05
book IN crate IN room2	0.30
book IN shelves IN room2	0.05

TABLE I
INITIAL CONFIGURATION PROBABILITIES USED IN EXPERIMENTS

Go to `room1`
↓
Search for `small_bookcase`
↓
Search for `book ON small_bookcase`
↓
Search for `box`
↓
Search for `book IN box`
↓
Go to `room2`
↓
Search for `crate`
↓
Search for `book IN crate`
↓
Object not found

TABLE II
POLICY GENERATED DURING EXPERIMENTS

for the above state, the policy chosen (assuming the book was never detected) is presented in Table II. The robot performs indirect search on the most likely landmarks in the first room, then moves on to the second room. The search is aborted if it fails all three strategies, the cost-to-probability ratio for the remaining possibilities falling below the threshold.

3) Results – Accurate probabilities: The system was run 20 times, with the target object placed at each configuration a number of times commensurate with the configuration probabilities provided the robot. The results were as follows:

Configuration	Freq.	Success	Avg. views
ON sm_bookcase IN r1	6	6	2.67
IN sm_bookcase IN r1	1	1	5
ON lg_bookcase IN r1	1	0	20
IN box ON table1 IN r1	1	1	7
IN box IN r1	3	2	10
ON table2 IN r2	1	0	22
IN crate IN r2	6	5	11.17
IN shelves IN r2	1	0	19
Overall	20	15	9.6

The results show that the policy selected for the given probabilities performs well, catching most of the configurations whose cost-to-probability ratio are not below the threshold. A different threshold would naturally mean more strategies examined, and thus longer searches, but also fewer failures.

4) Results – Inaccurate probabilities: To confirm that the proposed strategy selection algorithm makes proper use of the probabilities it is given, we also carried out two tests in which the robot was provided different configuration probabilities from those listed above, producing different search policies accordingly. It was then estimated, based on this and the previous runs, how successful and costly those policies would be, given that the reality (i.e. the actual configurations) were the same as originally.

Case 1: By shifting 0.2 worth of probability mass away from "book ON small_bookcase IN room1" to "book IN large_bookcase IN room1", and similarly swapping the probabilities of "book IN crate IN room2" and "book ON table2 IN room2", a policy is generated which tries strategies in this order: First "book IN box IN room1", then "book IN large_bookcase IN room1", and finally "book ON table2 IN room2".

Case 2: Similarly, swapping "book ON small_bookcase IN room1" with "book ON table1 IN room1" and "book IN crate IN room2" with "book ON table2 IN room2" yields strategies in the sequence: "book ON table1 IN room1", "book IN box IN room1" and "book ON table2 IN room2".

When these two policies, based on modified probabilities, are applied to the set of configurations drawn from the probabilities in Table I, (simulated) success rates and view counts worsen considerably:

Run	Appr. avg. views	Appr. success rate
Accurate	9.6	75%
Inaccurate 1	19.7	25%
Inaccurate 2	12.1	20%

These results indicate that the strategy selection algorithm does indeed make proper use of the probabilistic information it is provided.

C. Conclusions

We have presented a method for robust and scalable AVS using spatial relational information. We have introduced the idea of using object-object spatial relations as an abstraction method for AVS. We showed how groupings of spatial relations can be used as search strategies in the context of AVS. Furthermore, we provide a decision theoretic strategy selection method to obtain a near-optimal search behavior and to handle cases where some strategies fail to find the target object. We have finally concluded through real world experiments the feasibility and correctness of our presented ideas.

Using spatial relational information as a way of influencing object search greatly improves both the search efficiency and outcome. However the search performs poorly when the probabilities associated with these strategies do not correspond to the real world.

D. Future Work and Discussion

Directions for future investigation involve making use of dense 3D point cloud representation of scenes to guide the search. The functional aspects of our everyday world mean that 3D structure provides better cues to object locations compared to using only visual appearance. Therefore exploiting shape properties of scenes would be beneficial for a searcher robot.

In the experiments section the qualitative object location probabilities are given to the system manually beforehand and they are not updated based on the search result. Instead of hard-coded probabilities, we would like to give the robot access to a database of spatial knowledge. Abstracting this knowledge makes the the problem tractable and the knowledge representation more robust. This knowledge can be mainly regarded as spatial common sense knowledge that is not environment specific. For instance, the category of a room can be used to build a prior over the objects that are more likely to be found in that room [18]. Such general knowledge can be learned by the robot over the course of its operation, but can also be transferred from humans either directly or by an analysis of annotated databases (e.g. LabelMe [19], ConceptNet [20]) or results gathered using Internet search engines.

It also is important to develop the methods to maintain these probabilities over very long periods of time so that the searcher robot can adapt to its environment. One future direction is designing a probabilistic graphical model representing spatial knowledge at the conceptual level in order to perform inference on object locations using common sense knowledge and various types of information acquired from the robot's sensors. We propose to structure the abstracted spatial knowledge according to a semi-probabilistic ontological representation that combines high level spatial concepts as well as relationships between those concepts and instances of objects and rooms in the environment.

In order to fully represent the statistical dependencies between the random variables expressing the uncertainties captured by the representation, we need a more expressive model such as Bayesian Networks (BN) or Markov Random Fields (MRF).

However, in order to capture the different types of dependencies that exist in the model, as a future research direction we suggest using chain graph models, being a natural generalization of the above. Chain graph models have the advantage over either BNs or MRFs of being able to express both strictly causal relationships as well as symmetric and associative relations, both of which can be identified in representation of spatial knowledge [21].

References

[1] S. Vasudevan, S. Gächter, and R. Siegwart, "Cognitive spatial representations for mobile robots perspectives from a user study," In Proc. ICRA Workshop: Semantic Information in Robotics (ICRA - SIR 2007), Rome, Italy, 2007.

[2] R. Bajcsy, "Active perception vs. passive perception," in *Proc. 3rd Workshop on Computer Vision: Representation and Control*. Washington, DC.: IEEE Press, October 1985, pp. 55–59.

[3] A. Torralba, M. S. Castelhano, A. Oliva, and J. M. Henderson, "Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search," *Psychological Review*, vol. 113, p. 2006, 2006.

[4] M. Björkman and J.-O. Eklundh, "Vision in the real world: Finding, attending and recognizing objects," *International Journal of Imaging Systems and Technology*, vol. 16, pp. 189–208, 2006.

[5] J. K. Tsotsos, "On the relative complexity of active vs. passive visual search," *International Journal of Computer Vision*, vol. 7, no. 2, pp. 127–141, 1992.

[6] Y. Ye, "Sensor planning for object search," Ph.D. dissertation, 1997.

[7] F. Saidi, O. Stasse, and K. Yokoi, "Active visual search by a humanoid robot," *Recent Progress in Robotics: Viable Robotic Service to Human*, vol. 16, pp. 171–184, 2009.

[8] S. Ekvall, D. Kragic, and P. Jensfelt, "Object detection and mapping for service robot tasks," *Robotica: International Journal of Information, Education and Research in Robotics and Artificial Intelligence*, 2007.

[9] T. D. Garvey, "Perceptual strategies for purposive vision," AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Tech. Rep. 117, Sep 1976.

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

[11] K. Sjöö, A. Aydemir, D. Schlyter, and P. Jensfelt, "Topological spatial relations for active visual search," Centre for Autonomous Systems, KTH, Stockholm, Tech. Rep. TRITA-CSC-CV 2010:2 CVAP317, July 2010.

[12] P. Viswanathan, D. Meger, T. Southey, J. Little, and A. Mackworth, "Automated spatial-semantic modeling with applications to place labeling and informed search," may. 2009, pp. 284 –291.

[13] G. Hollinger, D. Ferguson, S. Srinivasa, and S. Singh, "Combining search and action for mobile robots," in *ICRA'09: Proceedings of the 2009 IEEE international conference on Robotics and Automation*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 800–805.

[14] H. Masuzawa and J. Miura, "Observation planning for efficient environment information summarization," oct. 2009, pp. 5794 –5800.

[15] T. Kollar and N. Roy, "Utilizing object-object and object-scene context when planning to find things," in *ICRA'09: Proceedings of the 2009 IEEE international conference on Robotics and Automation*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 4116–4121.

[16] J. Ma, "Real-time applications of 3d object detection and tracking," Ph.D. dissertation, California Institute of Technology, 2009.

[17] J. Folkesson, P. Jensfelt, and H. Christensen, "The m-space feature representation for SLAM," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 1024–1035, Oct. 2007.

[18] A. Pronobis, O. M. Mozos, B. Caputo, and P. Jensfelt, "Multi-modal semantic place classification," *The International Journal of Robotics Research (IJRR), Special Issue on Robotic Vision*, vol. 29, no. 2-3, pp. 298–320, Feb. 2010.

[19] B. Russell, A. Torralba, K. Murphy, and W. Freeman, "Labelme: A database and web-based tool for image annotation," *International Journal of Computer Vision*, vol. 77, pp. 157–173, 2008.

[20] C. Havasi, R. Speer, and J. Alonso, "Conceptnet 3: a flexible, multilingual semantic network for common sense knowledge," in *Recent Advances in Natural Language Processing*, Borovets, Bulgaria, September 2007.

[21] M. Frydenberg, "The Chain Graph Markov Property," *Scandinavian Journal of Statistics*, vol. 17, no. 4, 1990.