Visual search of an object in cluttered environments for robotic errand service

Xi Chen
The Intelligent Systems Research Institute, College of Information and Communication Engineering Sungkyunkwan University, Suwon, South Korea. pchenxi@gmail.com

Sukhan Lee
College of Information and Communication Engineering, and the Interaction Science Department, Sungkyunkwan University, Suwon, South Korea. lsh@ee.skku.ac.kr

Abstract—Object search is a basic function of a service robot. Previous approaches focus on cases where objects are lying out in the open, which transform object search into active visual search. However, in real life, objects may be behind cupboards occluded by other objects, instead of conveniently lying on a table by themselves. We present a searching method that uses object information, spatial constraints and Bayes probability calculation to handle problems such as occlusion while searching in a cluttered environment.

Keywords-component; occlusion; visual search; cluttered environment

I. INTRODUCTION

Advances in service robot technology have seen relatively slow compared to industrial robot technology. This discrepancy is largely due to the fact that the environments service robots have to cope with are far more dynamic and unpredictable. The task of object search, then, is to generate a set of sensing actions, which will bring the target object into the sensor’s field of view under given direction and angle constraints. For efficiency, this set should consist of minimal number of sensing actions with maximal object detection probability. In the case of search for a 3D object in 3D space, solving the active vision search (AVS) problem is far from trivial. The robot has to interact with objects of various sizes and shapes, and factors such as occlusion and illumination that significantly affect its search outcome.

We consider the case of a mobile robot looking for an object in an indoor environment. Both 2D and 3D features are used for object recognition. By taking advantage of the complementary roles of searching and recognition, we design a framework that allows cooperation and information between them. Finally, we demonstrate our method using a mobile robot. This system is being currently used in our 3rd generation service robot, Korus Homemate, which is equipped with an MS Kinect RGBD sensor, Bumblebee 2 stereo camera, as well as an onboard Intel Core i7 notebook. This robot has been used in several elderly-centers as a test and evaluation prototype.

The related works in this field are introduced in section II; a general description about object searching is presented in section III; our method is introduced in sections IV and V; and lastly, experimental results are provided in section VI.

II. RELATED WORK

Significant progress has been made in objects recognizing and searching. Ye and Tsotsos [1] first relaxed this problem by formulating object search as an active vision problem, where an efficient trajectory of camera views, that localizes the target object, is sought. Garvey [2] proposed the idea of indirect target search: first, a sensor is directed to search for an intermediate object that commonly participates in a spatial relationship with the target. The idea of using object contextual information has been repeatedly identified as an important facet of object search. Wixson et al. [3] elaborated the indirect search idea and have shown efficiency gains both theoretically and empirically. Other demonstrations of the idea have also shown good performance. Sjöö et al. [4] and Aydemir et al. [5] used spatial relations between objects to search more efficiently, and locate good views of the target object. Kollar and Roy [6] showed that object-object and object-location co-occurrence statistics can be used to predict target object locations and decrease the expected length of planning.

For object search, Bourgault et al. [7] employed a Bayesian approach, which uses a target probability density function as prior information. Kollar and Roy [4] used co-occurrence statistics gathered from the Web to train a Markov random field model to indicate the likelihood of objects being at given locations. Extensions to this approach involving information retrieval ([8]) and hierarchical spatial models ([9]) have also
been explored. Kunze et al. [10] applied the semantic similarity measurement, mentioned above, for object search by using a Web-trained ontology. Shubina, K., and Tsotsos [13] divided this problem into “where to look” and “where to go”. Finally, Jolo et al. [11] illustrated a way of combining both types of information by extracting features to train a reactive search heuristics

All previous works address only situations in which target objects are lying out in the open and can be found once it appears in camera’s field of view. In real life, however, objects are often located behind other objects and can only be seen from certain directions. Also, changes in environmental conditions, such as illumination, may lead to performance uncertainty of the recognition algorithm; for example, SIFT-based recognition will fail in an extreme dark environment, even if the target object is in the field of view. A searching process should respond to such situations and decide appropriately.

We provide a powerful searching agent that can deal with cluttered environments, occlusion, recognition uncertainty and different illuminations.

III. OBJECT SEARCH PROBLEM

Ye and Tsotsos define object search as a problem of maximizing the probability of detecting the target within a given cost constraint [1]. Their formulation includes the influence of a search agent’s initial knowledge and the performance of available recognition algorithms.

A search region \( \Omega \) is a 3D space to be searched. We assume that the boundaries of \( \Omega \) and the contexts inside are known exactly. The region \( \Omega \) is tessellated into a 3D grid of non-overlapping cubic elements \( c_i \), \( i = 1 \ldots n \). A sensing action \( s \) consists of taking an image of \( \Omega \) and running a recognition algorithm to find out whether the target object is present in the image or not. The parameter set of \( s \) consists of the camera position \( (x_c, y_c, z_c) \), pan-tilt angles \( (p, t) \), focal length \( f \) and a recognition algorithm \( a \); \( s = s(x_c, y_c, z_c , p, t,f,a) \). The cost function \( t(s) \) for an action \( s \) gives the time required for execution of an action. The time includes the times required to move the camera from one configuration to another, to acquire an image, to run the recognition algorithm and to update the information about the environment, respectively. The detection function \( d(c_i,s) \) gives the probability of detecting the target by action \( s \), given that the target object is centered at cell \( c_i \). If cell \( c_i \) is outside the image, \( d(c_i,s) = 0 \). If cell \( c_i \) is inside the image, the value \( d(c_i,s) \) is determined by various factors: the performance of the recognition algorithm, distance between the camera and cell \( c_i \), illumination of the environment, orientation of the object etc.

Then using the notation introduced above, the probability of detecting the target by action \( s \) becomes

\[
p(s) = \sum_{i=1}^{n} p(c_i,\tau) \ast d(c_i,s)
\]

where \( \tau \) is the time just before \( s \) is applied.

The task of object searching can also be represented as finding an ordered set of actions that we need to apply on the robot to finally find the target object. Ye has proved that this problem is NP-hard.

A “greedy” algorithm is sufficient for obtaining a good approximation to a solution. Here we can simply use the action \( s \) that will maximize \( p(s) \) in the current condition as the next action.

IV. THE SEARCH STRATEGY

The amount of the calculation for generating a next action increase exponentially with the size of searching space. We use several approaches, such as the spatial relation of the target object, to reduce the search space \( \Omega \) and hence, reduce the complexity of the calculation.

A. Spatial relations

Objects, used by humans, are not usually placed randomly in work space, but rather, well organized by humans, to fulfill various functional purposes. This organization is expressible in terms of spatial relations. Books are usually placed in a book shelf, while food is usually stored in the kitchen. If we know the position of the object/place that is related to the target, the searching space can be dramatically reduced even before the start of visual search.

We call the target-related object/place, which may have the target object on/in it, as “context”; and one target object may have multiple related contexts. We assume that the target object can only be found in the context area. We also assume that the sizes, heights, locations, orientations, and accessible angles of all the contexts are known in advance and that these characteristics will not change during the searching process. The range of the accessible angle of a context determines the direction in which the area on/in the context can be observed. For example, a regular table has a 360-degree accessible angle range. A container, such as a book shelf or a box, on the other hand, only has one or two open surfaces that can be observed.

When the target object has multiple related contexts, related probability is assigned to each context to represent the likelihood of finding that target object in/on each context.

B. Interaction with recognition system

Even though a cell-based representation provides a feasible solution to the NP-hard problem, it still poses a quantization problem, that is, the uniformity of probability distribution of finding object across one cell. The cell-based representation may not be the case in reality. If, for example, two candidate objects were located in one cell, and each object had its own probability to be the target object, describing the probability distribution using only one center point probability value will not be accurate. Furthermore, scanning a cell that has multiple candidate objects, from different perspectives will produce different results. For example, by observing a cell with two candidate objects, from one direction, we may be able to see both objects. By observing it from another direction, however, we may only see one object; which might be occluding the other object. In general, occlusion problem cannot be handled well by using only the cell based representation.
One solution is to reduce the size of each cell to be so small that only one object can be located inside of it. Such a solution, however, will increase the number of cells, and thus, increase the computation time exponentially. In our proposed method, we present a solution to this problem by introducing the context of “object” into each cell.

Our recognition system relies on two main features: SIFT-based recognition and geometric-based recognition. SIFT is a powerful, texture-based, 2D matching algorithm. To use it in our 3D recognition system, we extended it by comparing 3D points representation of matched 2D points; with 3D points of target object model. This allowed us, not only to identify and remove outliers, but also to estimate the 6DOF pose of the target object accurately.

We also developed our own geometric-based “3D Shape Descriptor” that takes advantage of our fast octree representation of a 3D camera dense point cloud. It extracts several evidences, such as geometric aspects, body shape index, extreme/edges mapping, and convex/concave descriptors. It then computes the probabilities of the evidences by using multivariate distribution against pre-stored training classifiers, and fuses these probabilities by using a Bayesian evidence structure.

SIFT will be recognize objects with unique texture, while the 3D shape descriptor will recognize objects with unique shape.

The recognition system also provides information about the objects in current FOV if there are any. Let $o_k, k = 1 \ldots n$ represent the candidate objects we observe in current FOV. Assuming that an object is located on a context accessible surface, $z_o = z_{\text{context}} + h_o/2$, the information about the object would consists of the following: its center point $(x_o, y_o)$, width, breadth, height $(w_o, b_o, h_o)$, probability of its being the target object $p(o_k)$, and the feature reliability $d(x, y, s)$ from the current robot location $(x, y)$. Using object center point $(x_o, y_o)$, we can locate objects on map, and using the object shape $(w_o, b_o, h_o)$ we can calculate which cell is visible.

**Figure 2.** An example of recognition output. (left) Image taken by camera. (right) Result of shape-based recognition algorithm using object shape and probability. The one in the red box is the target object.

### C. Action space

After search space $\Omega$ is reduced, the action space, in which searching action can be performed, can also be reduced. We define action space $\Psi$ as the set of cells from where we can detect at least one cell in $\Omega$. For a target object $k$, $D_{\min}(k)$ and $D_{\max}(k)$ are the minimum and maximum detectable distances, respectively. $C(c_j), c_j \in \Omega$ is the operator that identify and returns the context to which search space cell $c_i$ belongs. $a\left(c_i, C(c_j)\right), c_i \in \Omega$ is an operator that return 1 if $c_i$ is in the accessible direction of context to which $c_j$ belongs and return 0 otherwise. For any $c_i \in \Psi$, if there exists at least one cell $c_j \in \Omega$ that satisfies

$$D_{\min}(k) \leq \text{dist}(c_i, c_j) \leq D_{\max}(k)$$

$$a\left(c_i, C(c_j)\right) = 1$$

Where dist$(c_i, c_j)$ is the Euclidean distance between $c_i$ and $c_j$, we say $c_j$ can be detected from $c_i$.

**Figure 3.** One example search space $\Omega$ and candidate space $\Psi$. The cells in yellow belongs to $\Omega$ and the cells in other colors belongs to $\Psi$. The red line represents the accessible direction of context, and the circle means 360 degree accessibility.

### D. Action

The definition of action $s$ can now be rewritten as $s = s(c_i, z_o, p, t, f, a), c_i \in \Psi$. From equation (1), the value of $p(s)$ depends on the number of searching space cells in FOV, and their individual probabilities. Hence, a camera looking at a context will clearly produce a better FOV, than one looking somewhere else; such as the air and floor. Suppose there is only one configuration $(p, t, f)$ that can project a point $(x_o, y_o, z_o)$ onto the center of the camera image plane. Action $s$ can be rewritten as $s = s(c_i, z_o, c_j, a), c_i \in \Psi, c_j \in \Omega$ if size of $c_j$ is small enough. Using a robot with a fixed camera height will allow us to remove $z_o$ from the parameter space. Thus, the action $s$ to be used in this paper is $s = s(c_i, c_j, a)$ where $c_i \in \Psi, c_j \in \Omega$.

### E. Searching order

If the number of related contexts is large, the calculation will be intractable even after $\Omega, \Psi$ and action spaces are reduced. To make the calculation tractable, we decide to search one context at one time. The maximum size of $\Omega$ is equal to the maximum size of the context in the related context list. When there are more than one context to search, the visiting order must be decided. Related probability is the most important factor to consider. The context with the biggest probability of finding the target object should be searched first. The time needed for a robot to move from its current position to a position $c_j$ in which it can scan a context should be considered.
Our method aims to find a sequence of positions in $\Psi$, that allows visits to all contexts in minimum time. Let $l = \{C_1, C_2, ..., C_n\}$ be a combination of all related contexts. The time needed to visit all contexts is calculated by

$$t(l) = t(R, C_1) + \sum_{i=1}^{n-1} [t(C_i, C_{i+1}) + \prod_{j=1}^{i} (1 - p(C_j))]$$

(4)

where $t(R, C_1)$ is the time for a robot to move from its current position to a position in which it can scan the center of the first context in the list, $t(C_i, C_{i+1})$ is the time for a robot to move from context $C_i$ to $C_{i+1}$ and $p(C_j)$ is the probability of finding the target object on $C_j$.

Let $L = \{l_1, l_2, ..., l_n\}$ be a list of all possible search sequences that allows visits to all relevant contexts. The optimal sequence $l_{opt}$ is the one which makes

$$t(l_{opt}) = \text{argmin} \sum_{i=1}^{n} t(l_i)$$

(5)

![Figure 4. An example of searching space. The red cell is the current location of the robot and the yellow cells are the contexts locations](image)

V. SEARCHING PROCESS AND IMPLEMENTATION

The searching process starts from environmental data acquisition (recognition). After each recognition process, new recognition data should be added to the prior knowledge pool. Our algorithm does this through two main stages. In stage one, answers are found to questions such as “How many objects have we found?”,”Which cells are we looking at?”,”Which cells have high probability and “Which cells do not need to be searched further?” $\Omega$ will be updated in the first stage, so $\Psi$ need to be recalculated. The main goal of the second stage is to generate a pose or action for every candidate cell in the updated $\Psi$, and to select the best one to be the next pose.

A. Probability updating

After we apply action $s$, the probability of finding the target object in a visible cell or object will be updated by

$$p(l, \tau_s) = p(e_i | r_t, s) \ast p(r_t, s) + p(e_i | \tilde{r}_t, \tilde{s}) \ast p(\tilde{r}_t, \tilde{s})$$

(6)

where $l = (x, y, z)$ is a center of a cell or object in $\Omega$. $p(l, \tau_s)$ is the probability of finding the target object at $l$ after action $s$ is applied. $p(r_t, s)$ is the measured probability of recognizing the target at $l$ using action $s$. $e_i$ is an event indicating the presence of the target object at $l$ and $\tau_t$ is an event indicating the successful recognition of the target object at $l$ by the recognition system $\tilde{l}$. Then, $p(e_i | \tilde{r}_t)$ is the probability that the target object is actually located at $l$ given that the recognition system says it is at $\tilde{l}$. $p(e_i | \tilde{r}_t, s)$ represents the reliability of recognition result. Using Bayes’ theorem, we rewrite $p(e_i | \tilde{r}_t)$ as

$$p(e_i | \tilde{r}_t, s) = \frac{1}{1 + \frac{p(r_t, s)|e_i|p(e_i)}{p(\tilde{r}_t, s)|e_i|p(e_i)}}$$

(7)

$$p(e_i | \tilde{r}_t, \tilde{s}) = \frac{1}{1 + \frac{p(\tilde{r}_t, s)|e_i|p(e_i)}{p(\tilde{r}_t, s)|e_i|p(e_i)}}$$

(8)

where $p(e_i | \tilde{l}) = p(l, \tau)$. $p(r_t, s|e_i)$ is the probability that the recognition algorithm will recognize the target object at $l$. Similar with the concept of detecting $d(c_t, s)$ in Ye and Tsotsos approach, $p(r_t, s|e_i)$ is closely related to the recognition algorithm, as well as the conditions in which recognition measurements were taken, which include the distance between a camera and $l$, the illumination, and the orientation of the target? object.

For SIFT-based recognition, we built a database of objects, in which each object will have 16 images and a 3D model for every 22.5 degrees. During recognition, we extract SIFT points in 4 different octave scales for each image. As described in [12], we can calculate the number of expected detectable SIFT points at different distances, illuminations, and orientations. The detection probability $p(r_t | e_i)$ of a SIFT point is defined as a sigmoid function of the number of expected match points. For the 3D shape descriptor, $p(r_t | e_i)$ is defined based on the error distribution of a depth measurement of a 3D camera.

When the environment condition is bad for a recognition algorithm, for example the distance from the camera to the object is far or the illumination is too bright or too dark, the result of recognition is not reliable. At this time, the value of $p(r_t, s|e_i)$ will become very small, making the value of the updated probability similar to that of the prior probability. The probability will not affect much by the unreliable recognition result.

The minimum and maximum detectable distances of an object, $\kappa D_{\text{min}}(\kappa)$ and $D_{\text{max}}(\kappa)$, are also calculated using the same method. Suppose $N$ SIFT points are needed for recognition, the minimum distance $D_{\text{min}}(\kappa)$ and maximum distance $D_{\text{max}}(\kappa)$ for SIFT are the closest and the furthest distance that guarantee the expected number of extracted SIFT points to be greater than $N$ from at least one of the 16 DB images. For the 3D shape descriptor, $D_{\text{min}}(\kappa)$ is the minimum
distance at which a sensor can generate a 3D point cloud, and \( D_{\text{max}}(\kappa) \) is the maximum distance at which octree representation can still reliably preserve and represent the shape of the target object \( \kappa \).

B. The best pose

After updating the probabilities of a visible cell and object, the cell or object which does not need to be rechecked is eliminated. Let \( F \) be the threshold, and all cells and objects with a probability lower than \( F \) will be eliminated from \( \Omega \). The bigger \( F \) is, the faster the size of \( \Omega \) will be reduced. If \( F \) is 0, we will keep watching until all probability values reduce to 0.

Since \( \Omega \) is updated, the action space \( \Psi \) is also updated. For each cell \( c_i \in \Psi \), the possible action we can take at \( c_i \) is \( s(c_i, c_j) \) where \( c_j \in \Omega \) and \( c_j \) can be detected by \( c_i \). Let \( v(l_i, s) \) be an operator that returns 1 if point \( l_i \) is visible for action \( s \). The probability of detecting the target by action \( s = s(c_i, c_j) \) becomes

\[
p(s) = \sum_{i=1}^{n} p(c_i, r) * p(r_j, s|e_j) * v(c_i, s) + \sum_{j=1}^{m} p(o_j, r) * p(r_j, s|e_j) * v(o_j, s)
\]

where \( n \) is the number of cells that can be detected by \( s \) and \( m \) is the number of objects that can be detected by \( s \). \( p(o_j, r) \) is the probability of detecting the target at the \( j \)th object.

Let \( S(c_i) = (s_1 ... s_n) \) be all the possible actions we can take at \( c_i \). The one with maximum \( p(s) \) is

\[
M(c_i) = \arg \max \sum_{i=1}^{n} p(s_i)
\]

VI. EXPERIMENT RESULT

We tested our algorithm for different conditions. In the experiment shown here, we demonstrate the ability of our proposed algorithm to search and locate objects occluded in a highly-cluttered unknown environment.

First, we placed two tables, with highly cluttered objects, in the test environment, as shown in figure 6. The size of the searching environment was 2.2m \( \times \) 3.6m (width \( \times \) length) and the size of each table was 0.6m \( \times \) 0.6m (width \( \times \) length). We, then, placed our target object, the “Chilsung Cidar” soda can, behind the blue book on the white table. Furthermore, we surrounded the target object with other objects to limit its accessible range to less than 180 degrees, as shown in figure 7. Finally, we directed our mobile robot to search and locate the target object using the proposed algorithm.

Probability of detecting the target object at each cell is assigned, and best pose is selected, as shown in figure 8. Since, the target object was not present in this view, the recognition system provided object information with low probabilities for each of them. These objects were registered on the map, as shown in Figure 9, and the cell probabilities were updated.
Since this context was fully scanned, all of its cell probabilities were updated and the algorithm realized that the cell probabilities of the next context were higher. So, the probability of detecting the target object at the nearest accessible cell of the next context became the highest (despite the high cost of movement) and the robot selected it as the next best pose, as shown in Figure 11.

The next procedure is similar. The robot looks at context 2. Recognition system, however, wasn’t able to locate the target object (as it was occluded by the book). The objects are mapped and cells’ probabilities are updated as shown in Figure 12.

The next best pose was chosen so that it can see the occluded cells. The target object became visible to the robot and the recognition system was able to locate it successfully, as shown in Figure 13.

The search for an object in a 3D space benefits greatly from knowing the environment conditions, such as illumination, as well as identifying other surrounding objects. We present a solution by using octree to represent the 3D information of objects in the environment, and register it on a 2D map. As shown in the experimental results, this solution allows the searching agent to detect and avoid occlusion, which is the most challenging part of a search in a clutter environment. We reduced the searching space using spatial information at the beginning, and simplified the calculation of occlusion detection, which greatly reduced the calculation time.

In our future work, we will investigate more complicated environments, as well as objects with more recognition constraints such as beverage cans that can only be recognized from one direction, and multiple active approaches for occlusion avoidance, such as manipulation of objects to eliminate occlusion.

VIII. ACKNOWLEDGMENT

This research was performed for the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs (F0005000-2010-32), and in part by the KORUS-Tech Program (KT-2010-SW-AP-FSO-0004) funded by the Korea Ministry of Science, ICT and Planning (MSIP). This work was also partially supported by WCU (World Class University) program through the National Research Foundation of Korea funded by the Ministry of Education (R31-10062), and by MSIP, Korea under ITRC NIPA-2013-H0301-13-3001.

IX. Reference


