2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Hybrid Laser and Vision Based Object Search and Localization

Dorian Gálvez López, Kristoffer Sjö, Chandana Paul and Patric Jensfelt
Centre for Autonomous Systems, Royal Institute of Technology
SE-100 44 Stockholm, Sweden
dorian3d@gmail.com, {krsj,chandana,patric}@nada.kth.se

Abstract— **We describe a method for an autonomous robot to efficiently locate one or more distinct objects in a realistic environment using monocular vision. We demonstrate how to efficiently subdivide acquired images into interest regions for the robot to zoom in on, using receptive field cooccurrence histograms. Objects are recognized through SIFT feature matching and the positions of the objects are estimated. Assuming a 2D map of the robot's surroundings and a set of navigation nodes between which it is free to move, we show how to compute an efficient sensing plan that allows the robot's camera to cover the environment, while obeying restrictions on the different objects' maximum and minimum viewing distances. The approach has been implemented on a real robotic system and results are presented showing its practicability and the quality of the position estimates obtained.**

I. INTRODUCTION

As the field of mobile robotics expands and more ambitious goals are set for autonomous robots, one subfield that is opening up is interaction with objects. A great portion of the potential applications envisioned for autonomous agents involve some form of interaction with specific objects in the environment, if only in the form of observation and registration. Yet so far most robotic applications tend to be one of two things: Either entirely blind to everything in their surroundings except what is required merely for navigating through them, or else designed to function in a fixed setting, where they have a well-known and unchanging frame of reference that they can relate objects to. Nevertheless, we are beginning to overcome these limitations. For example, the robot league of the Semantic Robot Vision Challenge (SRVC) [1] is a promising attempt to advance understanding in this area. The purpose of this paper is to take a step towards merging efficient object detection and recognition with existing methods for robot navigation in a realistic, free-roaming setting.

The contributions of this paper lie within the subject of object search. As such, it touches simultaneously upon two areas hitherto not much integrated: view planning and visual search; in other words, optimizing our sensing strategy before and after we begin to acquire visual data, respectively.

View planning is a comparatively old but still thriving research area, and is crucial to efficient object detection in realistic-scale environments, as exhaustive search under such conditions is unfeasible. The general problem of finding a minimal set of viewpoints from which to observe all parts of the environment is called the *art gallery problem*. [7] proves that this problem is NP-hard, and thus approximate solutions are required. Another related view planning problem is the *watchman problem*, which entails computing a minimal continuous path through space from which all of the environment can be seen; here, the length of the path is what is crucial - in contrast to the art gallery problem, where the distance between viewpoints is immaterial. The watchman problem, too, is NP-hard when there are "holes" in the free space (as shown in [10]).

Early work on view planning was of a mostly theoretical nature, but as the field has matured more implementation-oriented results are emerging. [14] examines the problem of optimally covering the "view sphere", i.e. all angles that can be seen from a fixed point in space, given a probability distribution for the presence of the object. In [15], the approach is augmented with multiple viewpoints, each subsequent point selected by a greedy policy. Using a polygonal map of the robot's surrounding, [4] applies a sampling scheme to find an approximate solution to the art gallery problem while additionally taking into account the practical limitations of sensors by postulating maximum and minimum distances and maximum viewing angle. However, parameters for only a single object are considered. In [12], the costs of moving and processing views are combined in a single planning task, approximated as an integer linear problem (ILP). A set of candidate view points is assumed to be provided. Many other approaches exist to solving both the art gallery and watchman problems; [11] provides an extensive survey.

The approach to view planning proposed in this paper aims to move beyond considering only a single object, instead performing an efficient simultaneous search for several objects of different appearance and potentially different size, given a 2D map of the environment that has been acquired by the robot itself (as in for example [6]). This is accomplished through a greedy algorithm that selects the best viewing cones, constrained by the maximum and minimum viewing distances associated with each object.

Visual search is also highly relevant to efficient object detection and localization. Often, the image that a camera captures is not immediately sufficient to perform accurate object detection, especially with low-resolution cameras. For successful detection and recognition, either moving closer

This work was supported by the SSF through the Centre for Autonomous Systems (CAS), the EU FP6 IST Cognitive Systems Integrated Project "CoSy" FP6-004250-IP, the STINT foundation and the Swedish Research Council, contract 621-2006-4520

or zooming in on the object is necessary. Several different methods have been proposed for determining the area of interest of an image, so that the robot knows where to zoom in. [13] demonstrates a foveated dynamic attention system which uses edges and circular features to direct attention, though in a non-specific fashion; this is also the case in [5], where a measure of feature saliency inspired by human cognition is similarly used in order to provide a sequence of attentional saccades to potential interest points in the image. The top-placing entrants in [1] similarly use non-specific saliency to direct attention for object detection.

An attention control method that uses contextual information is described in [9], although its specificity is to the area surrounding objects rather than objects themselves. [2] describes an object-specific attentional mechanism utilizing receptive field cooccurrence histograms or RFCH, which provide different hypotheses for the occurrence of each object in the image.

We build on this principle, adding a distance estimate that facilitates zoom computation along with a more efficient view planning strategy. Using a combination of view planning and visual search, we show how existing computer vision methods can be combined to produce an autonomous robotic system that is able to efficiently detect and localize different objects in a realistic indoor setting. We have implemented the proposed system on a mobile robot and demonstrated its practicability in experiments.

A. Hardware

The robotic platform used in this work is a Performance PeopleBot. It is equipped with a SICK laser rangefinder with a 180 degree field, positioned near the floor (at about 30 cm), and with a Canon VC-C4R video camera, able to acquire low resolution footage (320×240 pixels) with pan/tilt functionality and up to $13\times$ magnification. The camera is mounted about 1m above the floor. The robot has a differential drive and a wireless LAN connection.

B. Training data

The system needs to be provided with training data for each object, both for the view planning and the visual search. This includes: An up-close image of the object, pre-segmented; its true width W_{real} and height H_{real} in metres (the objects are considered 2-dimensional for the purposes of recognition and distance estimation); and manually selected threshold levels T_{far}, T_{mid} and T_{near} used in the image search. Also, the view planning algorithm needs to know whether an object is unique or might occur several times.

The training image is processed by RFCH and SIFT algorithms to provide a set of histograms and key points, respectively, which are used as basis for both detection, distance estimation and recognition.

II. NAVIGATION

The robot is provided with a metric 2D-map, consisting of line features representing walls and other obstacles, as well as with its own location in this map. The map is assumed to have been generated in advance by the robot from laser data, using standard SLAM methods; in our case the ones previously described in [3]. It is also given a set of nodes in 2D-space with edges between them, constituting a *navigation graph* which represents known robot-navigable space [6]. This is generated during mapping; as the robot moves into unvisited areas, it drops nodes at regular intervals, and when it moves between existing nodes it connects them.

The object search task begins with a planning step, in order to determine a good way to explore the map. In this paper, only the navigation nodes are considered, as they are the only parts of the map guaranteed to be reachable. This constraint obviously simplifies the solution a great deal computationally, compared to view planning approaches that consider all the space.

The map is assumed to be of a single room, i.e. a basically convex space, though possibly with a lot of obstacles inside. Starting with a more complex map, the map of a single room can be obtained given a subset of the navigation graph constituting *door nodes* [6]. Cutting out all door nodes, each remaining subgraph represents a room, and all features of the map are assigned to the room which has the nearest navigation node. Planning efficient movement between rooms is beyond the scope of this paper and currently a closest-next-room-first strategy is used.

The navigation plan must provide the robot with combinations of nodes it needs to visit, views from those nodes that it needs to process and the objects it has to look for in each view, so that all parts of the room are searched for all objects, while keeping the number of visited nodes and visual searches as low as possible. Object constraints must also be fulfilled as, for example, a very small object can only be seen from a short distance away and vice versa. In addition, uniqueness must be taken into account: objects should be discarded once they are found, which means the exploration plan will need to be updated.

A. Grid-based view planning

1) Occupancy grid: The metric map that we get from SLAM is not geometrically perfect. Features extracted from laser data do not form a clean, continuous outline; typically, many different more or less overlapping line features explain the same sensor data; the resulting clutter would increase planning complexity. For this reason, a simpler occupancy grid-based method is used. The occupancy grid can be acquired either directly from laser data or by rasterizing an existing feature map (by simply marking a square as occupied if it contains any feature).

Note that the occupied squares are not assumed by the algorithm to obstruct vision. As the data comes from the laser, which is close to the ground, occupancy need not correspond to occlusion for the more highly placed camera.

Grid square size is a tuning parameter; a small square size will result in a lot of points to cover, which means higher accuracy but also higher computational cost. Small squares will be very closely packed and will get grouped into the same views. On the other hand, a too-large square size will lead to insufficient detail in the plan and may miss parts of the map to explore. The choice depends on the overall granularity of objects; in this work, a fixed square size of 0.5m is used.

2) Views: Using this grid, *views* can be calculated. A view is a triplet consisting of the map node to which the robot has to travel, the direction it should point its camera and the list of objects to be searched for. In order to simplify calculations, grid squares are considered visible in a view if their center point is inside the field of view.

3) Object constraints: As several types of objects are being looked for, their specifications must be taken into account; specifically, their sizes. Not all objects can be seen at the same distance. For this reason, for each object a minimum and a maximum distance are defined; the robot should attempt to find it only at distances in this interval.

There are separate distance constraints for object recognition and object detection. For recognition, the minimum distance is simply defined as the range at which the object would fill an entire image at no zoom; the maximum, as the range at which the object would occupy an entire image if maximum zoom were used. The minimum distance for purposes of detection, on the other hand, is given by the parameters of the detection algorithm; see (III-B).

Figure 1 shows an example of two potential views of a set of squares (in this case, a wall). Large circles represent nodes; dots denote grid square centers; the numbers next to them indicate the nodes they are associated with; and the shaded area represents the views (along with objects planned for in each view). Note that neither view allows for seeing all objects due to their different sizes; thus, a plan must incorporate two different views in this direction.

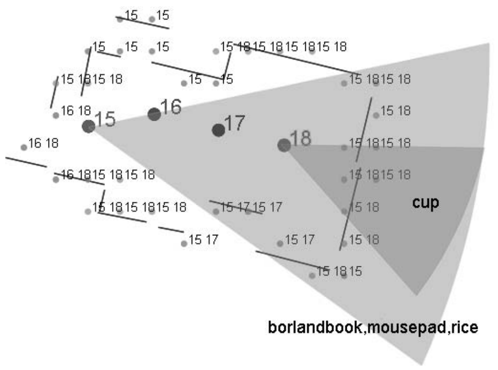


Fig. 1. Example of effect of distance constraints on view planning

4) Strategy: The objective of the algorithm is to ensure that any occurrence of the sought objects will be seen in some planned view regardless of which square it is in – in other words, each object-square combination must be covered by some view.

The space of possible views being continuous, a set of candidates is produced by the following discretization: for each node and for each point to cover, include the view that has that point at its leftmost edge (Figure 2). No views exist that contain more points than those in this set.

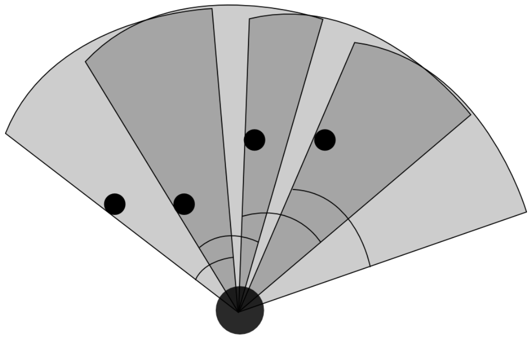


Fig. 2. Discretization of candidate views

Out of these candidates, the view covering the most object-square pairs is picked iteratively, removing any pairs covered from the list, until no pair is left uncovered that could yet be covered by some view. Any remaining pairs are considered impossible and subsequently ignored.

The plan consists of visiting the closest navigation graph node possessing a view that was picked, performing object search for all views from that node, then moving on to the next closest node and so on.

If any object known to be unique is found during the visual search, it can be eliminated from the rest of the plan; if any views in the plan are rendered empty by this, they can be removed as well. Non-unique objects will be searched for in every occupied square.

As is evident from the above, the algorithm proposed is greedy in terms of nodes and map squares. Although it does not ensure an optimal solution, it allows for obtaining a low number of views in polynomial time.

5) Tilt angle selection: Since we are using a 2D map of the environment, there is no direct information that could help in deciding how to use the tilt angle of the camera. Yet, the objects being sought might be at any height, and so some thought must be given to covering the vertical dimension as well as the horizontal.

Any grid squares which are closer to a given view's associated node than a set threshold (here, 2 meters) generate new views that cover the vertical extent of the objects' possible locations. Using the average distance for those grid squares, together with an upper and a lower boundary for objects' positions, one or more tilt angles are selected (with as little overlap as possible) and the resulting views are added to the plan.

III. VISION

A. System overview

The vision system enables the robot to look for objects using images taken by its video camera. For object detection, the system uses the concept of receptive field cooccurrence histograms (RFCH) as described in [2]. As potential objects are detected, the system calculates suitable interest regions for the camera to magnify. For this, the system needs an estimate of the distance in order to decide whether to proceed with recognition directly or zoom in further. In [2] this estimate was taken from the laser scanner; we instead obtain an estimate through the RFCH detection procedure itself.

If the distance allows for reliable recognition immediately, the system attempts SIFT feature matching in order to recognize the object [8]. Otherwise, the interest regions for the different objects are merged as far as possible and the camera zooms in on each region in turn. This procedure is repeated recursively, until each object either is found or its presence ruled out.

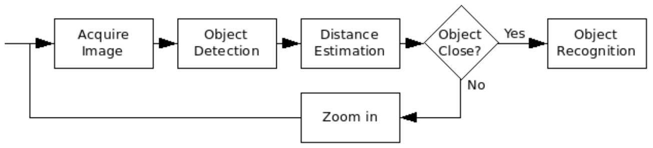


Fig. 3. Overview of image search

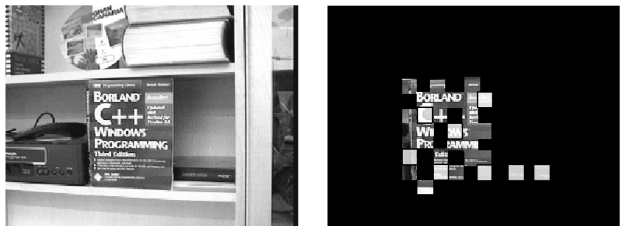
Figure 3 shows an overview of the vision system. The detailed operation of this system is presented in the following sections.

B. Object detection

The detection process works on a per-object basis and makes use of an RFCH algorithm. It consists of several steps:

First, an image is taken by the camera. Second, it is divided into cells. Receptive field cooccurrence histograms are computed (using clusters learned from each respective object during training), and matched against the training image's histogram, resulting in a similarity value for each cell and object. These values are called the object's *vote matrix*.

Third, hypotheses are generated; see Figure 4. A cell is a hypothesis for an object if its value is greater than those of its 8-connected neighbors, as well as greater than an object-dependent threshold, T_{mid} or T_{far} (which one depends on whether the camera is currently zoomed in or not).



(a) Example image (b) Hypotheses of the book

Fig. 4. Hypotheses extracted for the book in the example image. Lightest squares represent hypotheses.

The size of the vote cells in the above algorithm is a tuning parameter. Large cells mean faster histogram matching; however, they decrease the detection rate when they are larger than the objects in the initial image. Also, maximum distance allowed for object detection during the planning step is set to the distance at which an object would occupy a single cell at no zoom, which decreases as cell size grows. The value used in this work, 15×15 pixels, is a compromise between these considerations.

C. Distance estimation

In [2], the initial distance estimate used to determine zoom levels came directly from the robot's laser sensor. The problem with this design is that the distance given by the laser is often misleading: the laser sensor is placed at a fixed height above the floor (about 30 centimetres) and if an object is not at that height, e.g. if it is on a table, the estimate will be off. It typically works only for objects that are next to walls (such as on a bookshelf). If the distance estimate is much too small or large, the final zoom may either not be sufficient to make the object occupy enough of the image for matching, or otherwise may be too great causing only a small part of the object to be seen. Furthermore, even if the object is recognized, its estimated position might be inaccurate. To address these issues, in this work we use two alternative ways of getting a distance estimate.

1) Using the vote matrix.: Using the RFCH vote matrix for distance estimation consists of measuring how many cells are part of the object and treating the area they occupy in the image as the size of the object in the image. Here, cells are considered to belong to a hypothesis if their degree of match is above the threshold and there is an 8-connected path of cells with monotonically increasing value to the hypothesis (and no shorter such path to any other hypothesis), as shown in Figure 5. Only the strongest hypothesis and its associated 8-connected cells are taken into account, because it is likely to be the most reliable.

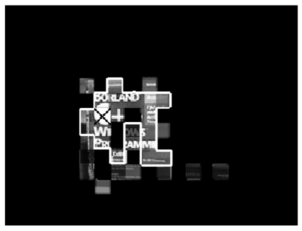


Fig. 5. A hypothesis, marked X, and its Eight-connected associated cells

Given the object's real size, stored in the training database, the distance can then be computed according to:

$$D = \frac{W_{real}\dfrac{W_{im}}{2 \cdot W_{vote}}}{\tan\left(\dfrac{\alpha}{2}\right)}$$

where D stands for the estimated distance (metres); W_{im}, for the width in pixels of the image, W_{vote}, for the width in pixels of the bounding box of the cells belonging to a hypothesis and α, the horizontal viewing angle.

This estimate is quick and rough, but sufficiently accurate to allow the object search algorithm to assign a valid zoom.

2) Using SIFT: SIFT extraction produces a scale parameter for each key point extracted. For each matched pair of key points in the training and recognition image, the quotient of the keys' scale parameter gives an estimate of their relative apparent size and hence their distance. This is done according to:

$$D = \frac{W_{real} \dfrac{W_{im}}{2 \cdot W_{tr}} \dfrac{S_{tr}}{S_{rec}}}{\tan\left(\dfrac{\alpha}{2}\right)}$$

where S_{tr} and S_{rec} stand for the scale of the SIFT point as extracted from the training image and the recognition image respectively; and W_{tr}, for the pixel width of the object in the training image.

As mis-matched key point pairs can produce wildly incorrect scale parameters, the final estimate of the object distance is taken as the median of the distance estimates from all matches. Experiments indicate that an adequate estimate is obtained given 10 or more SIFT matches. With 4 matches or more a passable rough estimate is typically obtained. If there are fewer than 4 matches, they are likely to be flawed and the resulting estimate is not used. Experimental results for this technique can be found in (IV-A).

Although the above method for calculating the distance is good, it has some drawbacks. The main problem is that extracting SIFT points from an image is computationally expensive, and using it to guide the zoom process may take too long to be feasible. Another problem is the number of SIFT points required to obtain a robust estimation; when the object is small in the image, it is unlikely that enough matches will be available.

D. Calculation of zoom

Given a training image of an object, its known real size, the distance to the object and the camera field of view, we want to calculate the magnification needed to make it fill the image as nearly as possible. The size of the object is approximated by the size of its bounding box.

In order to make the object fill the image, the horizontal angle of view (α), as well as the vertical (β), can be calculated as:

$$\alpha = 2 \arctan\left(\tfrac{W_{real}}{2D}\right) \quad \beta = 2 \arctan\left(\tfrac{H_{real}}{2D}\right) \qquad (1)$$

Since the object will typically not have the same aspect ratio as the image, only one of the angles α and β can be used to select the actual magnification to use; therefore, of the two levels of magnification suggested by the height and the width respectively, the lower level is selected, as given by the following rule: If $\frac{W_{im}}{W_{tr}} < \frac{H_{im}}{H_{tr}}$ (H_{im} and H_{tr} being the heights analogous to the widths W_{im} and W_{tr}), α is used; otherwise, β.

1) Hypothesis reduction: Even with the threshold, there are typically too many hypotheses to consider one by one. In order to avoid excessive zooming and processing, hypotheses are reduced in two steps. First, they are grouped together into *zoom windows*, which are regions of the image to be magnified and processed in the next iteration. The zoom windows' dimensions are those of the current window, divided by the limiting view angle (as explained above) times the current view angle. The distance estimate from the strongest hypothesis is used for all zoom windows; thus they are all the same size.

A simple greedy algorithm assigns hypotheses to zoom windows until all hypotheses are covered. An example of zoom window creation is shown in Figure 6(a).

However, the distance parameter the vision system calculates is not always very accurate. An error in this parameter propagates into the above calculation of the magnification and into the size of the zoom windows. This may lead to generating more zoom windows than is warranted and, consequently, lengthening the search process and consuming valuable time. Thus, in a second step it is desirable to remove those windows which do not contribute information to the search, as they contain few hypotheses and are located close to "richer" zoom windows.

Therefore, zoom windows which overlap more than 20% with another containing at least 3 times its number of hypotheses are removed. These conditions are quite conservative in order to ensure that no potentially important zoom windows are removed. Figure 6(b), 6(c) show an example of hypothesis reduction.

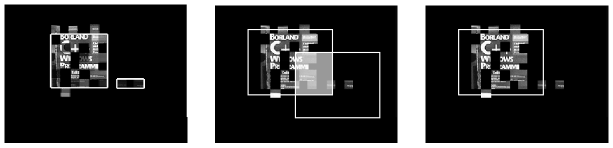

(a) Hypothesis assignment (b) Zoom windows and overlap (c) Final zoom window

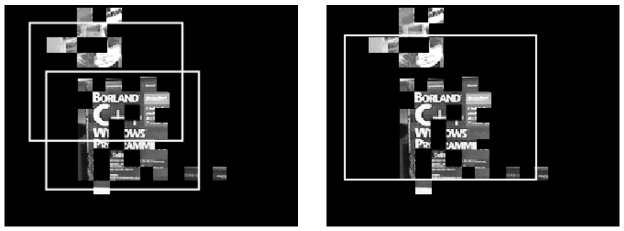
Fig. 6. Hypothesis reduction sequence. Rectangles show the hypotheses assigned to zoom windows and the windows themselves, respectively, and the shaded zone the zoom window overlap.

2) Zoom window sharing: When searching for several objects, the set of zoom windows obtained for each object is computed separately. After this is done, the combined set of windows needs to be merged together. In order to do this, we look for instances of a zoom window encompassing that of another object, in which case we can remove the latter.

Not all zoom windows that overlap in this way can be merged, as straying too far from each object's target magnification may cause object detection to fail. The object search process as a whole goes through up to 3 steps: the first step without zooming, a second step with a middle-level zoom and a third step with large zoom; see (III-F). It is not as important that the middle-level zoom be exact, since it is only used for hypothesis finding with RFCH. Thus, a maximum and a minimum value are defined for the middle-level detection step's zoom level, allowing some flexibility in selecting the windows to be used. The most important thing is to get the minimum zoom right: the lower it is, the more objects we can look for at one time, but the greater the risk that objects are missed due to appearing too small.

The algorithm works as follows: first all zoom windows are set to their minimum size. Then, each zoom window associated with an object A is compared with those of an object B. If the hypotheses contained in one of B's windows can be made to be contained in one of A's – expanding if needed, while conforming to the maximum size for A – then the B window is removed, and object B is added to the A window's list of candidate objects to look for in the next step. This procedure is repeated for each pair of objects. Figure 7 shows an example of how the zoom windows of two object might be merged into one.

Tests have showed that too flexible a window size tends to be harmful to detection. Accordingly, in this work the middle step zoom levels are constrained to a relatively narrow range.



(a) Zoom windows of mouse pad and book

(b) Merged zoom window

Fig. 7. Zoom window sharing example

E. Object recognition

The final object recognition is done once the object is deemed to occupy as much of the image as will fit. It consists of extracting SIFT points from the current image and matching them with the SIFT points obtained from the training image. SIFT points are scale-, position- and rotation invariant up to a certain level, meaning that many of the points will match even if the object is seen from a different angle or under different lighting conditions from the training image. However, it is usually the case that the number of SIFT matches during the search is much lower than the number extracted from the training image, due to changes in position and background. Because of this, we consider an object to have been found when there are matches for at least 5% of its SIFT points. This value has previously been demonstrated to result in few false positives [2].

Once an object is recognized, its position in space is calculated from the estimated position of the object inside the image and the distance calculated by the system; see (III-C).

Because of the large variation present in the images acquired by the robot in a realistic setting, it is very probable that false detections, where no object is present, may reach the last step of the visual search. In order to reduce the amount of unneccessary extraction of SIFT points – which is a relatively expensive procedure – the same RFCH algorithm that is used for detection is used one last time on the entire fully zoomed image before running the recognition algorithm. The SIFT-based recognition is performed only if this match is above the threshold T_{near}.

F. Object search algorithm

Figure 8 illustrates the whole of the object search procedure in detail. Starting with an image at $1\times$ magnification, each object is processed independently, whereupon the resulting zoom windows are merged and each gives rise to a new, zoomed image and the procedure repeats for each of them.

The algorithm has three steps: initial, middle and final. It progresses through them according to the following:

- Initial: No magnification used. After distance estimation and zooming, proceeds to the middle step.
- Middle: Magnification given by output from zoom window sharing (Section III-D.2). If new distance estimate indicates current magnification is too small (not within $1.8\times$ of new desired middle magnification), repeats this step. If on the other hand it is within $1.2\times$ of the desired final magnification, skips straight to recognition. Otherwise, moves to final step without further zooming.
- Final: Magnification in accordance with Eq. 1. Performs recognition.

Typically, each step will run once only.

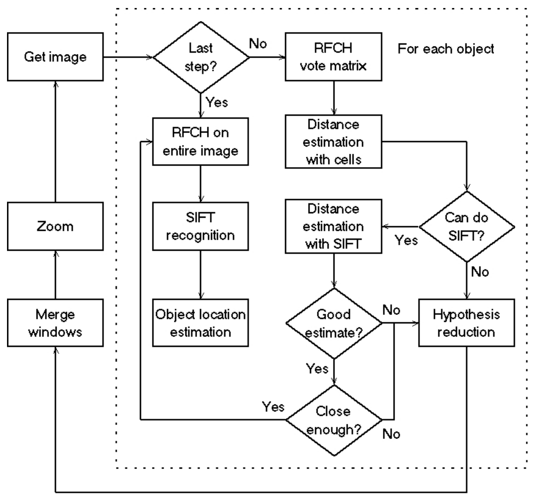


Fig. 8. Object search algorithm

In the first two steps, an RFCH vote cell grid is created and used to extract a set of hypotheses (III-B). Then, distance is estimated (III-C) using the strongest hypothesis and, if the distance found is small enough (here, requiring less than $3\times$ the current magnification), SIFT matching is performed for a more accurate distance measure. If, in turn, this indicates that the object is sufficiently magnified the algorithm skips straight to recognition; otherwise, the most reliable distance is used to produce a zoom window for the next step. Hypothesis reduction (III-D.1) prunes the result for each object; then, hypothesis sets for the different objects are merged.

The final step in the object search consists simply of recognition, wherein SIFT matching is preceded by a "sanity check" RFCH match on the entire image, as mentioned in (III-E). If the object is found, its location in space is computed from its position in the image, the distance estimate, the camera's pan/tilt angles and the robot's pose.

The output of the algorithm is a list of objects that were found in the current view and their calculated locations and distances.

IV. EXPERIMENTS

Several experiments were performed to test the algorithm proposed in this work. These experiments used a set of test objects comprising: a book, a rice carton, a printed mouse pad, a printed cup and a large robot. The sizes of the front sides of the book, the carton and the mouse pad are quite similar, about 18x20 centimetres; on the other hand, the cup is small (14x10 cm), whereas the robot is the biggest object (63x55 cm).

A. SIFT-based distance estimations

Distance estimation based on RFCH was only qualitatively tested as it is not used for localization, and its usefulness in the object search makes up part of the results presented in Section IV-B.

To test the quality of final SIFT-based distance estimation, several rounds of experiments were conducted using the book, the rice carton and the mouse pad respectively, each at two different distances. These tests consisted of extracting and matching SIFT points for images taken at three different magnification levels corresponding to those used by the visual search algorithm, resulting in different numbers of SIFT matches. Each unique test was done 7 times.

A summary of the results can be seen in Table I. This table shows the dependency of both systematic error and precision on the number of SIFT matches.

	SIFT points	Systematic error (cm)	Standard deviation (cm)
Book	> 10	0.31	10.9
	≤ 10	8.8	16.8
Mouse pad	> 10	5.0	10.1
	≤ 10	9.3	12.4
Rice	> 10	8.8	11.3
	≤ 10	14.9	12.9

TABLE I
ERROR AND STANDARD DEVIATION BY OBJECT

B. View planning and object search

The view planning algorithm was tested in three rooms, looking for several sets of our objects. Maps (including navigation graphs) were created using SLAM prior to the experiments.

Some results of these experiments can be seen in the table II, where objects *B*, *C*, *D*, *M* and *R* stand for the book, the cup, the robot, the mouse pad and the rice, respectively, and rooms *L*, *M* and *W*, for living room, meeting room and workshop. Each experiment comprises a set of objects, the room to search and the number of nodes the map of the room is divided into. The amount of nodes used and the searches performed are the result of each testcase. We can see how the number of objects involved in the exploration alters the amount of searches needed. Note that the first case requires many more searches; this is because both the cup and the robot are considered and their sizes do not allow them to be looked for in the same views. This effect is also evident when only one of them is included.

Testcase			Results	
Objects	Room	Nodes	Nodes used	Searches
BCDMR	L	8	7	18
BDMR	L	9	5	8
BCMR	L	8	4	8
	M	9	3	8
	W	4	2	6
BM	M	9	2	5
	W	4	2	3

TABLE II
VIEW PLANNING RESULTS

The previous experiments were performed using distance estimations based on visual data, and not on laser readings. That is because laser-based estimates are not as reliable as vision-based ones. Consider for example the following experiment, similar to the second testcase of Table II: the book, the rice carton, the mouse pad and the robot were placed at different positions inside the living room, as seen in Figure 9. Searching the room using laser-based distance estimations only, produces the results shown in Figure 10. Only two objects were found (one was found twice) and were located in erroneous positions. On the other hand, with estimations based on visual data, all the objects are found and are more accurately localized (Figure 11).

V. CONCLUSIONS AND FUTURE WORK

We have presented an approach to the problem of object search in a realistic environment, incorporating planning for efficient view selection and search within images using a combination of receptive field cooccurrence histograms and SIFT features. We have verified the practicability of this approach through experiments on an actual robotic system. The view planning strategy can be used even on a feature map with a clutter of overlapping features, or alternatively directly on an occupancy grid. It is easy to understand and uncomplicated to implement, because the reduction to discrete points relieves us from complex geometrical calculations. This also saves computation time, allowing it to cope well with big rooms and lots of objects,

Nevertheless, many issues remain to be solved. Using a 2D map obtained from laser scans for view planning is problematic; without very strong assumptions of spatial layout, it does not really convey a reliable picture of occlusions, nor of the probability of the occurrence of objects. It is also very sensitive to flawed room subdivision: squares belonging to neighboring rooms that may well be completely hidden will still affect the plan, leading to pointless image searches. Some sort of 3D representation, whether obtained from vision or range scans, could help in this regard.

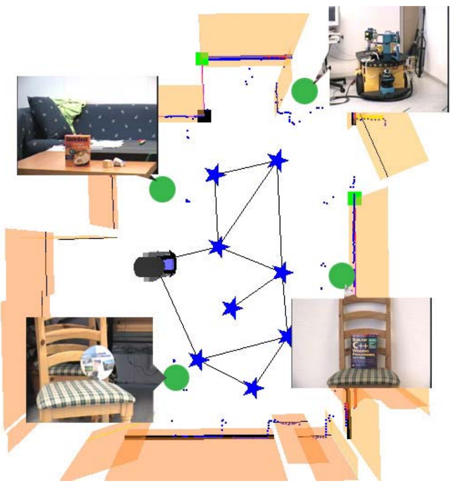


Fig. 9. Distribution of objects. Stars represent nodes; circles, the actual position of objects.

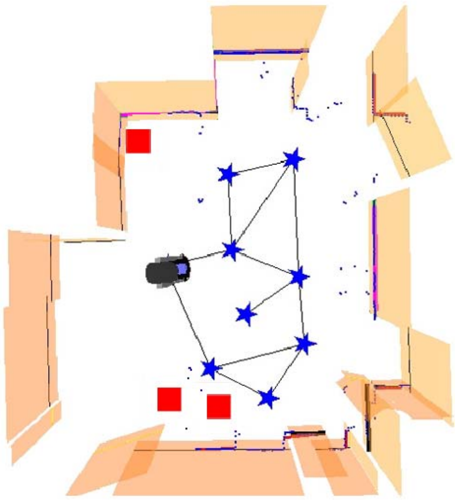


Fig. 10. Using laser data for distance. Squares represent estimated object positions.

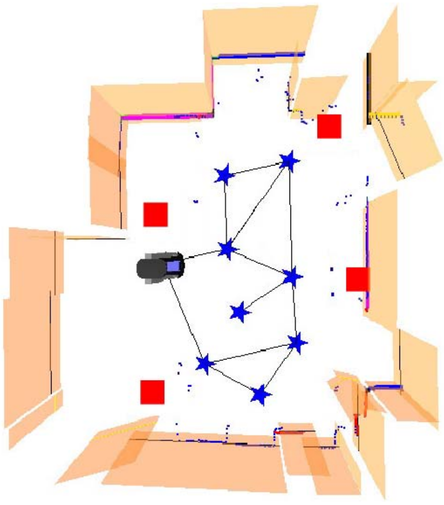


Fig. 11. Using image-based distance estimates.

Another problem is that the vision algorithm does not in its current form take into account the fact that objects may be difficult or impossible to detect or identify when seen from some angles. A simple approach involving looking at each grid square from two different vantage points was tried, but proved very inefficient and would not be guaranteed to succeed in any case. More information about each object would be required to solve this problem.

The system currently makes use of a number of thresholds and other parameters that were set through experimentation and that depend upon the objects and environment. It would be highly desirable to make the algorithm adaptive enough to eliminate the need for these parameters.

It would also be very interesting to incorporate knowledge of different objects' likely locations, especially in a semantic framework or using episodic memory.

Other promising avenues of research include: simultaneous integrated object detection and mapping, online object learning, hierarchical approaches to detection, as well as general optimization of the current approach.

REFERENCES

[1] The semantic robot vision challenge. http://www.semantic-robot-vision-challenge.org/.

[2] Staffan Ekvall, Danica Kragic, and Patric Jensfelt. Object detection and mapping for service robot tasks. *Robotica: International Journal of Information, Education and Research in Robotics and Artificial Intelligence*, 2007.

[3] John Folkesson, Patric Jensfelt, and Henrik Christensen. Vision SLAM in the measurement subspace. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'05)*, pages 30–35, April 2005.

[4] Gonzalez-Banos and Latombe. A randomized art-gallery algorithm for sensor placement. In *COMPGEOM: Annual ACM Symposium on Computational Geometry*, 2001.

[5] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell*, 20(11):1254–1259, 1998.

[6] Geert-Jan M. Kruijff, Hendrik Zender, Patric Jensfelt, and Henrik I. Christensen. Situated dialogue and spatial organization: What, where...and why? *International Journal of Advanced Robotic Systems*, 4(2), 2007.

[7] D. T. Lee and Arthur K. Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory*, 32(2):276–282, 1986.

[8] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[9] Aude Oliva, Antonio B. Torralba, Monica S. Castelhano, and John M. Henderson. Top-down control of visual attention in object detection. In *ICIP (1)*, pages 253–256, 2003.

[10] Wei pang Chin and Simeon C. Ntafos. Shortest watchman routes in simple polygons. *Discrete & Computational Geometry*, 6:9–31, 1991.

[11] T.C. Shermer. Recent results in art galleries [geometry]. In *Proceedings of the IEEE*, pages 1384–1399, 1992.

[12] Pengpeng Wang, Ramesh Krishnamurti, and Kamal Gupta. View planning problem with combined view and traveling cost. In *ICRA*, pages 711–716. IEEE, 2007.

[13] Carl-Johan Westelius. *Focus of Attention and Gaze Control for Robot Vision*. PhD thesis, Linköping University, Sweden, SE-581 83 Linköping, Sweden, 1995. Dissertation No 379, ISBN 91-7871-530-X.

[14] Y. Ye and J. K. Tsotsos. Where to look next in 3D object search. In *Symposium on Computer Vision*, pages 539–544, 1995.

[15] Yiming Ye and John K. Tsotsos. Sensor planning in 3D object search. *Computer Vision and Image Understanding*, 73(2):145–168, 1999.