Searching Objects in Known Environments: Empowering Simple Heuristic Strategies

Ramon Izquierdo-Cordova[1], Eduardo F. Morales[1], L. Enrique Sucar[1], and Rafael Murrieta-Cid[2]

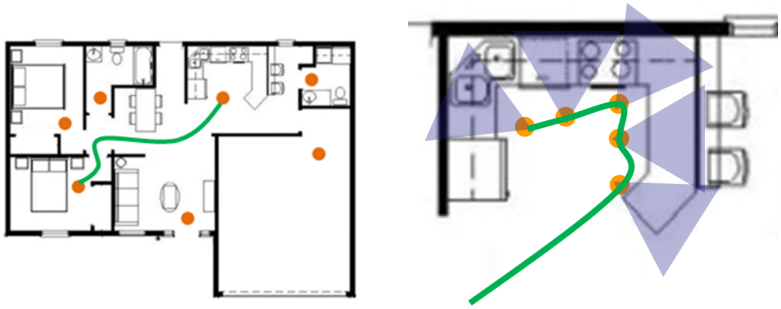
[1] Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE)
Luis Enrique Erro No. 1, Tonantzintla, C.P. 72840, Pue, México
{izquierdocr,emorales,esucar}@inaoep.mx
[2] Centro de Investigación en Matemáticas, A.C. (CIMAT)
Jalisco S/N, Col. Valenciana, C.P. 36023, Guanajuato, Gto, México
murrieta@cimat.mx

Abstract. We consider the problem of exploring a known structured environment to find an object with a mobile robot. We proposed a novel heuristic-based strategy for reducing the traveled distance by first obtaining an exploration order of the rooms in the environment and then, searching for the object in each room by positioning the robot through a set of viewpoints. For the exploration order we proposed a heuristic based on the distance from the robot to the room, the probability of finding the object therein and the room area; integrated in a $O(n^2)$ complexity greedy algorithm that selects the next room. The experimental results show an advantage of the proposed heuristic over other methods in terms of expected traveled distance, except for full search which has a complexity of $O(n!)$. For the exploration within each room, we integrate the localization of horizontal flat surfaces with the generation of poses. With the set of poses, a similar heuristic establishes the exploration order that guides the robot path inside the room. The evaluation of the set of poses shows an average coverage of the flat surfaces of more than 90% when it is configured with an overlap of 40%. Experiments were performed with a real robot using three objects in a six-room environment. The success rate for the robot finding the object is 86.6%.

Keywords: service robots, object search

1 Introduction

A desirable skill of a service robots is to assist people by fetching and carrying objects that they require, even if they do not have precise information about the place where to find the object. The robot must then make decisions about the strategy to explore the environment and find the object as quick as possible. To find an object, given a map of the environment, the robot has to decide in which order to traverse the map and how to search for objects in each place, as depicted in Figure 1.



(a) Define a searching order (b) Look inside the room

Fig. 1: To find an object the robot must first decide (a) an order to explore the rooms, and then (b) choose and visit a set of viewpoints inside the room.

We present a two-step object search strategy that uses a novel heuristic to determine the exploration order of a set of rooms and information about horizontal flat surfaces existing in the environment to complete an exploration route inside each room. The first step is based on the probability that the object is in the room, the room area, and the distance from the robot current position to each room. For the exploration in each room, it is proposed the integration of horizontal flat surfaces in the robot map for pose generation. With the set of poses, heuristics based on distances and visible areas guide the robot path inside the room. Experiments were performed with a real robot to search three different objects in a six-room environment, achieving a success rate of 86.6%.

2 Related Work

Searching objects in known environments requires, among others aspects, to define an exploration strategy, to plan and execute the routes according to the exploration strategy, and to recognize the object. Previous work addresses some of these aspects. Robot localization and object search are considered in [12], where a robot moves to unexplored 2D areas creating new nodes in a navigation graph representing the environment. The search path is calculated with a nearest-room-first strategy. With the same goal of finding the shortest path, a 3D exploration strategy is described in [11], where harmonic functions lead the robot to exploration boundaries. An alternative approach is presented in [1] where a robot explores and gathers information about the scene to determine the room in which it is located.

In [10] the authors assume that the robot has a map of the environment represented as 2D polygonal maps. To find routes that minimize the search time, the environment is decomposed into convex sections. A two-level heuristic looks, in the first level, for an order of exploration along sections based on a

uniform probability density function which characterizes the object location. In the second level, individual segments are refined using the calculus of variations. This approach is extended in [5] to 3D environments, where the search is made with a mobile robot equipped with a robotic arm of seven degrees of freedom having a limited scope camera placed at the top of its hand. A decomposition of the 3D environment is proposed in convex regions and a function to determine whether to move the robot or the arm. An alternative exploration strategy based on three steps is proposed in [4]: represent the workspace using Minkowsky sums, find a set of locations to cover as much as possible the environment, and find a route that passes through the points and minimizes the traveled distance. In [2], the authors base their strategy on finding a set of relations between objects that minimizes the expected search cost considering probability and distance. The work in [8] selects a group of locations where the object is supposed to be, based on probabilities obtained from the Open Mind Indoor Common Sense (OMICS) database. An exploration order of all rooms in the group is obtained based on Euclidean distances. A planner-independent formulation is presented in [13], in which the object search problem knowing the probability of the object being in a determined place is defined formally as a Stochastic Shortest Path. Three different algorithms to find the shortest route are proposed.

This work focuses on finding an exploration order in a known domestic environment given a probability distribution for the position of the object over the rooms, and the subsequent problem of finding a set of poses inside each room and a route for visiting them. In contrast to previous work, we propose a very efficient and simple heuristic to determine the room exploration order, which gives results close to the optimal solution. Additionally, we integrated the room exploration technique and the object search approach inside each room, and implemented the complete method in a real mobile robot.

3 Determining the Exploration Order

When exploring a set of rooms for searching an object several elements can be considered. If we want to minimize time, we could go to the closest places first. If we have information that the object has a good chance of being in a certain room, we could visit that room first as it gives us a higher chance of success. An additional element to take into account is the area of the room, since exploring a large room takes longer than a small one. Our proposed heuristic function seeks to balance the search criteria leveraging the strengths of simple strategies.

3.1 Heuristic Approach

Given a map of the environment with rooms on the map, $R_1, R_2, ..., R_n$, previously tagged by a user, our problem consists in finding an order to visit the n rooms, to minimize the expected time to find the object. Assuming that the robot moves at a constant velocity, this is equivalent to minimizing the expected traveled distance. We assume the robot can obtain $P_1, P_2, ..., P_n$ probabilities of

finding the object in the locations; has information of the area of each room, $A_1, A_2, ..., A_n$; and can compute $d_1, d_2, ..., d_n$, the respective distances from the current position of the robot to each of the rooms. We define the proposed heuristic function as:

$$H(R_i) = \frac{P_i}{d_i \sqrt{A_i}} \tag{1}$$

Since the value of the heuristic is only used to select which room will be explored next, the area and distance values do not need to be normalized. Finally, to obtain the full order we use the greedy *BestLocalRatio* algorithm based on the previous heuristic (see Algorithm 1), where the best room to search is selected using Eq. 1, and from that room the next best room is selected. The process continue until all the rooms have been selected.

Algorithm 1 Best Local Ratio

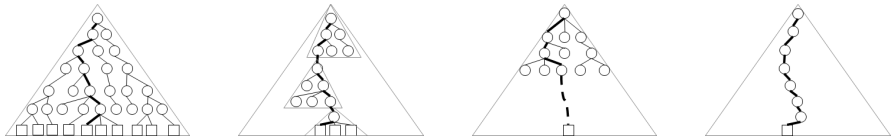
Require: *RoomList*
Ensure: *ExploringOrderList*
1: *ExploringOrderList* ← *empty*
2: **repeat**
3: **for all** *Room* in *RoomList* and not in *ExploringOrderList* **do**
4: Compute heuristic in formula (1)
5: **end for**
6: Select $Room_i$ with the maximum heuristic value
7: Add *ExploringOrderList* ← $Room_i$
8: **until** *RoomList* = *ExploringOrderList*
9: **return** *ExploringOrderList*

3.2 Alternative Strategies

Determining the exploration order with the minimum expected time (distance) is NP-Hard [10], so several authors have proposed finding a solution through heuristics. In this work we have selected four alternative approaches, illustrated in Figure 2. Five strategies are compared against the proposed heuristics in our experiments. Two use simple heuristics: nearest room and most probable room, and the other three try to minimize the expected distance traveled by the robot:

$$E(d) = \sum_{i=1}^{n} \left(D_i + \sqrt{A_i} \right) P_i, \tag{2}$$

where n is the number of rooms or nodes, P_i is the probability of finding the object in room i and D_i is the accumulated distance from the starting point of the robot to room i following the route through all rooms $j (j < i)$. i and j are room indexes according to the exploration order being evaluated. This equation is based on [10], adding the term $\sqrt{A_i}$ as an estimate to take into account the distance that the robot has to travel inside each room.



(a) (b) (c) (d)

Fig. 2: A qualitative illustration of the exploration trees for the different strategies: (a) exhaustive search, (b) dominant strategies heuristic, (c) Monte-Carlo based methods, (d) greedy algorithms.

Exhaustive Search. A complete search in all possible orders is performed using the algorithm described in [6]. The route with the smallest expected value of distance of all possible paths that go through all rooms is selected. This method has a computational complexity $O(n!)$. Even using linear programming methods, the best existing algorithm has a complexity $O\left(n^2 2^n\right)$ [3].

Dominating Strategies Heuristic. For constructing the full path the room with the highest value using P_i/d_i is selected, where P_i is the probability of finding the object in room i, and d_i is the distance. After selecting the room with the highest value, a breath-first exploration is performed from that node until it reaches a depth of $\log n$ nodes, avoiding expanding those nodes that are not strictly dominant. The partial route with the shortest expected value of distance according to Eq. 2 is selected. This procedure is repeated until a full path is completed [10]. It has a computational complexity of $O\left(n^3 logn\right)$.

Monte-Carlo based Search. An exploration of the search tree is made until it reaches a predefined depth. In the experiments a depth of 10 nodes was used. From the leaf nodes in the expanded subtree, complete routes are constructed by randomly choosing nodes. These routes are used to evaluate the expected distance normalizing its value to $[0, 1]$ with the formula $1/(1 + E(d))$. Generating several random routes for each leaf node gives an estimation of the quality of the sub-route from the starting node to that leaf node. For determining the end node of the subtree to complete the random route it is used the Upper Confidence Bound for Trees (UCT) [7], that consists of evaluating from the root node in the exploration tree, which one is the best child.

Nearest and Most Probable Room. These two strategies use the greedy algorithm so they keep a computational complexity of $O\left(n^2\right)$. The nearest room strategy selects the closest room, while the most probable room strategy selects the most likely place. The results show that despite being extremely simple strategies, they get excellent results in special conditions.

4 Exploration inside the Room

Getting the minimum set of poses to cover a polygon when the robot has limited visibility is a complex problem [9]. In order to reduce complexity, we have used a method based on randomly generating a set of N poses. Once we have the

random set, each pose is evaluated to select those ones which observe the largest portion of the flat surfaces in the room (see Algorithm 2).

Algorithm 2 Pose Generator

Require: N = Number of randomly poses to generate
Ensure: *PosesList*
1: Randomly generate N poses
2: Delete unreachable poses
3: **for all** pose in *PosesList* **do**
4: Compute *Visible Area*
5: **end for**
6: Delete pose with no *Visible Area*
7: **for all** pair of poses in *PosesList* **do**
8: **if** pair is *nearby* **then**
9: Delete pose in pair with less *Visible Area*
10: **end if**
11: **end for**
12: **for all** pair of poses in *PosesList* **do**
13: **if** pair is *redundant* **then**
14: Delete pose in pair with less *Visible Area*
15: **end if**
16: **end for**
17: **return** *PosesList*

Unlike related work, our algorithm observes a surface from different viewpoints but not looking for complete reconstruction. For object search it is only important to see if the object can be recognized. We have a configurable control to re-observe flat surfaces as much as it is required depending on the conditions of the environment. The clutter condition can be determined based on the number and spatial distribution of objects on flat surfaces.

Once the robot is inside a room, the pose generation algorithm determines a set of poses for the robot for looking for the object (assumed to be over a flat surface), as well as a route to visit all the selected poses, or until the object is found. We use: (i) the *field of view (FOV)*, the maximum angle at which the sensor is capable of perceiving the observable world and (ii) the *depth of field (DOF)*, the space in front of the focus plane, between the first and last points acceptably sharp. When we intersect FOV with DOF we obtain what is known as *sensor visibility cone*. Since in this work the object search is supposed to be in a two-dimensional space, the visibility cone will be framed by a circular trapeze, but to facilitate geometric calculations we use a simple trapeze.

The pose generator algorithm randomly generates poses and then eliminates those that can not be reached by the robot. The next step eliminates poses that do have a visible area in flat surfaces. The next filter compares each pair of poses and removes one of each pair considered as *nearby* poses. The pose of the pair that is kept is that one with the largest visible area. If the angle and the distance

differences between two pairs of poses are lower than the given thresholds U_α and U_d, then they are considered *nearby*. A third filter eliminates *redundant* poses. Two poses are considered *redundant* if the intersection of their observed areas is larger than a proportion of the visibility cone of the object recognizer. This ratio, which we call overlap threshold, is a given value U_t, $0 \leq U_t \leq 1$. Finally, the algorithm delivers a set of poses that are reachable by the robot and observe the largest proportion of flat surfaces existing in the environment while the amount of overlap over the visibility area of different poses is controlled.

Once we have a set of poses to explore the environment, we must now decide how to go through them in order to find the object as fast as possible. The problem is similar to the traveling salesman problem, so we propose and compare three heuristic strategies to visit the set of poses:

Nearest Pose (NP). Assuming that the robot moves at a constant speed, the entire route is planned iteratively selecting the pose that has the shortest distance from the current one.

Largest Visible Surface Pose (LVSP). The idea behind this strategy is that a flat surface with the largest area is more likely to contain the searched object. The set of poses is grouped according to the flat surface they observe and each group is then sorted according the nearest pose. The group of poses with observations to the largest flat surface are visited first.

In Room Best Local Ratio (IRBLR). We use a similar strategy to *BestLocalRatio* that considers distances and visible areas. Since we assume equal probabilities to find the object in each flat surface, our heuristic takes the form: $H(R_i) = \frac{1}{d_i\sqrt{A_i}}$, being d_i the distance from the last pose to the new one, and A_i the visible area of pose i.

5 Experiments

We performed experiments to evaluate the performance of the proposed algorithms and show experimental tests with a real robot.

5.1 Room Order

To evaluate the strategies for the exploration order of rooms, a database of maps was created from real home plans divided in categories according to the number of rooms in each map. The selection includes small homes with three rooms to big residences with thirteen rooms. For each category, five different maps are considered. In total 55 home plans were tested (Figure 3). Each map is labeled manually by the user, selecting a point within each room, and then automatically creating a complete graph with distances between each of the rooms.

Objects in real world have almost as many probability distributions as the number of them. Instead of selecting specific objects for testing, we chose several probability distributions to evaluate the exploration strategies and see how these differences affect the results. The distributions considered are: a uniform

Strategy	**Average Distance wrt Optimum**	**Std. Dev.**
Exhaustive Search	1.0000	0.0000
Best Local Ratio	1.0281	0.0222
Monte-Carlo	1.0621	0.0900
Dominant Strategies	1.0845	0.0678
Most Probable	1.1523	0.1030
Nearest	1.2506	0.1188

Table 1: Comparison of evaluated methods to generate a room exploration sequence. Distances are shown as the proportion of the optimal distance. Averages are calculated considering 3-13 rooms with four probability distributions.

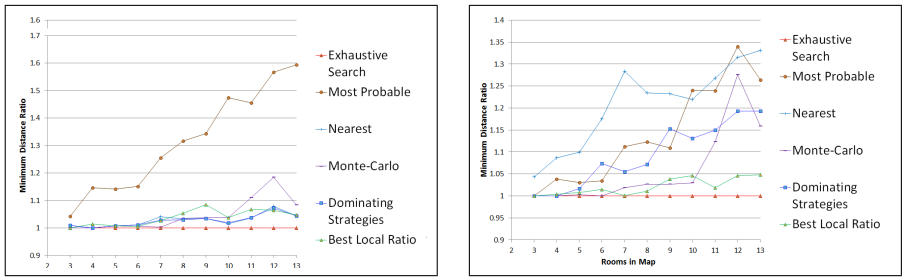
distribution; a normal distribution ($\mu = 0.5$, $\sigma^2 = 1.0$); a gamma probability distribution ($\kappa = 2.0$, $\theta = 0.2$),; and an exponential distribution ($\lambda = 6.5$).

Although the objective is to minimize the expected time to find the object, we use instead the expected distance, assuming a constant speed for the robot. We decided to use as measure of distance the ratio over the optimal expected distance (Eq. 2) obtained with the exhaustive search strategy. This measure allows us to compare between maps with different distances between their rooms and even between maps with different numbers of rooms. Our simulation experiments were evaluated computing values of the expected distance as described in Eq. (2). All evaluated strategies compute only orders, and paths are computed by the same planning algorithm for all strategies.

Figure 4 illustrates the average expected distance for maps containing from 3 to 13 rooms, five maps for each size, for each one of the probability distributions used for object locations. The nearest room strategy produces values of expected distance close to the optimal values when the probabilities are equal between rooms (uniform distribution) but it has the worst results when the difference between probabilities is large (exponential distribution). The opposite case is presented with the most probable strategy. However, the proposed strategy being a combination of these two simple heuristics, produces expected distances close to the optimal values for all the distributions.

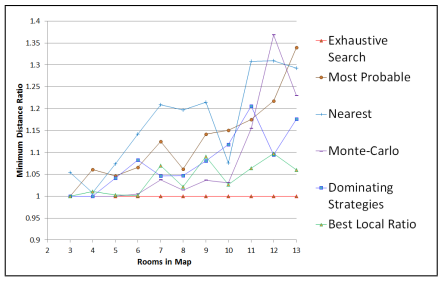
Table 1 shows average expected distances for all the map sizes and probability distributions. The proposed algorithm produces the smallest average distances after exhaustive search with the lowest variability. Also, it is among the most efficient algorithms.

Fig. 3: Some of the maps tested in the experiments.

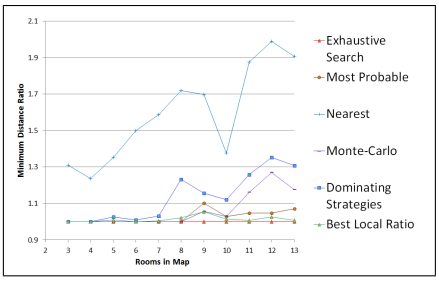


(a) Uniform

(b) Normal



(c) Gamma



(d) Exponential

Fig. 4: Comparison of exploration strategies for real maps between 3 and 13 rooms using as reference the optimal expected distance. The probability of finding the object follows different distributions over the rooms.

5.2 Pose Generation and Exploration

We evaluated the quality of the set of generated poses considering the percentage of coverage of existing flat surfaces in the map and the percentage of the total visible area observed from more than one pose. We use a fixed number of poses (1000) because the rooms have similar areas, but it is possible to take into account the area of each room. We varied the overlap threshold from 0 to 1 in 0.2 increments and tested pose generation in seven different rooms. Results are summarized in Table 2. As expected, there is a relation between coverage and overlap. With a robust object recognizer and/or an almost-free-occlusion environment, it is desirable to have low overlap, with a threshold between 0.2 and 0.4, and a larger value in scenarios with more occlusions.

We compare the three strategies (NP, LVSP and IRBLR) to visit the set of poses. The evaluation metric used is the expected distance to be traveled by the robot. We assume an uniform distribution about the object location over every flat surface. Then the likelihood of a pose is given by the amount of visible area from that pose. The metric that evaluates the path over n poses is defined as:

$$E(d) = \sum_{i=1}^{n} (D_i P_i), \tag{3}$$

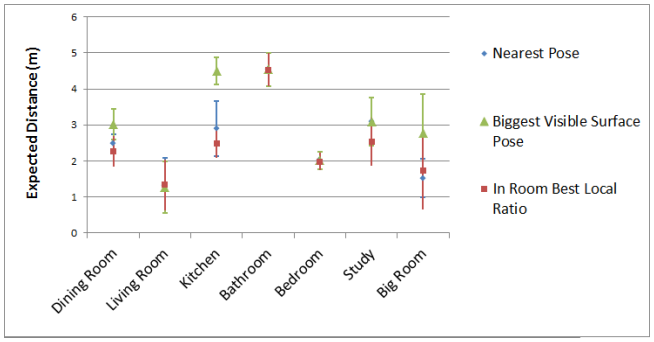


Fig. 5: Comparison of expected distances produced by the strategies to visit a set of poses that search for an object within a room.

where D_i is the accumulated distance from the starting position of the robot to the pose i and P_i is the value of the visible area of the pose i normalized by the total area of all existing flat surfaces in the room.

Finding distances for all pairs of the complete set of poses has a high computational cost, so we use the Euclidean distance to speed up the process of generating paths. We generate five sets of different poses for each of the seven rooms in the environment. Some rooms contain one or two flat surfaces while the large room, which is the union of the other six, contains nine. For each strategy a path for each of the sets of poses is created and evaluated with the expected distance, Eq. 3. The results are shown in Figure 5. In the rooms *bathroom*, *bedroom* and *study* there is just one flat surface, so that strategies do not differ in their traveled distances, while in the rest of the rooms with at least two flat surfaces, a slight advantage for the IRBLR method can be observed.

	Coverage (%)		**Overlap (%)**	
	Average	**Std. Dev.**	**Average**	**Std. Dev.**
0.0	61.44	30.57	0.00	0.00
0.2	86.49	15.47	1.64	3.05
0.4	91.99	8.68	9.88	11.52
0.6	91.29	12.45	55.81	25.87
0.8	93.27	11.51	96.32	43.21
1.0	96.59	4.07	101.25	39.70

Table 2: Evaluation of the set of poses generated with Algorithm 2 in terms of percentages of coverage and overlap of the total flat surfaces when we change the overlap threshold.

5.3 Real mobile robot

We conducted experiments with a real robot searching for three different objects: *coke can, remote control* and *mug* in an environment similar to an apartment with 6 rooms. The room in which the objects were located was randomly generated following certain probability distribution for each object, and the flat surface for the object was selected according to each surface area. We located the object in the flat surface with a random pose using a uniform distribution. The robot started from a fixed initial position. We used a differential two wheeled Peoplebot platform equipped with a laser for navigation and a Kinect sensor for object recognition. As recognition algorithm we used *Tabletop* [14] configured with default parameters due to its good recognition performance, but specially for its capacity to recognize flat surfaces.

Table 3 summarizes the results. The robot has a success rate for the three objects of 86.6%. Although the average search times may seem high for the size of the environment, it should be noted that in the case of the *coke can* and *remote control* it includes the cases where the objects were not found and the whole path was traveled. In general, the locations of objects are concentrated in the first two most probable rooms according to the probability distribution.

	Coke Can		**Mug**		**Remote Ctrl**		**No Object**	
Success/Trials	4/5		5/5		4/5		5	
	avg	**stdev**	**avg**	**stdev**	**avg**	**stdev**	**avg**	**stdev**
Time (min)	6.69	5.03	2.41	1.73	5.31	4.20	11.696	0.652
Distance (m)	22.59	16.03	7.85	5.61	20.85	14.91	39.549	3.023
Turns (degrees)	5251.00	3688.16	1809.00	1198.94	3722.40	3006.01	9155.222	757.460
Planned poses	16.40	9.81	8.40	4.62	12.00	10.07	27.333	1.581
Reached poses	13.00	10.65	4.20	3.42	9.20	7.36	23.889	2.421
Visited rooms	3.40	2.41	1.60	0.55	2.40	1.52	6.00	0.00

Table 3: Results from five searches with three objects located at different positions in each repetition. *No Object* corresponds to the whole path of the six rooms and is presented for comparison.

6 Conclusions

We proposed a novel heuristic-based strategy for a mobile robot to search for an object in a known environment based on a two-step approach. Firstly, obtaining an exploration order of the rooms in the environment based on a novel heuristic which takes into account the probability, the distance and the area of the rooms. And secondly, searching for the object in each room by randomly generating poses that are filtered and that observe flat surfaces in the environment. Additionally, we propose a heuristic method for visiting the set of poses based on distance and visible area. We implemented the proposed methods in

a real robot and evaluated the integrated approach in a realistic environment. The experimental results with the real robot give evidence that the proposed approach can be used in object search applications with service robots. As future work we will develop strategies to determine the probabilities of different objects being in certain room in a home environment based on the Web and incorporate information of where objects were previously found using a Bayesian approach.

References

1. Aydemir, A., Pronobis, A., Gobelbecker, M., Jensfelt, P.: Active visual object search in unknown environments using uncertain semantics. IEEE Transactions on Robotics 29(4), 986–1002 (2013)
2. Aydemir, A., Sjöö, K., Folkesson, J., Pronobis, A., Jensfelt, P.: Search in the real world: Active visual object search based on spatial relations. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 2818–2824. IEEE (2011)
3. Bellman, R.: Dynamic programming treatment of the travelling salesman problem. Journal of the ACM (JACM) 9(1), 61–63 (1962)
4. Cabanillas, J., Morales, E.F., Sucar, L.E.: An efficient strategy for fast object search considering the robot's perceptual limitations. In: Advances in Artificial Intelligence, vol. 6433, pp. 552–561. Springer (2010)
5. Espinoza, J., Sarmiento, A., Murrieta-Cid, R., Hutchinson, S.: Motion planning strategy for finding an object with a mobile manipulator in three-dimensional environments. Advanced Robotics 25, 1627–1650 (2011)
6. Heap, B.: Permutations by interchanges. The Computer Journal 6(3), 293–298 (1963)
7. Kocsis, L., Szepesvári, C.: Bandit based monte-carlo planning. In: Machine Learning (ECML), pp. 282–293. Springer (2006)
8. Kunze, L., Beetz, M., Saito, M., Azuma, H., Okada, K., Inaba, M.: Searching objects in large-scale indoor environments: A decision-theoretic approach. In: IEEE Intl. Conf. on Robotics and Automation (ICRA). pp. 4385–4390 (2012)
9. Rourke, J.O., Supowit, K., et al.: Some np-hard polygon decomposition problems. Information Theory, IEEE Transactions on 29(2), 181–190 (1983)
10. Sarmiento, A., Murrieta-Cid, R., Hutchinson, S.: An efficient motion strategy to compute expected-time locally optimal continuous search paths in known environments. Advanced Robotics 23, 552–561 (2009)
11. Shade, R., Newman, P.: Choosing where to go: Complete 3d exploration with stereo. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 2806–2811. IEEE (2011)
12. Sjö, K., Gálvez-López, D., Paul, C., Jensfelt, P., Kragic, D.: Object search and localization for an indoor mobile robot. Journal of Computing and Information Technology 17, 67–80 (2009)
13. Trevizan, F., Veloso, M.: Finding objects through stochastic shortest path problems. In: Proceedings of the International Conference on Autonomous Agents and Multi-agent Systems. pp. 547–554. International Foundation for Autonomous Agents and Multiagent Systems (2013)
14. Willow Garage and ROS Community: Ork - Object Recognition Kitchen (2011), `https://github.com/wg-perception/object_recognition_core`