FLAP for CAOS: Forward-Looking Active Perception for Clutter-Aware Object Search [1]

Thorsten Gedicke, Martin Günther and Joachim Hertzberg

{tgedicke, martin.guenther, joachim.hertzberg}@uos.de

Abstract: In this paper, we present a system for autonomous object search and exploration in cluttered environments. The system shortens the average time needed to complete search tasks by continually planning multiple perception actions ahead of time using probabilistic prior knowledge. Useful sensing actions are found using a frontier-based view sampling technique in a continuously built 3D map. We demonstrate the system on real hardware, investigate the planner's performance in three experiments in simulation, and show that our approach achieves shorter overall run times of search tasks compared to a greedy strategy.

Keywords: autonomous mobile robots, heuristic searches, knowledge acquisition, path planning, probabilistic models, robot vision, sampling actions, searching systems

1. INTRODUCTION

Mobile service robots need to operate in dynamic environments that are subject to change not only as a result of the robot's own actions but often due to independent change. The robot's knowledge about the physical state of such an environment may get wrong and incomplete. As an example of such a domain, consider a robot waiter in a restaurant. Since there are other waiters and guests involved, who interact with the environment and introduce changes without announcing these changes to the robot, it can never fully rely on the internally presumed state of the world, especially regarding the whereabouts of task-relevant objects. Instead, it needs to actively incorporate explicit sensing actions into its plans to search for objects or to verify beliefs about the current state of the world. View obstructions due to clutter must be accounted for and sensing actions at a variety of locations may be necessary. Systems that reason about such *active perception* actions to enhance their operational capability and flexibility in changing or unknown environments comprise several interrelated aspects and have been studied in diverse contexts.

While our work focuses on bringing the robot's sensor into configurations desired for data acquisition in a collision-free way, interactive perception actions that move occluding objects out of the way have also been considered in recent literature (Dogar et al., 2014; Gupta et al., 2013; Wong et al., 2013). Collision-free active perception systems typically determine a *next best view*, i.e., a sensor configuration that the robot should assume next to optimize a utility function that incorporates the expected information gain and often also predicted costs, which typically amount to the action's execution time. Next best view algorithms can be seen as a variation of Connolly's (1985) "Planetarium Algorithm", which compares simulated range images to successively select the view pose with the largest area of yet-unknown space in the image. While the objective of exploration is different from object search, it can be argued that a general object search system covers the use case of exploration in the special case that it terminates only after exhausting the search space and is not guided by prior knowledge regarding some target object.

We present a system for object search and exploration in cluttered environments that aims to minimize expected total task execution times using a continual planning method that exploits partial knowledge of the environment and a probabilistic model of the target object's location. The software is implemented using the ROS framework and designed as an active perception module for the artificial cognitive system RACE (Hertzberg et al., 2014). It is available as open source at `https://github.com/uos/uos_active_perception`.

After reviewing related work in Sec. 2, we present our approach in Sec. 3. We show experimental results in Sec. 4 and conclude with a discussion of future work in Sec. 5.

2. RELATED WORK

Establishing a prior hypothesis regarding probable locations of the target object allows to direct the search process towards such locations and enables the system to work in large environments where uninformed search would be infeasible. Ye and Tsotsos (1999) [also Shubina and Tsotsos (2010)] maintain a probability distribution function for a single target object that assigns to each cell in a discretized workspace the degree of belief of the target being located there. Approaches that have been examined to obtain the required world knowledge include semantic probabilistic environment models (Kunze et al., 2012), visual saliency (Rasouli and Tsotsos, 2014), as well as relational models of

[1] All authors are with the Institute of Computer Science, University of Osnabrück, Albrechtstr. 28, 49076 Osnabrück, Germany. Martin Günther and Joachim Hertzberg have a second affiliation with the DFKI Robotics Innovation Center, Osnabrück Branch, 49076 Osnabrück, Germany. This work is supported by the RACE project, grant agreement no. 287752, funded by the EC Seventh Framework Programme theme FP7-ICT-2011-7.

object classes (Wong et al., 2013), affordances (Moldovan and Raedt, 2014), and spatial relations (Kunze et al., 2014; Anand et al., 2011).

Once a region of interest has been identified, the robot needs to select a next best view, i.e., a sensor configuration that brings that region into sight, considering criteria that usually include visibility of unexplored space and sensing costs, e.g., the time needed to travel to the sensing pose. Ye and Tsotsos (1999) and Shubina and Tsotsos (2010) decompose view selection into robot relocation and choice of sensor configuration. Sensor configurations are selected iteratively at each location based on their expected probability to detect the target until travel to a new location becomes necessary. Yamauchi (1997) proposes to guide autonomous exploration in 2D by targeting *frontiers*, i.e., boundaries between free space around the robot and unknown space beyond, a concept which Surmann et al. (2003) extend to autonomous mapping in 3D by sampling potential sensing poses in a 2D projection of 3D space. Blodow et al. (2011) evaluate frontiers in full 3D only for promising view poses candidates previously identified in a 2D projection of the map. Dornhege and Kleiner's (2013) frontier-void-based approach explicitly associates clusters of unknown map cells with neighboring clusters of frontier cells to identify high visibility view poses with six degrees of freedom. The original deterministic form of this approach is modified in later research by Dornhege et al. (2015) to employ sampling. While these systems operate in a cycle of view selection and sensor relocation, Shade and Newman (2011) present an approach that merges view selection and path planning to generate smooth exploration trajectories towards frontiers using a potential field method inspired by fluid dynamics.

The above cited systems, with the exception of Dornhege et al. (2015), select views using a greedy strategy. Computing an optimal search/exploration strategy is NP-hard and related to the set cover, art gallery, and traveling salesman problems (Ye and Tsotsos, 1999; Sarmiento et al., 2003). Planning multiple steps ahead is necessary in certain applications without regard for optimality, e.g., to avoid traversing unknown space to scan a target (Renton et al., 1999) or to expose unknown space behind multiple layers of concealment (Gupta et al., 2013). Several approximate methods to generate more efficient plans have been examined that rely on divide and conquer (Dogar et al., 2014), pruning heuristics (Sarmiento et al., 2003), and a set cover/TSP decomposition of the problem (Dornhege et al., 2015).

Planning systems for object search have been shown in experiments to provide useful results without prohibitive computational cost (Sarmiento et al., 2003; Dornhege et al., 2015). However, the advantage of planning compared to a greedy approach is outweighed by computational cost when a certain amount of replanning is considered (Dornhege et al., 2015). Since search/exploration systems are likely to encounter circumstances that deviate from the presumed world state, it is desirable to lower the computational burden and make continual planning feasible.

Since it does not introduce any restrictions on sensing poses and allows dynamic scaling of view pose density, sampling seems to be a favorable approach to generate sensing pose candidates that has already been used successfully for forward-looking search (Dornhege et al., 2015). The process can be guided towards promising locations using the concept of frontiers (Blodow et al., 2011).

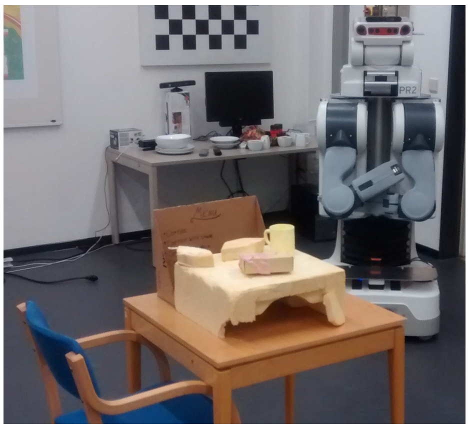
Fig. 1. PR2 during a search process. The target volume on top of the table has several occlusions and a cavity.

This paper contributes three innovations for mobile object search and exploration. First, we present an efficient anytime system for 3D view computation that uses frontier-based sampling to find next best view candidates. Second, we develop an object search strategy that plans multiple sensing actions ahead of time to minimize the expected search time using a predictive model of robot movement speed and a probabilistic model of target object locations. Third, we show the performance of the search planner to be suitable for offline as well as continual planning and demonstrate the complete system running in the real world.

3. APPROACH

Our system searches for an object of a target class (e.g., a mug) within a given target region. It is designed as a subcomponent of a cognitive architecture that is capable of recognizing individual objects when they are detected by the sensor and maintains a belief state regarding probable object locations. When the object search component is triggered by the system, it is given a set of bounding volumes (e.g., a box that encompasses the top of the table shown in Fig. 1) and the degree of belief for each of these volumes to contain an object of the target class. The actions available to the robot are navigating to a new position (using a path planner), pointing the head, and raising or lowering the telescoping torso, and any combination of these may be used to transition between view poses. We rely on the system to terminate the search process when the target is detected within the sensor's field of view.

Fig. 2 shows an overview of the data flow between the involved components and highlights the active perception components that are subject of this paper in a darker shade. A 3D map is continually built from sensor data and is used by a view sampling module, which computes a set of possible view poses along with the expected information gain for each pose. These samples are per request fed to an object search planning and execution module, which aims to minimize the expected time until the target is found and iterates between requesting view samples, planning, moving the robot, and sensing.

3.1 Mapping and View Sampling

The system maintains an octree representation of the robot's workspace with cells that take values of either free, occupied, or unknown. The OctoMap library (Hornung et al., 2013) provides a mapping framework that allows to efficiently query the map and to integrate point clouds using a probabilistic sensor model. We maintain a set of frontier cells that consists of all unknown cells with at least one known free neighbor cell, akin to Blodow and colleagues' (2011) "fringe voxels". The map is built incrementally by continual integration of preprocessed sensor data.

Arbitrary regions of the map can be reset to unknown to express that the region is assumed to have changed and should be observed anew. This can be triggered manually or by external modules that use semantic knowledge to deduce that the information regarding a certain region is out of date based on the observation of some situation or activity. Future work may implement a version of fading knowledge that gradually decreases confidence in individual cells when they have not been observed for a certain amount of time.

The map constitutes the basis of the view sampling process that determines and evaluates possible sensor poses for the observation of unknown cells within some region of interest. View sampling is an anytime operation that allows to adjust the amount of computational effort invested on the fly and, given enough time, will find all feasible view poses. The sampling system's output consists of 2-tuples, wherein the first element is a sensor pose and the second element is a set of unknown cells that are expected to be visible from this pose. In the following, such pairs shall be called *views*.

A query to the sampling system consists of a region of interest (ROI), defined as a set of bounding volumes $\mathcal{V}$. For each volume $V_i \in \mathcal{V}$, the set of all frontier cells inside V_i is gathered. Unknown cells at the boundary of V_i that are not marked as frontiers are additionally included as frontiers for this volume only to enable the system to find views for regions even if they are enclosed in unknown space. The union of all collected frontier cells within the ROI is in the following denoted as $\mathcal{F}$.

The position of each view sample is picked from a constrained space around a randomly chosen target frontier cell $f \in \mathcal{F}$. All views are oriented to point directly towards their respective target cell, whereby the position remains to be chosen randomly without violating any of three general constraints. The first constraint simply demands the position to be within the sensor's range limits relative to the target cell. Secondly, the sensor pose must be attainable according to the robot's kinematics; for example, the sensor's height is always limited, if not fixed, and also the pitch angle may be variable within limits. Finally, all potential camera poses should be located at the known-space side of the frontier, looking towards the ROI. This constraint is enforced by estimating the frontier plane's normal vector at f and restricting the angle between the line of sight and the frontier normal to be less than 90°. The frontier normal of a target cell is calculated by connecting the mean of the center points of all unknown neighbors with the mean of the center points of all free neighbors. While a straightforward implementation could simply pick random coordinates and reject those samples which lie outside the constrained region, we reduce the amount of rejects significantly using a local consistency approach in the above outlined constraint network.

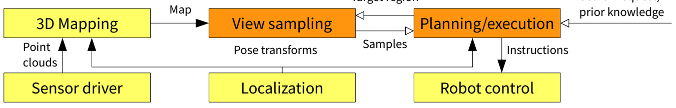

Fig. 2. Structural overview of inter-component data flow.

To determine the set of cells that are expected to be revealed by the sensor at a sample pose, we scan the field of view of a virtual camera and compile the set of all unknown cells within the ROI that are traversed by rays. When a ray hits an unknown cell, it is allowed to pass through; hence, view sampling assumes unknown space to be free and is, in this sense, optimistic.

The sampling procedure loops until either the required number of views is generated or a timeout is reached. See Fig. 3 for a visualization of the generated view samples.

3.2 Object Search

The object search component acts on search tasks that define a target object and a search region and runs until either the target is located or the search region is exhausted. A planner constructs sequences of views with the aim to minimize the expected duration of a search process that successively visits all views in the sequence.

The search region defined in each task is represented as a set of bounding volumes $\mathcal{V}$. Each volume $V_i \in \mathcal{V}$ is annotated with a probability P_i of the volume containing *at least one* entity of the target type according to the robot's current state of belief. Since all P_i are independent of each other, it is not necessary that $\sum P_i = 1$. Each cell $c_k \in V_i$ is assumed to equally and independently contribute to P_i, resulting in a probability value for each cell of

$$\forall c_k \in V_i : p(c_k) = \begin{cases} 1 - \sqrt[N_i]{1 - P_i} & \text{if } \mathit{unknown}(c_k) \\ 0 & \text{else}\,, \end{cases} \tag{1}$$

where N_i is the number of cells in V_i. Comparable object search systems, such as those of Ye and Tsotsos (1999) and Shubina and Tsotsos (2010), model the probability distribution of a single target object's location. In such a model, the sum of all location probabilities necessarily equals 1, and all probabilities must be updated accordingly when a set of cells is seen without locating the target. Because our approach makes no assumption about the number of target objects, all P_i and all $p(c_k)$ can be considered independent and no update step is necessary.

View sampling is called for the region of interest to generate a set of n views $\mathcal{S}$. The returned views include only unknown cells, so that the total detectable space according to the view set is

$$\hat{V}(\mathcal{S}) = \bigcup_{s_i \in \mathcal{S}} \mathit{cells}(s_i) \subseteq \mathcal{V}\,, \tag{2}$$

where $\mathit{cells}(s_i)$ yields the set of all cells expected to be seen from the view pose s_i. We disregard the reliability of detection algorithms and assume that if a target is in view of the sensor, it will be detected every time. The

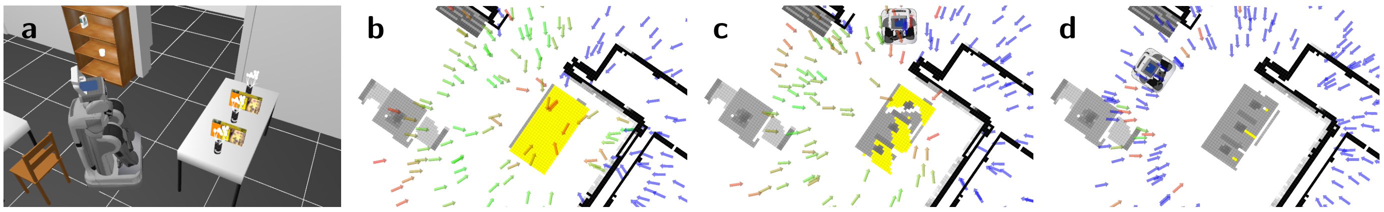

Fig. 3. View sampling during different stages of object search. Views are visualized as arrows with color-coded normalized target space visibility (blue for zero visibility). Unknown cells within the ROI on the table are marked yellow. From left to right: (a) Overview of the setting. (b) Initial samples. (c) Unknown cells after partially observing the ROI. The unknown cells closest to the robot are below the sensor's field of vision. (d) After observation from a second angle, few occluded cells remain. Note: We show an orthographic projection; all parts of the algorithm work in 3D.

probability of detecting at least one instance of the target type by moving the sensor to a view pose s_i can be defined in terms of the set of cells seen $C = \mathit{cells}(s_i)$ as

$$p(C) = 1 - \prod_{c_i \in C} (1 - p(c_i)) \,. \qquad (3)$$

The total expected probability to see a target using all views is simply $p(\hat{V}(\mathcal{S}))$. The object search task contains a termination criterion in the form of a threshold probability ω. The search process terminates when $p(\hat{V}(\mathcal{S})) < \omega$.

An external path planner is used to generate paths from the current robot location to all view locations. All unreachable views are filtered out. Subsequently, predicted transition times $t(s_i, s_j)$ between the remaining views are calculated. Since these times are needed in every step of the planning process, a lot of time is saved by pre-computing them as a lookup table. Further time is saved by clustering close-by view poses together to reduce the quadratic number of path planning requests necessary. Transition times comprise the driving time (predicted using a linear regression over path costs returned by the planner), the time needed to lift the robot's torso to achieve different sensor heights, the time needed to move the pan/tilt head, and a fixed amount of sensing time at the target pose.

Let a sequence of views be denoted as $L = \langle L_0, \ldots, L_n \rangle$, where L_0 is the current state and each L_k has a corresponding view $s_{L,k} \in \mathcal{S}$. The goal is to find a sequence $\hat{L}$ that minimizes the expected time until the target is found which is given by

$$E[T|L] = \sum_{i=0}^{n-1} \prod_{k=0}^{i} \big(1 - p(s_{L,k} | \langle L_0, \ldots, L_{k-1} \rangle)\big) t(L_i, L_{i+1}) \,, \qquad (4)$$

where $p(s_i|L)$ is the probability to find a target with observation s_i after having already seen the sequence L. Accounting for all cells already seen by L, this is calculated as

$$p(s_i|L) = p\left(\mathit{cells}(s_i) \setminus \bigcup_{k=0}^{n} \mathit{cells}(L_k) \right) . \qquad (5)$$

Sarmiento et al. (2003) examine this optimization problem in a similar context and employ a heuristic algorithm called *utility greedy* to quickly compute approximate solutions. Our planner uses the key heuristics of utility greedy in a depth-first branch and bound approach to construct plans without consuming too much memory. Since the objective function is the plan's expected runtime, the obvious greedy heuristic to determine the order in which new views $s_i \in \mathcal{S}$ are appended to partial plans is to minimize the local increase in expected runtime. However, this performs poorly, confirming the observation made by Sarmiento et al. who propose to improve efficiency using a different utility function given by

$$\mathit{util}(s_i|L) = \frac{p(s_i|L)}{t(L_n, s_i)} \,. \qquad (6)$$

Since this function proves to be effective, we adopt it for our planner[2]. To further accelerate planning, Sarmiento et al. sacrifice optimality and introduce a pruning heuristic that discards *strictly dominated* views s_k given a partial plan L according to

$$\begin{aligned} &\mathit{dominated}(s_k|L) :\Leftrightarrow \\ &\exists s_i \in \mathcal{S} : p(s_i|L) > p(s_k|L) \wedge t(L_n, s_i) < t(L_n, s_k) \,. \end{aligned} \qquad (7)$$

In what follows, use of this pruning heuristic by the planner will be signaled with the flag $\xi \in \{0, 1\}$.

We introduce two additional mechanisms to reduce the amount of branching in the planner. First, branching is only allowed up to a *depth limit* ψ. Second, only expansions are allowed for which the utility value is not smaller than the best utility value divided by a *branch limit* factor ϕ. As the value of ϕ increases, more branching takes place, since less views are discarded. For $\psi = 0$ or $\phi = 1$, the planner constructs a single greedy sequence without branching.

As evidenced by (4) to (7), the planner needs to perform set operations to track a plan's quality and to evaluate heuristic functions. It should be noted that, although (5) is formulated in terms of set union, the planner is implemented using only set difference, resulting in faster set operations with increasing planning depth.

The total probability of success for a plan can be calculated analogously to (3) as

$$p(L) = 1 - \prod_{L_i \in L} \big(1 - p(L_i | \langle L_0, \ldots, L_{i-1} \rangle)\big) \,. \qquad (8)$$

A plan is complete when the remaining probability to encounter a target by adding more view poses falls below the termination threshold ω, as expressed by

$$\mathit{complete}(L|\mathcal{S}) :\Leftrightarrow p(L) > 1 - \frac{1 - p(\hat{V}(\mathcal{S}))}{1 - \omega} \,. \qquad (9)$$

After planning finishes, the robot navigates to move the sensor into the first planned position, senses, and waits

[2] See Sarmiento et al. (2003) for an analysis of the relation between the utility function and the objective function and why the naive utility function performs poorly.

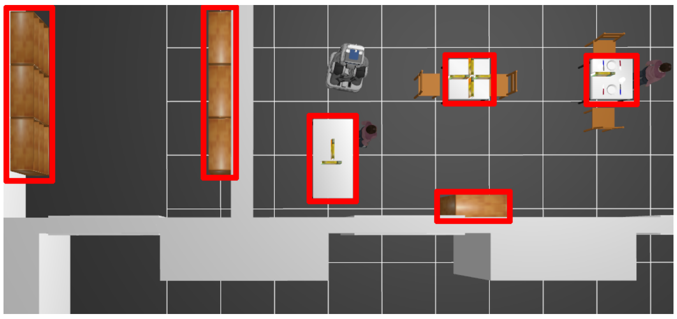
Fig. 4. Top-down view of the simulated environment used in the experiments, consisting of two connected rooms with various target volumes (red boxes: two shelf arrangements, one single shelf, three cluttered tables).

for sensor data integration. If the search process is not externally interrupted, that means that no target object has yet been found, and the procedure starts another cycle. Each cycle retains the previously planned views, which are reevaluated and augmented with additional views using the view sampling system. A new plan is constructed continually in each cycle until $p(\hat{V}(\mathcal{S})) < \omega$.

4. EVALUATION

The robot used in the experiments is a Willow Garage PR2 with a head-mounted RGB-D camera which can be panned 350° and tilted 115°. The head is mounted on a telescoping torso with a lift range of 31 cm. The real-world functionality of the active perception system has been shown as part of the final demonstration of the RACE project[3]. We evaluated the system in detail using the Gazebo simulator with a simplified robot model that retains the physical limitations of the real version. Robot motion is replaced by updating poses directly in the simulation with simulated transition times that are obtained from the same model that is used by the planner to predict execution times. Robot localization is obtained directly from simulated ground truth. This intentionally eliminates any influence that imperfect robot localization and navigation as well as an inaccurate execution time model may have on object search performance. Fig. 4 shows an overview of the simulated environment used in all experiments. Run times of all algorithms were measured on an Intel Core i7 870 CPU.

We ran three series of experiments evaluating several planner configurations, and, while computing the optimal solution was not feasible, compared the performance of the planning algorithm with the greedy approach. The task in all experiments is to locate a target within the volumes shown in Fig. 4 with a probability of $P_i = 0.2$ for each volume to contain at least one instance of the target. Each experiment comprises 20 trials starting with a random initial robot pose and ending when the remaining probability of detecting the target falls below $\omega = 0.05$. To measure object search efficiency, we determine the mean expected value of the time needed to complete search tasks as the sum of an execution time component (motion and sensing), and a planning time component[4].

[3] A video of the system running on the PR2 can be found at `http://kos.informatik.uos.de/flap4caos/`

[4] See (4) for the calculation of expected values

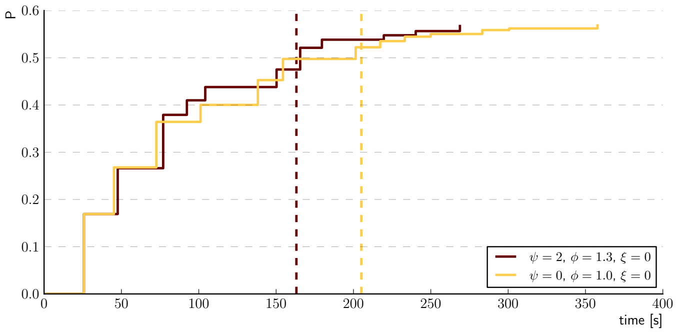

Fig. 5. Cum. prob. of success over time for one run of greedy search vs. continual planning with initial map. Vertical steps: sensing actions; horizontal steps: pose transitions; dashed lines: expected execution times.

We generate 200 view samples for each trial, sufficient for almost complete coverage of our test environment, and share them between all planner configurations to enable a direct comparison. Accordingly, a lookup table for predicted transition times between views is generated once per trial and shared between all planner configurations. Table 1 shows the average computation times for these operations over all experiments. The total probability to find a target is determined by the structure of the environment, the set of available view poses, the choice of target volumes, and their target encounter probabilities, and is hence equal for all planner configurations within each trial. In the first two experiments, we test offline and continual planning given a full 3D map of the environment, thereby enabling our planner to fully optimize the search process from the start. In the third experiment, we test continual planning without an initial map (only walls are known in advance), so that the system is forced to adapt the search strategy on the go.

5. DISCUSSION AND FUTURE WORK

As shown in Fig. 6, the object search planner outperforms the greedy strategy in all experiments with regard to the expected execution time of the resulting plan as well as the combined expected run time of the search process. In case a map is available beforehand, continual planning achieves run times that are on average 10 % shorter compared to greedy, and it compares preferably to offline planning, which is remarkable considering the computational cost of repeated planning. While the advantage of planning decreases when no initial map is available, we still achieve a 7 % reduction of average search time using knowledge acquired during the ongoing search process itself. A visualization of search progress during a single task is shown in Fig. 5. The planning algorithm achieves coverage of the search region using fewer sensing steps (11 vs. 13) and also tends to reduce average transition times between sensing poses. Although the planning algorithm yields a shorter total search time, the simple greedy strategy is still a viable alternative with the advantages of being parameter free and computationally cheap. The relative competitiveness

Table 1. Average algorithm run times

View sampling (200 views)	2.4 ± 0.1 s
Transition time prediction (50 clusters)	6.4 ± 0.6 s

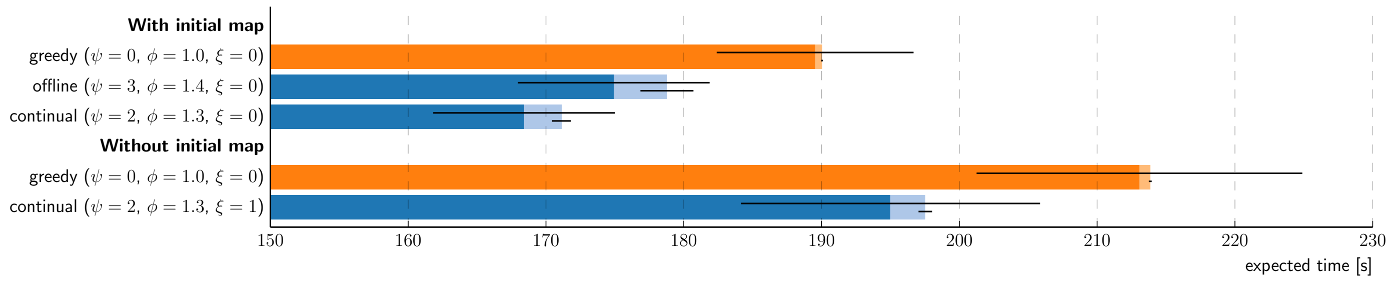

Fig. 6. Mean expected execution times (dark) and mean expected total planning times (bright) for the greedy strategy vs. the best performing planner configuration in each experiment. Note: The x-axis is truncated and starts at 150 s.

of greedy strategies is not uncommon in related work as reviewed in Sec. 2.

It is part of future work to examine how appropriate planner parameterizations change with different search tasks and working environments. The speed of plan generation may be less important in applications with different cost functions, e.g., energy consumption or risk of failure, which our planner can be adapted for by choosing appropriate parameters with enlarged planning scope. As stated in Sec. 3.2, the current approach disregards issues of target recognition. Future developments will have to address the fact that targets in real applications cannot be recognized with full reliability and may be only partially visible due to occlusion or field of view limits. Additionally, some unexplored volumes could be excluded from search simply because they are too small to contain the target object. Our experiments used a simple model of target location probabilities and manually selected target volumes. In principle, both can come from arbitrary sources (see Sec. 2) and can be updated during the search process. A source for target object location probabilities is available with the probabilistic anchoring module of the RACE system, based on a probabilistic model of spatial relations, and is currently being evaluated separately. Future work will investigate the integration of both systems.

REFERENCES

Anand, A., Koppula, H.S., Joachims, T., and Saxena, A. (2011). Contextually Guided Semantic Labeling and Search for 3D Point Clouds. *CoRR*, abs/1111.5358.

Blodow, N., Goron, L.C., Marton, Z., Pangercic, D., Rühr, T., Tenorth, M., and Beetz, M. (2011). Autonomous Semantic Mapping for Robots Performing Everyday Manipulation Tasks in Kitchen Environments. In *Proc. IROS 2011*. San Francisco, CA, USA.

Connolly, C.I. (1985). The Determination of Next Best Views. In *Proc. ICRA 1985*. St. Louis, Missouri, USA.

Dogar, M.R., Koval, M.C., Tallavajhula, A., and Srinivasa, S.S. (2014). Object Search by Manipulation. *Auton. Robot.*, 36(1-2), 153–167.

Dornhege, C. and Kleiner, A. (2013). A Frontier-Void-Based Approach for Autonomous Exploration in 3D. *Adv. Robotics*, 27(6), 459–468.

Dornhege, C., Kleiner, A., Hertle, A., and Kolling, A. (2015). Multirobot Coverage Search in Three Dimensions. *J. Field Robot.* (online, print version in press).

Gupta, M., Rühr, T., Beetz, M., and Sukhatme, G.S. (2013). Interactive Environment Exploration in Clutter. In *Proc. IROS 2013*. Tokyo, Japan.

Hertzberg, J., Zhang, J., Zhang, L., Rockel, S., Neumann, B., Lehmann, J., Dubba, K.S.R., Cohn, A.G., Saffiotti, A., Pecora, F., Mansouri, M., Konecny, S., Günther, M., Stock, S., Lopes, L.S., Oliveira, M., Lim, G.H., Kasaei, S.H., Mokhtari, V., Hotz, L., and Bohlken, W. (2014). The RACE Project – Robustness by Autonomous Competence Enhancement. *KI*, 28(4), 297–304.

Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Auton. Robot.*, 34(3), 189–206.

Kunze, L., Beetz, M., Saito, M., Azuma, H., Okada, K., and Inaba, M. (2012). Searching Objects in Large-scale Indoor Environments: A Decision-theoretic Approach. In *Proc. ICRA 2012*. St. Paul, Minnesota, USA.

Kunze, L., Doreswamy, K.K., and Hawes, N. (2014). Using Qualitative Spatial Relations for Indirect Object Search. In *Proc. ICRA 2014*. Hong Kong, China.

Moldovan, B. and Raedt, L.D. (2014). Occluded Object Search by Relational Affordances. In *Proc. ICRA 2014*. Hong Kong, China.

Rasouli, A. and Tsotsos, J.K. (2014). Visual Saliency Improves Autonomous Visual Search. In *Proc. CRV 2014*. Montreal, QC, Canada.

Renton, P., Greenspan, M.A., ElMaraghy, H.A., and Zghal, H. (1999). Plan-N-Scan: A Robotic System for Collision-Free Autonomous Exploration and Workspace Mapping. *J. Intell. Robot. Syst.*, 24(3), 207–234.

Sarmiento, A., Murrieta-Cid, R., and Hutchinson, S. (2003). An Efficient Strategy for Rapidly Finding an Object in a Polygonal World. In *Proc. IROS 2003*. Las Vegas.

Shade, R. and Newman, P. (2011). Choosing Where To Go: Complete 3D Exploration With Stereo. In *Proc. ICRA 2011*. Shanghai, China.

Shubina, K. and Tsotsos, J.K. (2010). Visual Search for an Object in a 3D Environment Using a Mobile Robot. *Comput. Vis. Image Und.*, 114(5), 535–547.

Surmann, H., Nüchter, A., and Hertzberg, J. (2003). An Autonomous Mobile Robot with a 3D Laser Range Finder for 3D Exploration and Digitalization of Indoor Environments. *Robot. Auton. Syst.*, 45(3-4), 181–198.

Wong, L.L.S., Kaelbling, L.P., and Lozano-Pérez, T. (2013). Manipulation-Based Active Search for Occluded Objects. In *Proc. ICRA 2013*. Karlsruhe, Germany.

Yamauchi, B. (1997). A Frontier-Based Approach for Autonomous Exploration. In *Proc. CIRA 1997*. Monterey, California, USA.

Ye, Y. and Tsotsos, J.K. (1999). Sensor Planning for 3D Object Search. *Comput. Vis. Image Und.*, 73(2), 145–168.