2012 IEEE International Conference on Robotics and Automation
RiverCentre, Saint Paul, Minnesota, USA
May 14-18, 2012

Searching Objects in Large-scale Indoor Environments: A Decision-theoretic Approach

Lars Kunze, Michael Beetz
Intelligent Autonomous Systems Group
Technische Universität München
{kunzel, beetz}@cs.tum.edu

Manabu Saito, Haseru Azuma,
Kei Okada, Masayuki Inaba
Johou Systems Kougaku Laboratory
The University of Tokyo
{saito, chen, k-okada, inaba}@jsk.t.u-tokyo.ac.jp

***Abstract*— Many of today's mobile robots are supposed to perform everyday manipulation tasks autonomously. However, in large-scale environments, a task-related object might be out of the robot's reach. Hence, the robot first has to search for the object in its environment before it can perform the task.**

In this paper, we present a decision-theoretic approach for searching objects in large-scale environments using probabilistic environment models and utilities associated with object locations. We demonstrate the feasibility of our approach by integrating it into a robot system and by conducting experiments where the robot is supposed to search different objects with various strategies in the context of fetch-and-delivery tasks within a multi-level building.

I. Introduction

Autonomous mobile robots that are to perform everyday tasks like, e.g., fetching and delivering objects need the appropriate capabilities to obtain task-related objects from the environment. That is, they first have to search for the required objects and retrieve them to accomplish the task.

Let us consider, e.g., a scenario where a robot is supposed to deliver a sandwich to a person (Figure1). In order to do so, the robot has to reason at which places in its environment it would most likely find a sandwich while also considering utilities associated with these places, e.g., travel costs. That is, the robot should go to the place which maximizes the expected utility to find a sandwich. For instance, the robot might infer that the most likely place to get a sandwich is a fast-food restaurant. However, since it is far away the robot decides first to search in a kitchen, which has a lower probability of success but it is closer to its current position.

When we look at the performance of humans in object search tasks, it seems, that in most cases humans can find objects in their environment relatively effortless and at very high success rates. This is not only because humans have excellent perception capabilities, but also because humans can rely on a large body of commonsense knowledge and experience to conduct the search for objects more effectively.

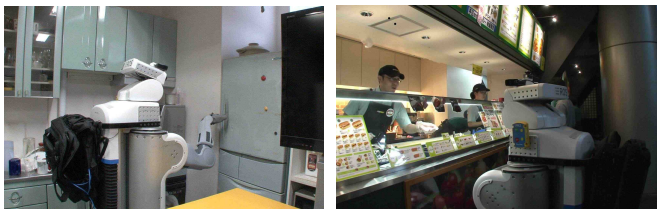

Fig. 1. Example scenario: searching a sandwich at different places, e.g. kitchen (left) and restaurant (right).

Although perception plays a key role in object search within robotics, it is also very important to employ semantic knowledge to narrow down the search space, especially in large-scale environments. Pruning the search space is mainly important because of two reasons, first, robot perception is computational expensive, and second, if robots have no clue about potential object locations, objects will only be found by employing exhaustive search methods or by chance.

In the present work, we investigate the problem of how robots can use semantic environment models to perform object search tasks in large-scale environments more effective and efficiently. The contributions of this work are as follows. First, we extend the representations and reasoning methods for semantic maps that have been introduced in [1] to account for large-scale indoor environments, i.e. multi-level buildings. Second, we utilize commonsense knowledge acquired from Internet users to bootstrap probabilistic models about typical locations of objects which can be updated during the robot's lifetime according to its observations. Third, we present a decision-theoretic framework and several object search strategies using the above models while also considering utilities associated with the potential object locations. Finally, we integrate the developed methods within a robot system and provide experimental results for object search in fetch-and-delivery tasks within a multi-level building.

II. The Fetch-and-Delivery Scenario

In fetch-and-delivery tasks robots are, e.g., supposed to serve food and drinks, deliver letters, or collect used cups from workplaces and take them to the kitchen. An essential sub-task of such tasks is to search for task-related objects in the environment. In this paper, we investigate two scenarios: In the first scenario the robot is supposed to find cups and bring them back to its starting position and in the second scenario we ask the robot to deliver a sandwich.

Let us look at the cup scenario in more detail. For example, consider a situation where a robot is supposed to fetch a particular cup, let us say Michael's cup. In order to find the cup in its environment the robot first replaces the possessive attribute with one or more perceptual attributes, e.g., *Owns(Michael, Cup)* is replaced by *HasLogo(Cup, PR2)*. Either the robot knows about the perceptual attributes of

978-1-4673-1404-6/12/$31.00 ©2012 IEEE

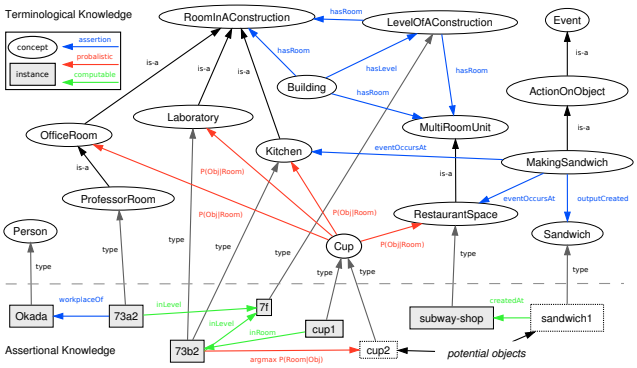

Fig. 2. Simplified overview of concepts and relations of the semantic model for large-scale environments.

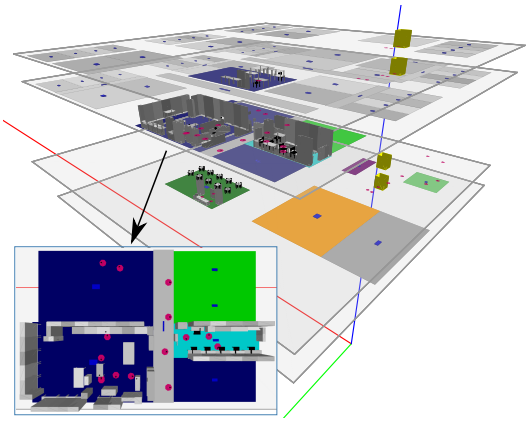

Fig. 3. Engineering Building No. 2, Hongo Campus, University of Tokyo. The map comprises several floors, elevators (yellow boxes), rooms with different types, furniture, and a subway restaurant.

the cup because of its own perceptual experience or this information has to be provided to the robot. Then the robot retrieves the set of known cup instances from its belief state that fulfill the partial description, e.g. *HasLogo(Cup, PR2)*, or at least, do not contradict it. However, if the robot does not know about any cup instance in the environment, it has to rely on more general knowledge about cups. For example, the robot could infer that cups are typically stored in the cupboards of a kitchen and that they are occasionally located in a dishwasher. In the next step, the robot retrieves the poses of the potential cup locations as well as the poses from which the cups might be perceivable from the semantic map. The latter poses are used as goal locations for the autonomous navigation of the robot. In order to find the cup, the robot moves to the respective goal locations while taking path costs (utilities) into account. Having reached a potential cup position the robot uses perception routines for detecting and localizing the cup in the environment. If a cup is detected and fulfills the partial object descriptions the robot applies more specialized perception routines in order to gather all information needed for effectively grasping the cup. However, if the cup was not found, the robot will continue until it has searched the remaining locations.

III. Semantic Environments Models

This sections describes the underlying representations of the semantic environment models. Figure 2 gives an overview of the different ontological concepts and relations, whereby we distinguish mainly between three types of relations: assertional, computable, and probabilistic.

A. Semantic Maps of Large-scale Environments

In this work, we basically build on the representation formalisms as described in [1]. The underlying representation is based the Web Ontology Language (OWL). In OWL, classes are defined within a taxonomy and derive all the properties of their super-classes. Relations between classes are described by properties. Objects are represented as instances of classes and can also have relations with other instances. The upper ontology is derived from OpenCyc[1].

With respect to semantic environment maps, this formalism allow us to structure the knowledge about classes hierarchically, e.g. a *ProfessorOffice* is an *Office* which is a *RoomInAConstruction*. Given the mechanism of (multiple-) inheritance a class derives all the properties of its super-class(es), e.g., *ProfessorOffice* have also the property *roomNumber* since it has been defined for *RoomInAConstruction*. Objects in the map are described by instances and have a certain *type*, e.g. *Office*, properties to other instances, e.g. *workplaceOf*, and simple data properties like *widthOfObject*. The spatial extensions of an object are described by their bounding box, i.e. depth, width, and height. Although this representation is very simple, it allows us to reason about several spatial relations between objects like *in-Container* or *on-Physical* more efficiently. Furthermore, objects are related to instances of perception events. These events contain information about the object's pose at a point in time. Thereby we can track the pose of an object over time and answer questions like where was an object five minutes ago.

In [1], the map representation is mainly focused on small environments like rooms, especially kitchens. However, in the context of fetch-and-delivery tasks it is important to represent also the knowledge about environments at a larger scale. Therefore we introduced concepts like *Building*, *LevelOfAConstruction*, *RoomInAConstruction*, and *Elevator*. Figure 2 show a small excerpt of the extended ontology. Additionally, we introduced properties like *floorNumber*, *roomNumber*, *inLevel*, *inRoom*, and *workplaceOf*. Whereas properties like *roomNumber* are directly asserted to particular instances (assertional property), other properties like *inRoom* are computed based on the actual physical configuration of the environment (computable property), i.e. the current pose and dimensions of an object are taken into account. Thereby we can infer whether spatial relations between objects like *in* and *on* hold at some point in time.

In this work, we created a semantic environment map of the Engineering Building No. 2 at the Hongo Campus of The University of Tokyo. Figure 3 visualize some aspects of the semantic map including levels, rooms, furniture and places.

[1]http://www.opencyc.org/

B. Knowledge about Object Locations

Knowledge about object locations can be both concrete knowledge about individuals, e.g., *cup4* is in *kitchen1*, and abstract knowledge about object classes, e.g., given we have seen a cup the probability that we are in a kitchen is 0.25. Knowledge of the latter kind becomes very important when the robot has no information about individual objects. In this section, we describe how we bootstrap probabilistic models for reasoning about the locations of object classes.

In [2], we investigated how commonsense knowledge that was acquired from Internet users within the Open Mind Indoor Common Sense (OMICS) project [3] can be transformed from natural language to formal representations and integrated into a robot's knowledge base.

The *locations* relation in the OMICS database describes objects and their typical locations, i.e. rooms. Following our previous approach, we first transform the natural language database entries to ontological concepts. For example, the tuple *(mug,kitchen)* is mapped to well-defined ontological concepts *(Cup,Kitchen)*. Then, we calculate the conditional probability of an object given the room type by counting the database entries as suggested by [3], i.e.:

$$P_{cs}(x_i|\rho) = (C(x_i,\rho) + \lambda)/(C(\rho) + \lambda n) \qquad (1)$$

where x_i denotes an object, ρ a room type, n the number of objects, and λ the parameter according to Lidstone's law. The λ parameter basically influences how the probability distribution account for unseen tuples. In our experiments, we set $\lambda = 0.5$ (Jeffrey-Perk's law).

In total *locations* database table has more than 3500 entries. However, since we could not map all entries to ontological concepts and we restricted the set of rooms concepts to nine different types that fit our map, we used only 448 entries for calculating the probabilistic properties.

We just explained how we can bootstrap a probabilistic model $P_{cs}(x_i|\rho)$ for commonsense knowledge about object locations. However, the likelihood of object locations in the robot's environment might deviate from this model. That is, in addition to the commonsense model $P_{cs}(x_i|\rho)$ we capture the robot's knowledge derived from its individual experiences in a second model $P_{exp}(x_i|\rho)$.

We believe that the main difference between these models is in the way they are updated. Whereas the commonsense model P_{cs} is updated by the experiences of many robots, the P_{exp} model is only updated by the robot's own experience and thereby it captures the idiosyncrasies of a particular environment. To update a model we simply add an object/location tuple to the database. For initializing P_{exp} we also used the commonsense knowledge from the OMICS database.

Obviously, when reasoning about object locations robots should consider both models, i.e., P_{cs} and P_{exp}. How these models are combined is explained in the following section.

IV. Semantic Object Search

We first present the decision-theoretic framework, second, we explain various search strategies, and finally, we describe how utilities influence the robot's behavior.

A. Decision-theoretic Object Search

The basic idea of the decision-theoretic framework is that the search location that maximizes the expected utility is determined by both the probability of success to find an object at a location of a certain type and the utility associated with a particular location which is influenced by, e.g., travel costs and task context. That is, the location (or room) r^* that maximizes the expected utility is computed by

$$r^* = \underset{r}{\operatorname{argmax}} \sum_{\rho} P(\rho|x)U(r) \qquad (2)$$

where room r is of type ρ. How the probability distribution $P(\rho|x)$ and the utility function $U(r)$ are calculated is explained the following two sections.

B. Search Strategies

In the introduction of this paper we already mentioned the excellent ability of humans to find objects, sometimes even in previously unknown environments. Among other reasons, this is because humans make use of different strategies when looking for an object. Therefore, robots should also be equipped with a repertoire of strategies they can employ, depending on what knowledge they have about the environment. For example, a robot that has knowledge about some instances of a sought object class should exploit this information before considering more general knowledge. However, if a robot does not have any information about object instances, it has to rely on common sense. How these different kinds of search strategies should be orchestrated is another interesting problem which is beyond the scope of this paper. Hence, in this work we assume that different searches are conducted in a kind of *try-in-order* procedure.

In the following we explain some basic predicates that have been implemented in PROLOG in order to search for objects with different strategies using the semantic environment models described in the previous section.

locatedAt(Obj, Pose) denotes the pose of an object. Poses are described by the translation and rotation of an object decoded in a 4×4 transformation matrix.

lookForAt(Obj, Pose) denotes the pose where an object instance might be perceivable.

hasType(Obj, Type) denotes the type of an object. When only providing a type all object instances with the specified type are mentally retrieved from the map.

locationOf(LocType, ObjType, P) denotes the probability P to encounter a given object type at a location type. The probability is calculated from the probabilistic models explained in Section III-B. The distributions are combined linearly

$$P(\rho|x) = \sum_i w_i P_i(\rho|x) \qquad (3)$$

where P_i denotes either P_{cs} or P_{exp} and the weights w_i scale the influence of a knowledge source whereby $\sum_i w_i = 1$. Later, we plan to integrate also distributions about individuals of objects and/or rooms.

The value of P_i is calculated from the respective models using Bayes' rule:

$$P_i(\rho|x) = \frac{P_i(x|\rho)P_i(\rho)}{\sum_j P_i(x|\rho_j)P_i(\rho_j)} \quad (4)$$

To retrieve the location with the highest probability we simply apply the *argmax* operator $\underset{\rho \in \rho}{\operatorname{argmax}}\ P(\rho|x)$.

Given these basic predicate definitions it is already possible to implement different kinds of search strategies when looking for an object. The most noticeable difference is that some predicates use knowledge about known instances in the robots environment whereas others only consider general knowledge about classes.

C. Search Utilities

In the previous section, we explained some basic predicates a robot can use to retrieve objects and locations from the semantic environment map. However, the decision to which location the robot will move to and look for an object, as explained in Section IV-A, depends also on the location's utility. This section describes how utilities for locations, which depend on the situational context of the robot, are calculated. In this paper, the context is solely determined by the robot's current pose.

In this work we consider a rough heuristic to calculate the path cost from the current position of the robot to a potential goal location. For the heuristic we distinguish two situations:

1) robot pose and goal pose are in the same level
2) robot pose and goal pose are in different levels

If the robot pose and the goal pose are in the same level, the path cost is determined simply by the Euclidean distance between the two poses. Obviously, the path cost calculation can also be approximated by using path planner on 2D occupancy grid maps.

If the robot pose and the goal pose are not in the same level, the overall path cost is determined by the sum of three cost calculations: (1) the path cost from the current position to the elevator in the same level, (2) the cost using the elevator, and (3) the path cost from the elevator to the goal pose. The costs (1) and (3) are calculated as explained above. The elevator cost is determined by the distance of levels. However, the cost model for using an elevator could be replaced by a more complex model considering for example the waiting time, intermediate stops, and the hour of the day. Figure 4 visualizes the path costs from room *73a4* in floor 7 to other rooms in the building.

The utility is calculated from the path costs as follows:

$$U(r) = 1/(\operatorname{atan}(path\_cost(current\_pose, r)) + \epsilon) \quad (5)$$

where ϵ (0.01) accounts for the cases when the distance is zero. The underlying idea of choosing this utility function is to give more weight on the probability of success than on the utility function (path costs) of a room. However, whether this choice is reasonable has to be evaluated through experiments.

The utility function is realized by the following PROLOG predicate: *utitilyOf(Location, CurrentPose, Utility)*. Internally, it uses a path cost function as described above.

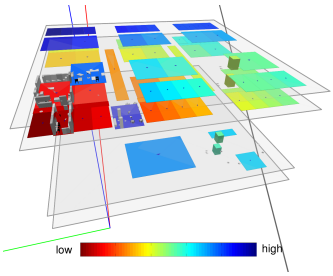

Fig. 4. Path costs visualized as heatmap calculated from room *73a4*, 7th floor (dark red).

V. Experimental Results

First, we introduce the robot system, and second, we present the results of two example scenarios. Videos of both scenarios are available here[2].

A. The Robot System

Within the experiments, we use the PR2[3] robot platform and a ROS[4]-based software infrastructure. An overview of the different software components is shown in Figure 5.

Basically, the robot's main control program is responsible for performing the fetch-and-delivery tasks. It receives a task goal from an iPad interface and performs the task autonomously. Alternatively, the task can be carried out in interaction with a user. After having received the task goal, i.e. an object to search for, one of the search strategies is used to query the semantic environment models for potential object locations. When the task-level control program recognizes that the robot has to change the floor, the relevant information is retrieved from the semantic environment model, e.g. the number of the current and the target floor, and the position of elevator control panels. These information are then send to the inter-floor navigation module to navigate the robot to the respective locations. At the goal location the robot tries to detect the object under request by using a sift-based template matching approach. Finally, the robot uses motion planning to figure out how to grasp the object.

B. Cups

In the first scenario, we look at situations in which a robot is supposed to find cups in the environment.

In the first situation the robot is asked to look for a particular cup, a cup that has a certain logo on it. Let us assume the robot knows some cup instances in the environment, and it also has a model of the logo. However, this logo is not associated to any of the cups, i.e., the following query fails:

```
?- hasType(C,'Cup'), hasLogo(C, 'CMU').
```

Hence, the robot has to move to all possible cup locations and try to match the logo with one of the cups perceptually. The following PROLOG query shows how the robot can retrieve the positions of potential cup locations from the semantic map and calculate their respective utilities. Since

[2]Videos: http://www.jsk.t.u-tokyo.ac.jp/~kunzel/objsearch.html
[3]http://www.willowgarage.com/pages/pr2/overview
[4]http://www.ros.org/

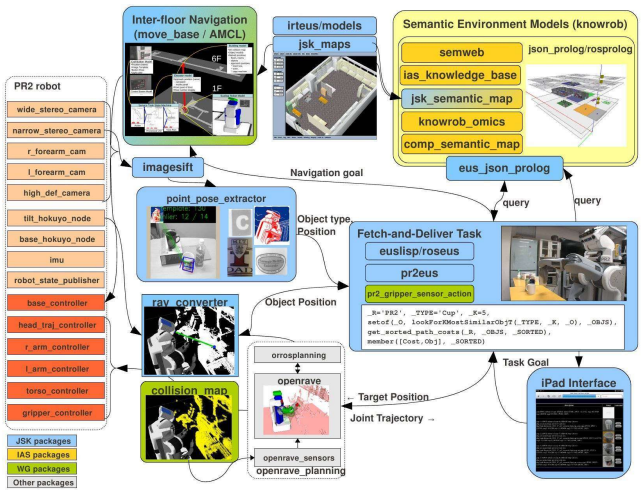

Fig. 5. Overview of the main software components realizing the execution of fetch-and-delivery tasks.

we assume that the known cup instances are found at their respective locations with the same probability the expected utilities are solely determined by the utilities itself. The list of poses ordered by their utilities is than passed to the navigation framework which uses the *lookForAt(Obj,Pose)* predicate to determine the navigation goals.

```
?- findall(Pose, (hasType(Obj, 'Cup'),
                  not(hasLogo(Obj, AnyLogo)),
                  locatedAt(Obj, Pose)), Poses),
   utilitiesOf(Poses, 'current-pose', PUtils).
```

Figure 6 visualizes the query result in the semantic map and show some images of the carried out robot experiment. After detecting a cup with a different logo in the first room, the robot navigates to another room where it eventually find the sought cup. We varied the experiment by placing the individual cups at the different locations in various permutations or removing individual cups completely.

In the previous situation we assumed that the robot had some knowledge about cup instances. However, if the robot has no such information it has to rely on more general knowledge. The following query illustrates how the robot retrieves potential locations for an object of type *Cup*.

```
?- findall([P, RoomType], locationOf(RoomType ,'Cup' , P),
           RTProbabilities),
   findall(R, hasType(R, 'RoomInAConstruction'), Rs),
   findall(Pose,(member(Room,Rs),
                 inRoom(Spot,Room),
                 hasType(Spot,'Place'),
                 locatedAt(Spot,Pose)),
           Poses),
   utilitiesOf(Poses, 'current-pose', PUtils),
   expectedUtilities(RTProbabilities, PUtils, ExpUtils).
```

First, the robot finds all tuples of probability *P* and room type *RoomType*, given that it is looking for a *Cup*. Figure 7 shows some probabilities that an object type can be found in a certain room type. Second, it retrieves all room instances from the semantic map and considers all known places in these rooms. Third, for all these places a utility is calculated, and finally, the expected utilities are computed based on the probabilities of room types given a *Cup* and the utilities of places given the current robot pose. Eventually, the robot

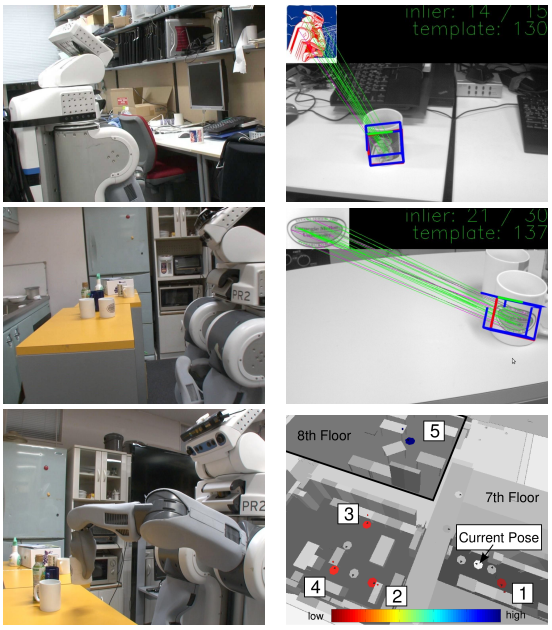

Fig. 6. Row 1: Robot at place (1). Found cup, but has PR2 logo. Row 2: Robot at place (2). Found cup with desired CMU logo. Row 3: Robot picks up cup. Visualized query results in semantic map.

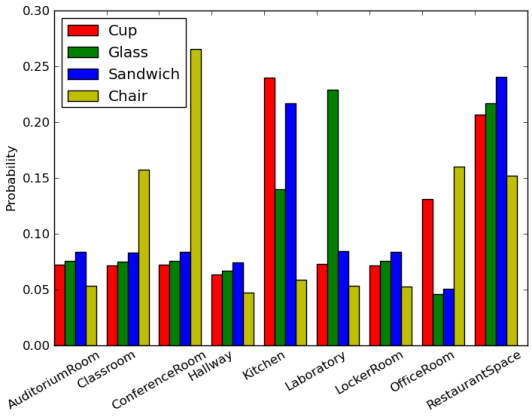

Fig. 7. Examples of probabilities to find a given object in certain room, i.e. $P(\rho|x)$. The initial configuration is bootstrapped from the OMICS database.

moves to the place for which the expected utility is maximal where it tries to perceive a cup at different angles.

Using the query above the robot gets, for example, the information that the expected utility to find a *Cup* instance is maximal for some place in a *Kitchen*. However, not all places in the kitchen make sense to look for a cup, e.g., one place is in front of a TV. In future work, we plan to incorporate additional knowledge within the utility function, e.g., that places should be in front of a table, shelf, or cupboard.

In addition to the autonomous object search, we also developed an interactive mode where users can search for objects using an iPad interface. Basically, users can ask for an object and receive a list of possible object locations. After taking a user's request, the system searches the semantic environment models for possible locations using a selected search strategy, generates a list of potential locations including information

Fig. 8. PR2 using an elevator, approaching subway, and ordering cd a sandwich.

about floor, room, costs, and if available an image of the object's visual appearance. The user can then decide from which location the robot should fetch the object.

C. Sandwiches

Figure 8 shows some aspects of the scenario where the robot is asked to get a *Sandwich*[5].

The robot uses the decision-theoretic framework to determine the expected utilities for each room and then simply goes to the location where the expected utility is maximal. If we assume that the robot is at some place in room *73b2* (7th floor) and searches for a sandwich ($x = Sandwich$) the related expected utilities, probabilities of success, utilities, path costs, floor numbers, room numbers, and room types are shown in Table I. Since the robot tries to maximize the expected utility it first decides to look for a sandwich in room *73b2*, then at the *subway* restaurant and so on. It is notable that the order of rooms with respect to the expected utility is neither equivalent to the order with respect to the probability of success nor to the utility function. Alternatively, the robot could also use a greedy strategy by re-computing the expected utilities after unsuccessful trials.

TABLE I
EXPECTED UTILITIES

$EU(r)$	$P(\rho\|x)$	$U(r)$	path cost	f	r	ρ
0.1543	0.2165	0.7127	5.5639	7f	73b2	Kitchen
0.1528	0.2400	0.6368	95.0167	2f	subway	Restaurant
0.1426	0.2165	0.6587	15.8837	7f	7a	Kitchen
0.0532	0.0833	0.6383	69.7814	2f	21a	LockerRoom
0.0528	0.0827	0.6378	77.1014	2f	22a	Classroom
0.0343	0.0506	0.6783	9.3487	7f	73a3	OfficeRoom
0.0338	0.0506	0.6691	11.5633	7f	73a2	OfficeRoom
...	...	...	...	...	...	...
0.0323	0.0506	0.6387	65.9891	7f	71d1	OfficeRoom

VI. RELATED WORK

In recent years, the usage of semantic information has become more and more prominent in the context of robotics.

In [4], object-room relations are exploited for mapping and navigation tasks. Such kind of knowledge is often represented by the means of Description Logics [5], [6], or probabilistic models [7], [8].

How to use semantic knowledge within object search tasks is explored by [9], [10], [11]. Our approach can be considered in a similar line of research. Probabilistic models are used in [9] to guide a simulated robot during object search tasks in structured indoor environments, namely supermarkets. Work by [10] utilizes also probabilistic methods on a robot system to make inferences about the spatial relations between object instances in the environment. Similar to our approach, [11] bootstraps commonsense knowledge from the OMICS database to initialize the probabilistic representations. However, other forms of reasoning are not investigated in their robot experiments.

VII. CONCLUSION AND FUTURE WORK

In this paper, we investigated how semantic information about the robot's environment can be used in object search tasks. To this extent, we extended the representation formalisms introduced in previous work ([1]) and implement several PROLOG programs to infer the potential locations of an object. As proof-of-concept, we integrated the developed methods into a robot system that searches objects within a multi-level building in the context of fetch-and-delivery tasks. The realized system is able to navigate to inferred objects location using different kinds of semantic information.

In future work, we will include further probabilistic models about spatial relations like *in*, *on*, and *next-to* in order to restrict the search space of objects and thereby make the search tasks even more efficiently. Additionally, we will integrate probability distributions that represent the locations of individual objects directly into the decision-theoretic framework to unify the inference process.

[5]http://www.youtube.com/watch?feature=player_embedded&v=RIYRQC2iBp0

REFERENCES

[1] M. Tenorth, L. Kunze, D. Jain, and M. Beetz, "KNOWROB-MAP – Knowledge-Linked Semantic Object Maps," in *Proceedings of 2010 IEEE-RAS International Conference on Humanoid Robots*, Nashville, TN, USA, December 6-8 2010.

[2] L. Kunze, M. Tenorth, and M. Beetz, "Putting People's Common Sense into Knowledge Bases of Household Robots," in *33rd Annual German Conference on Artificial Intelligence (KI)*. Karlsruhe, Germany: Springer, September 21-24 2010, pp. 151–159.

[3] R. Gupta and M. J. Kochenderfer, "Common sense data acquisition for indoor mobile robots," in *Nineteenth National Conference on Artificial Intelligence (AAAI)*, 2004, pp. 605–610.

[4] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. Fernández-Madrigal, and J. González, "Multi-hierarchical semantic maps for mobile robotics," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Edmonton, CA, 2005, pp. 3492–3497.

[5] H. Zender, O. Martínez Mozos, P. Jensfelt, G. Kruijff, and W. Burgard, "Conceptual spatial representations for indoor mobile robots," *Robotics and Autonomous Systems*, 2008.

[6] K. Sjöö, H. Zender, P. Jensfelt, G.-J. M. Kruijff, A. Pronobis, N. Hawes, and M. Brenner, "The explorer system," in *Cognitive Systems*, H. I. Christensen, G.-J. M. Kruijff, and J. L. Wyatt, Eds. Springer, 2010.

[7] A. Bouguerra, L. Karlsson, and A. Saffiotti, "Handling uncertainty in semantic-knowledge based execution monitoring," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2007, pp. 437–443.

[8] S. Vasudevan, S. Gächter, V. Nguyen, and R. Siegwart, "Cognitive maps for mobile robots - an object based approach," *Robotics and Autonomous Systems*, vol. 55, no. 5, pp. 359–371, 2007.

[9] D. Joho and W. Burgard, "Searching for objects: Combining multiple cues to object locations using a maximum entropy model," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, May 2010, pp. 723–728.

[10] A. Aydemir, K. Sjöö, J. Folkesson, A. Pronobis, and P. Jensfelt, "Search in the real world: Active visual object search based on spatial relations," in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[11] M. Hanheide, C. Gretton, and M. Göbelbecker, "Dora, a robot exploiting probabilistic knowledge under uncertain sensing for efficient object search," in *Proceedings of Systems Demonstration of the 21st International Conference on Automated Planning and Scheduling (ICAPS)*, Freiburg, Germany, June 2011.