KNOWROB-MAP – Knowledge-Linked Semantic Object Maps

Moritz Tenorth, Lars Kunze, Dominik Jain and Michael Beetz
Intelligent Autonomous Systems, Technische Universität München
{tenorth, kunzel, jain, beetz}@cs.tum.edu

***Abstract*— Autonomous household robots are supposed to accomplish complex tasks like *cleaning the dishes* which involve both navigation and manipulation within the environment. For navigation, spatial information is mostly sufficient, but manipulation tasks raise the demand for deeper knowledge about objects, such as their types, their functions, or the way how they can be used. We present KNOWROB-MAP, a system for building environment models for robots by combining spatial information about objects in the environment with encyclopedic knowledge about the types and properties of objects, with common-sense knowledge describing what the objects can be used for, and with knowledge derived from observations of human activities by learning statistical relational models. In this paper, we describe the concept and implementation of KNOWROB-MAP and present several examples demonstrating the range of information the system can provide to autonomous robots.**

I. INTRODUCTION

Environment models or maps are resources that robots are equipped with or that they acquire in order to perform their tasks more reliably and efficiently. Being equipped with sub-symbolic maps, including occupancy grid and neurally implemented maps, the robot can typically infer its position, determine its destination, and then compute navigation plans to get there safely and fast. These decisions are based on the information stored in, or implied by, the map, such as whether or not the destination is reachable, and taken by inferring the appropriate action parameterizations from the map. Topological and object maps structure the environment model into meaningful pieces, such as distinctive places, objects, regions, gateways, etc. They thereby enable the robot to have semantic knowledge such as class labels for the map entities or for regions depending on the types of objects the regions contain [1]. Using these types of maps robots can in addition navigate "semantically" – go into a kitchen or position themselves relatively to objects in the map.

Even more expressive and powerful are *knowledge-linked semantic object maps*. In *knowledge-linked semantic object maps*, all object models in the map have a symbolic name that is visible in the associated knowledge base. Also, the data structures contained in the map are defined in terms of a terminological knowledge base – the categories of entities and their attributes. So, given these definitions robots can automatically translate the data structure of a semantic objects map into a set of facts in a symbolic knowledge base.

In this paper, we introduce KNOWROB-MAP, a system for the acquisition, representation, and use of knowledge-linked semantic object maps. As an example use case, let us assume the robot is looking for a device for heating food. Figure 1 visualizes the result of this query in two different environments. In the left kitchen, there is a regular oven, in the right one there is a microwave. However, the system can infer, based on its common-sense knowledge, that all specializations of *Oven*, like a *RegularOven* or a *MicrowaveOven*, can be used, and it can localize a suitable object in the map.

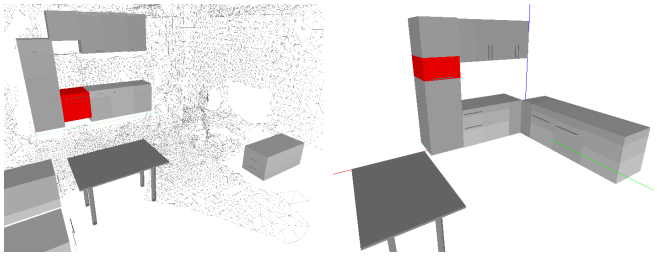

Fig. 1. Results of a query for a device for heating food. The environment model inferred that both a *RegularOven* (left) and a *MicrowaveOven* (right) can be used, and returns the respective object depending on the map. The abstract knowledge about objects is represented separately from the spatial information and can therefore easily be transferred to a new environment.

This paper makes the following contributions. First, we explain how we can link a semantic object model to a terminological knowledge base. Given this link we show how the robot can be equipped with encyclopedic, commonsense, and probabilistic knowledge about its environment. The consequence is that we can specify robot control programs that are more general in that they can automatically infer the right action decisions and parameterizations. Thus, the gain in using KNOWROB-MAP is that the robot can act more appropriately in more environments with less programming work. Thus, using KNOWROB-MAP the robot can carry out actions like turning off all heating devices or bringing clean glasses by inferring which cupboards they are probably stored in.

In the following sections, we will first give an overview of related work and describe the main concepts and important components of the system. We will then explain how the maps are represented and combined with encyclopedic knowledge, common-sense knowledge, and learned probabilistic models, before we describe the integration into the robot control program. A detailed scenario description provides examples of how the knowledge is used to accomplish a complex, under-specified task.

II. Related Work

Over the last years, several approaches to building "semantic maps" have been developed. Some of them detect rather coarse entities like the floor, the ceiling and walls in 3D laser scans [2], or focus on distinguishing different kinds of rooms based on the objects inside [1] or spatial relations between them [3]. Other methods, like [4], the one we are using as input in this system, or [5] detect, identify and localize objects in the scene and therefore create the basis for more abstract representations. However, these approaches only recognize and localize objects, but do not store further semantic information: Humans immediately associate various properties to something classified as a "cupboard", but robots do not have this knowledge. Without an explicit knowledge representation, different robots or even different parts of the same robot may have a very different notion of an object. Extending maps to provide such kind of knowledge is the goal of KnowRob-Map.

Deeper semantic representations, which also describe object properties like the point where to grasp or the opening of a bottle, are used by [6], but are mainly hand-coded and do not leverage the power of hierarchical, abstract knowledge representations. Galindo et al. [7] present a system for automatically building maps that combine a *spatial hierarchy* of local metric and global topological maps with a *conceptual hierarchy* that describes some semantic properties of rooms and objects. In that, their approach is similar to ours, but the conceptual hierarchy is much simpler and the spatial description much coarser. Zender et al. [8] present a system for coupling maps of recognized objects with a knowledge base, but with several limitations: They use a rather small, hand-crafted ontology compared to that in KnowRob-Map, and the map is only two-dimensional, which limits its use for robot manipulation.

III. System Overview

KnowRob-Map represents and reasons about semantic environment models by linking the output of an object recognition and mapping system to formally represented knowledge about the detected objects (Figure 2).

The maps are generated by the system described in [4] and semantically represented as part of the KnowRob knowledge processing system [9]. KnowRob provides the basic knowledge representation and reasoning capabilities, whereas KnowRob-Map adds the integration with the mapping system and object-related knowledge. The system is implemented in SWI Prolog using its Semantic Web Library for representing the robot's knowledge in the Web Ontology Language (OWL). OWL is a form of description logics and as such distinguishes between *classes* and *instances*. General knowledge about types of objects is modeled on the *class* level, whereas *instances* describe the actual objects in the map. Classes may be hierarchically structured and inherit the properties of their (potentially multiple) parents. *Roles* can link classes or instances and describe their properties.

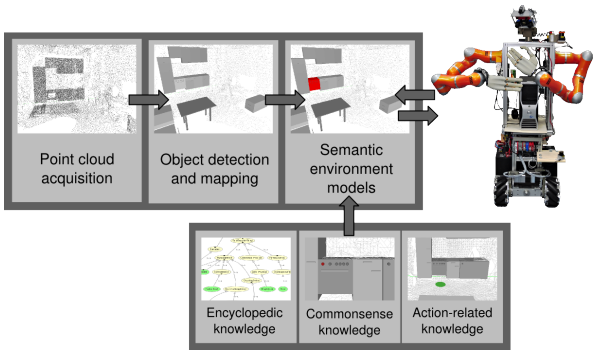

Fig. 2. Block diagram of the main components of KnowRob-Map. We extend an object detection, segmentation, recognition and mapping system with encyclopedic, action-related and common-sense knowledge to create semantic environment models. These can be queried by the robot control system for planning its actions.

IV. Map Representation

In KnowRob-Map, a *knowledge-linked semantic object map* consists of the semantic object map data structure and a schema that serves for translating this representation into a set of instances in the knowledge base. It gives the robot the ability to construct a semantic environment model out of its map data structures.

We define a schema for constructing *knowledge-linked semantic object maps* as a triple $SemObjMap = (D, O, R)$ consisting of a definition of the data structure produced by mapping system, an ontology describing the knowledge-based map representation, and a set of rules that translate between the two representations. This schema describes a whole class of knowledge-linked semantic object maps that can be instantiated once a specific map of an environment has been built. Applying the schema to the map data structures generates an ABOX (assertional box), i.e. a set of typed object instances in the knowledge base. Conceptually, this approach is similar to creating *views* in data bases: Views are a different representation of the same data, which can simply use different names for some of the fields, but can also perform complex combinations of multiple values.

Let us assume that the mapping system is able to describe objects in the environment by a unique identifier and a set of attribute-value pairs and create a data structure like the following:

obj-instance:	**37**	**obj-instance:**	**43**
class:	cupboard	class:	oven
width:	0.40	width:	0.60
depth:	0.35	depth:	0.60
height:	0.55	height:	0.74
xPos:	2.75	xPos:	1.29

This information is to be translated into a format compatible to the robot's knowledge base (Figure 3), creating a new view on this data. For this purpose, we define so-called *computable predicates*, a feature of KnowRob for loading external data into the knowledge representation during run-time. A computable predicate defines how the semantic properties (like volumeOfObject) can be calculated, in this case based on the information in the map:

```
owl_individual_of(Obj, Cl) :-
    class(Obj, Cl).
owl_has(Obj, kb:volumeOfObject, Vol) :-
    width(Obj, W), depth(Obj, D), height(Obj, H),
    Vol is W*D*H.
```

Using the computable predicates, the system creates typed object instances for each detected object and determines their properties, like the pose, dimensions, or sub-parts such as door handles. The following listing is an OWL description of a cupboard including links to its pose matrix (homography) and the associated door instance. An example of a complete map file is available on-line at `http://ias.cs.tum.edu/kb/ias_map.owl`.

```
<kb:Cupboard rdf:about="#cupboard31">
    <kb:depthOfObject>0.62</kb:depthOfObject>
    <kb:widthOfObject>0.3</kb:widthOfObject>
    <kb:heightOfObject>0.7</kb:heightOfObject>
    <kb:properPhysicalParts rdf:resource="#door7"/>
    <kb:pose rdf:resource="#rotmatrix3d_14"/>
</kb:Cupboard>
```

V. Encyclopedic Knowledge

The map representation is embedded into a hierarchy of classes and properties which describe and inter-relate these classes. The lowest level of the hierarchy assigns types to (parts of) objects in the environment, whereas the higher levels group these types into more and more abstract classes.

A small excerpt of the knowledge base, which only contains the taxonomy for some of the objects in our test environments, is given in Figure 3. The full ontology, which also describes actions and events, is available on-line at `http://ias.cs.tum.edu/kb/knowrob.owl`. The class structure is inspired by the OpenCyc ontology.

Due to the hierarchical structure, properties can be defined either specifically for e.g. cupboards, or generally for more abstract classes like containers. Compared to a flat representation, where such properties are directly assigned to the object instances in the map, a hierarchical structure reduces the amount of redundant information that has to be stored, helps keep the representation consistent, and facilitates changes.

Each class can have multiple parents, which correspond to the different semantic aspects of that class. For example, the class *Dishwasher* in Figure 3 is derived from *BoxTheContainer*, providing information about its shape and the fact that it can contain objects, from *CleaningDevice*, which indicates its main function, and from *ElectricalHouseholdAppliance*, which describes where such objects can be found and what they need to operate.

Since properties of object types are separated from the environment-specific map, they can easily be transferred to a new environment. The example in Figure 1 showed how the system can adapt to different environments, using the same common-sense knowledge.

VI. Common-Sense Knowledge

When dealing with an everyday object like a dishwasher, people have a natural understanding for what and how it can be used, and which problems might occur during its usage. To equip robots with similar knowledge, we extended KnowRob with common-sense knowledge which has been collected and made publicly available by the Open Mind Indoor Common Sense project (OMICS) [10], and is especially

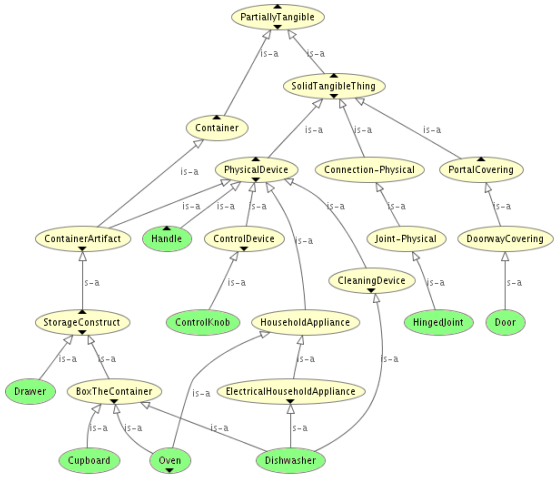

Fig. 3. Excerpt of the knowledge base taxonomy, with those concepts highlighted that are instantiated in the semantic environment map.

intended for the use in indoor mobile robotics. The knowledge was acquired from more than 3,000 voluntary Internet users and comprises more than 1.1 million statements.

These sources of knowledge complement each other: Cyc provides a broad categorization of things and dictionary-like descriptions, OMICS contains detailed action-related knowledge about everyday objects, e.g. that unloading a dishwasher will change its state from full to empty, or that dishes might be wet while unloading it.

The knowledge in OMICS is stored in natural language and has to be translated into a logic-based description to be integrated into KnowRob-Map. The first step in the automated procedure is to determine the meanings of a word using the WordNet [11] database. Afterwards, the system makes use of pre-existing mappings between WordNet and concepts in the Cyc ontology [12] to determine the formal ontological concept for a word. The concepts in Cyc are identical to those in KnowRob and can therefore be used here. Relations expressed in natural language are translated to predicates linking concepts in the knowledge base. For example, the *parts* relation in OMICS is translated to the *properPhysicalParts* relation in Cyc:

```
parts(dishwasher, motor)  =>
(( wordnetCyc(dishwasher, Dishwasher),
   wordnetCyc(motor, Engine))  =>
   properPhysicalParts(Dishwasher,Engine)).
```

This automated process can also translate more complex instructions like *place dishes onto the dishwasher rack* using the methods described in [13] for translating task descriptions from websites.

parts: PowerLine, Framework-SupportingStructure, SoapDish, Engine
locations: Kitchen
proximity: CookingRange, Sink, EatingVessel
actions: Loading, Unloading, TurningOnPoweredDevice, TurningOffPoweredDevice, UsingAnObject, EmptyingAContainer, OpeningSomething, ClosingSomething
states: OpenPortal, ClosedPortal, DeviceOn, DeviceOff, Full, Empty
causes: (Dishwasher,ClosedPortal)→(Dishwasher,DeviceOn)
(Dishwasher,DeviceOn)→(Bowl-Eating,Clean)
desires: (Sink,Full)→(Dishwasher,Empty)
(Dishwasher,DeviceOff)→(Dishwasher,DeviceOn)
state changes:
(Dishwasher,DeviceOff)→$_{\text{TurningOnPoweredDevice}}$(Dishwasher,DeviceOn)
(Dishwasher,Full)→$_{\text{Unloading}}$(Dishwasher,Empty)
responses: (Dishwasher,Full)→(TurningOnPoweredDevice)

(Dishwasher,Full)→(Unloading)
(Dishwasher,Empty)→(Loading)
problems: (Unloading,Dishwasher)→(DinnerPlate,Wet)
(Unloading,Dishwasher)→(EatingVessel,Dirty)
paraphrases: (Dishwasher,Loading)↔(Dishwasher,FillingProcess)
(Dishwasher,Unloading)↔(Dishwasher,EmptyingAContainer)
(Dishwasher,Unloading)↔(EatingVessel,Unloading)
reversibility: (Unloading)⊥(Loading)
(TurningOnPoweredDevice)⊥(TurningOffPoweredDevice)

Figure 4 depicts a small fraction of knowledge associated with the dishwasher and dishes, together with related actions and states. These relations were automatically extracted from OMICS and translated into formal expressions using the methods explained above.

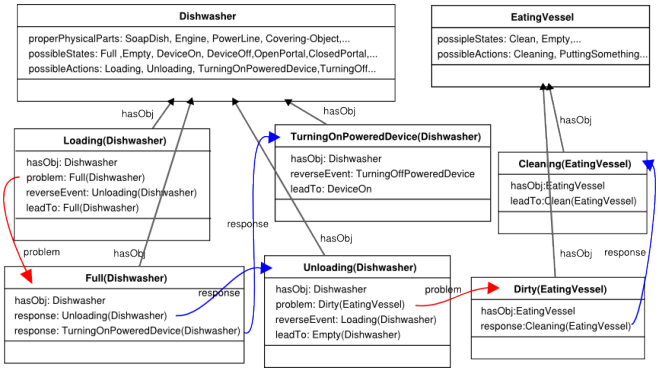

Fig. 4. Common-sense knowledge about a dishwasher integrated from OMICS. The diagram shows some relations, actions, states, and objects that are closely related with the concept *Dishwasher*. How the relation for potential problems can be used in robot control and how solutions can be determined is further explained in Section IX.

VII. Probabilistic Environment Models

With encyclopedic knowledge, we can represent that an object can potentially be or is definitely related to a number of other objects, but we cannot decide which objects are *most likely* to be related. However, there are many aspects of the environment that are subject to uncertainty, which, most importantly, concerns all aspects pertaining to humans' use of the environment. Therefore, we extend the knowledge representation with statistical relational models [14], which, in particular, can extend the semantics of description logics to a probabilistic setting [15].

For example, we are interested in modeling the properties of containers and appliances – e.g. which containers are likely to contain which types of objects (given their environmental context). Furthermore, since environments are inextricably linked to the activities performed within them, we also consider models that allow us to predict likely locations and states of objects based on the activities that are currently being carried out, or, inversely, to draw conclusions about activities given the locations of objects in the environment.

In KnowRob-Map, Bayesian Logic Networks (BLNs, [16]) are used to represent such models. In a nutshell, A BLN is a statistical relational model that represents general principles about a particular domain within a template model, which can be applied to instantiate a ground model representing a full-joint probability distribution given a particular set of entities that we are interested in (e.g. a particular set of kitchen devices and containers). Due to space constraints we cannot describe the models in more detail; please see [16] for more information.

As a concrete example, consider a BLN that models the probability with which containers in a kitchen environment contain objects of particular types – given the entire spatial context of the respective containers, including their proximity to devices such as the oven. A fragment network for this model is shown in Figure 5. This model can help the robot find storage locations for particular objects in a yet unknown environment.

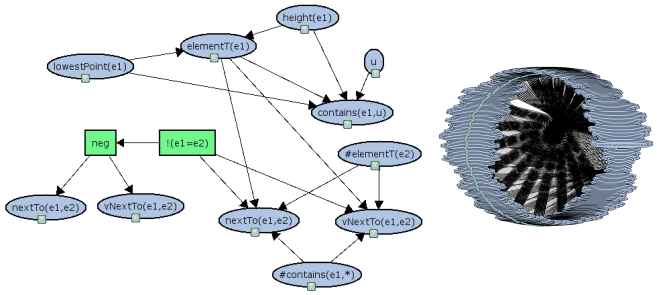

Fig. 5. Graphical template structure in a Bayesian logic network (left). This rather small (manually defined) template structure gets instantiated automatically into the large ground model with hundreds of nodes describing the actual objects in the environment (right).

VIII. Integration into the Robot Controller

Integrating knowledge processing into robot control programs is important for keeping control routines flexible and general, i.e. independent of the environment. Integration thereby means both providing methods for communication between the environment model and the planning system and actually using the knowledge when taking decisions. The first aspect is realized by providing a ROS service for language-independent access to KnowRob-Map. The robot plans, which are written in Lisp, use a wrapper library that allows to integrate queries to KnowRob-Map into the control flow of a robot:

```
(query-vars  (?handle ?pose)
(and (owl_individual_of ?heatingDevice HeatingDevice)
     (owl_has ?heatingDevice properPhysicalParts ?handle)
     (owl_individual_of ?handle Handle)
     (owl_has ?handle pose ?pose)))
(perceive ?handle ?pose)
(achieve (entity-gripped ?handle))
```

Here, the planning system queries for the pose of a handle of a heating device, parametrizes the perception system with the handle's estimated pose to restrict the search space, and finally sends the grasping command to the robot controller. Note that the binding of the variable *?heatingDevice* depends on the environment; Figure 1 shows that the result can be an instance of any subclass like a *RegularOven* or *MicrowaveOven*. Also, *properPhysicalParts* is a transitive relation, that is, it also finds handles attached to parts of the *?heatingDevice*, e.g. its door.

IX. Usage Scenario

KnowRob-Map is designed to support the robot in a large number of everyday household tasks. As such, its performance can best be measured by the range of queries it supports. In order to demonstrate its capabilities and to show

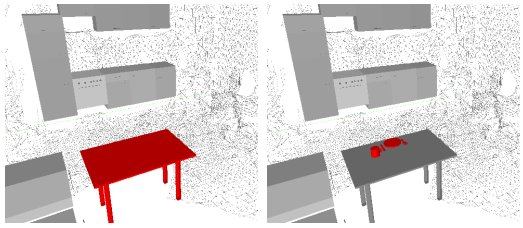

Fig. 6. Left: Instance of a table in the kitchen environment. Right: Dishes and silverware that are on the table.

Fig. 7. Left: Household appliance for washing dishes. Right: Entities and relations used as evidence for the probabilistic models.

how it can help a robot take decisions, we will describe one exemplary scenario that covers many facets of the system: A mobile household robot is asked to clear the table. There is no plan for this task in the robot's plan library, so the robot has to infer which actions to perform in which order, which objects to interact with in which way, and what to watch out for. We assume, however, that the robot has a set of parameterizable routines for low-level tasks like picking up an object as described in [17].

The information used in the following example was obtained from pre-existing, public resources, like OMICS and OpenCyc, in a completely automated way. In the (slightly simplified) queries, words starting with an uppercase letter denote variables, instance names begin with a lowercase letter, class names are written within single quotes, e.g. `'KitchenTable'`. The figures are visualizations of the content of the knowledge base, not simulations. All input data was acquired on the real robot platform using the methods described in [4] and [18]. Having received the command to clear the table, the robot queries its common-sense knowledge base for a description of the actions it needs to perform:

```
?- taskSubTasks('clear the table', Task, subTasks).
```

The OMICS database contains a large number of step-by-step instructions for common household tasks which we transformed as described in Section VI. Therefore, the above query returns the concept *ClearTheTable* as binding of the variable `Task`, and a list of actions bound to `subTasks`. Table I shows the natural language instructions obtained from OMICS and the formal representation in description logic that has been generated from them.

Since the plan importer directly maps verbs to action concepts, it often produces rather general concepts like *PuttingSomethingSomewhere* as translation of "put". For successfully executing the tasks, however, it is often helpful to have more specific plan descriptions. Therefore, the system searches in its action taxonomy for subclasses that have the properties given in the instruction, like *LoadingADishwasher* as a specialization of *PuttingSomethingSomewhere* with the *toLocation* being a *Dishwasher*, and replaces the general action class with the specific one.

TABLE I

TASK INSTRUCTIONS FOR: CLEAR THE TABLE

Natural Language	Description Logic
remove dirty dishes	RemovingSomething ⊓ **objActedOn.**EatingVessel
remove silverware	RemovingSomething ⊓ **objActedOn.**SilverwarePiece
put dishes into dishwasher	PuttingSomethingSomewhere ⊓ **objActedOn.**EatingVessel ⊓ **toLocation.**Dishwasher
remove table cloth	RemovingSomething ⊓ **objActedOn.**TableCloth

As described in [13], the system also comprises a plan generator that translates the action concepts in the knowledge representation into the respective goal statements in the robot's plan language. *RemovingSomething*, for instance, is mapped to a routine for picking up an object. Before the action can be executed, necessary action parameters have to be resolved. Some values, like the *objActedOn*, are directly specified in the instructions by the properties printed in bold in Table I. The following queries retrieve the object instances referenced by the *objActedOn* and *toLocation* properties from the environment model (Figure 6 and Figure 7 (left)).

```
?- owl_individual_of(Table, 'KitchenTable').
   Table = kitchentable1.
?- owl_individual_of(Obj,   'EatingVessel').
   Obj = cup3;
   Obj = plate1.
?- owl_individual_of(Obj,   'SilverwarePiece').
   Obj = fork1;
   Obj = knife1.
?- owl_individual_of(Obj,   'Dishwasher').
   Obj = dishwasher0.
```

Now the object references are resolved, but the robot is still missing a description for the *LoadingADishwasher* task. Again, the robot queries its common sense knowledge base to retrieve a sequence of actions to perform (Table II):

```
?- subTasks('LoadingADishwasher', subTasks).
```

After having resolved the references to the *RackOfDishwasher*, the robot has all the information to clear the table and load the dishwasher. In reality, however, there are lots of potential problems that can make the plan fail. Therefore, manually created plans include failure detection and recovery routines to verify that a task has been performed correctly. In the automatically created action sequences described here, such checks can be generated from the *problems* relation in OMICS, e.g. by querying for all problems related to a *Dishwasher*:

```
?- actionOnObject(Action,'Dishwasher'),
   owl_restriction_on(Action,
                      restriction(problem,some(Problem))).
   Action  = 'Loading_Dishwasher',
   Problem = 'Dishwasher_Full' ;
   Action  = 'Unloading_Dishwasher',
   Problem = 'DinnerPlate_Wet' ;
```

TABLE II

TASK INSTRUCTIONS FOR: LOAD THE DISHWASHER

Natural Language	Description Logic
collect dishes	Collecting ⊓ **objActedOn.**EatingVessel
open dishwasher	OpeningSomething ⊓ **objActedOn.**Dishwasher
pull out dishwasher rack	PullingAnObject ⊓ **objActedOn.**RackOfDishwasher
place dishes onto rack	PuttingSomethingSomewhere ⊓ **objActedOn.**EatingVessel ⊓ **toLocation.**RackOfDishwasher
push in dishwasher rack	PushingAnObject ⊓ **objActedOn.**RackOfDishwasher
close dishwasher	ClosingSomething ⊓ **objActedOn.**Dishwasher

Possible solutions or responses to these problems are also provided by the common-sense knowledge (Figure 4). The correct response clearly depends on the situation at hand: When the dishwasher is full of dirty dishes it should be turned on, whereas in the case of clean dishes it should be unloaded. Once the dishwasher has been loaded and cleaned the dishes, the robot is faced with the new task of returning the dishes where they belong. Given only the environment model of the kitchen, the robot has to sensibly choose locations at which the now clean objects should be placed. Looking for the storage location of a cup, a pot and cutlery, it might issue the following query to the probabilistic model described in Section VII,

$$\begin{aligned} &P(\text{contains(?c, Cup)}, \text{contains(?c, Pot)}, \text{contains(?c, Cutlery)} \mid L) \\ &\approx \langle\langle\ \text{TopCupboard}_3\text{: 0.51}, \text{TopCupboard}_1\text{: 0.23}, \ldots\rangle, \\ &\quad\langle\ \text{BottomCupboard}_1\text{: 0.65}, \text{TopCupboard}_3\text{: 0.36}, \ldots\rangle \\ &\quad\langle\ \text{Drawer: 0.75}, \text{TopCupboard}_2\text{: 0.16}, \ldots\rangle\rangle \end{aligned}$$

where L is the complete environment specification of the kitchen, involving the types of all containers and appliances as well as their vertical and horizontal neighborhood relations (see Figure 7 right). If the result is ambiguous, the robot can select among the candidate cupboards depending on other objects they contain. If we assume that similar objects are usually placed together, the "semantic similarity" of the concepts in the ontology can be a useful hint. We used the *wup* similarity measure [19] that is defined as

$$sim(C_1, C_2) = \frac{2 \cdot d(S)}{d_S(C_1) + d_S(C_2)}$$

where S is the least common superconcept of C_1 and C_2, $d(C)$ is the (lowest) depth of concept C in the ontology, and $d_S(C)$ is the (lowest) depth of concept C in the ontology when taking a path through superconcept S of C. Table III shows some examples of objects and their similarity to cups, cooking pots, and cutlery.

X. Conclusions

In this paper, we described how environment models can be enriched with knowledge to become more than just maps of obstacles: By linking recognized objects to encyclopedic and common-sense knowledge obtained from large, publicly available knowledge bases, the system can provide a robot with information about what these objects are, what they can use used for, and how to use them. We presented methods for describing the map in a way that it can be related with different kinds of knowledge, for acquiring general object-related knowledge as well as probabilistic models of object locations, and finally demonstrated the usefulness of the approach in an extensive example scenario.

TABLE III
Concept similarity based on the KnowRob-Map ontology.

	glass	plate	salad bowl	platter	knife	spatula
Cup	**0.78**	0.67	0.67	0.67	0.52	0.52
Pot	0.67	0.67	0.67	0.67	0.6	0.7
Cutlery	0.58	0.58	0.58	0.58	**0.78**	0.76

	cakepan	colander	pasta	cereals	mop	detergent
Cup	0.67	0.53	0.5	0.53	0.53	0.53
Pot	**0.78**	0.7	0.5	0.53	0.6	0.6
Cutlery	0.6	0.6	0.48	0.5	0.6	0.6

XI. Acknowledgements

This work is supported in part within the DFG excellence initiative research cluster *Cognition for Technical Systems – CoTeSys*, see also `www.cotesys.org`. We would like to thank the Honda Research Institute USA Inc. for providing the OMICS data.

References

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

[2] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Journal of Robotics and Autonomous Systems (JRAS), Special Issue on Semantic Knowledge in Robotics*, vol. 56, no. 11, pp. 915–926, 2008.

[3] R. Triebel, O. Mozos, and W. Burgard, "Relational learning in mobile robotics: An application to semantic labeling of objects in 2D and 3D environment maps," in *Studies in Classification, Data Analysis, and Knowledge Organization*, C. Preisach, H. Burkhardt, L.Schmidt-Thieme, and R.Decker, Eds. Springer-Verlag, 2008, pp. 293–300.

[4] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D Point Cloud Based Object Maps for Household Environments," *Robotics and Autonomous Systems Journal*, 2008.

[5] S. Ekvall, D. Kragic, and P. Jensfelt, "Object detection and mapping for service robot tasks," *Robotica: International Journal of Information, Education and Research in Robotics and Artificial Intelligence*, vol. 25, no. 2, pp. 175–187, March/April 2007.

[6] K. Okada, M. Kojima, S. Tokutsu, T. Maki, Y. Mori, and M. Inaba, "Multi-cue 3D object recognition in knowledge-based vision-guided humanoid robot system," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2007.*, pp. 3217–3222, 2007.

[7] C. Galindo, J.-A. Fernández-Madrigal, J. González, and A. Saffiotti, "Robot task planning using semantic maps," *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 955–966, 2008.

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

[9] M. Tenorth and M. Beetz, "KnowRob — Knowledge Processing for Autonomous Personal Robots," in *IEEE/RSJ International Conference on Intelligent RObots and Systems.*, 2009.

[10] R. Gupta and M. J. Kochenderfer, "Common sense data acquisition for indoor mobile robots," in *Nineteenth National Conference on Artificial Intelligence (AAAI-04*, 2004, pp. 605–610.

[11] C. Fellbaum, *WordNet: an electronic lexical database*. MIT Press USA, 1998.

[12] C. Matuszek, J. Cabral, M. Witbrock, and J. DeOliveira, "An introduction to the syntax and content of Cyc," *Proceedings of the 2006 AAAI Spring Symposium*, pp. 44–49, 2006.

[13] M. Tenorth, D. Nyga, and M. Beetz, "Understanding and Executing Instructions for Everyday Manipulation Tasks from the World Wide Web." in *IEEE International Conference on Robotics and Automation (ICRA).*, 2010.

[14] L. Getoor and B. Taskar, *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.

[15] P. C. G. da Costa, M. Ladeira, R. N. Carvalho, K. B. Laskey, L. L. Santos, and S. Matsumoto, "A first-order bayesian tool for probabilistic ontologies," in *FLAIRS Conference*, 2008, pp. 631–636.

[16] D. Jain, S. Waldherr, and M. Beetz, "Bayesian Logic Networks," IAS Group, Fakultät für Informatik, Technische Universität München, Tech. Rep., 2009.

[17] M. Beetz, F. Stulp, P. Esden-Tempski, A. Fedrizzi, U. Klank, I. Kresse, A. Maldonado, and F. Ruiz, "Generality and legibility in mobile manipulation," *Autonomous Robots Journal*, vol. 28, no. 1, pp. 21–44, 2010.

[18] D. Pangercic, M. Tenorth, D. Jain, and M. Beetz, "Combining perception and knowledge processing for everyday manipulation," in *IEEE/RSJ International Conference on Intelligent RObots and Systems.*, Taipei, Taiwan, October 18-22 2010.

[19] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics Morristown, NJ, USA, 1994, pp. 133–138.