Exchanging Action-related Information among Autonomous Robots

Moritz Tenorth and Michael Beetz

Intelligent Autonomous Systems Group
Department of Informatics, Technische Universität München
{tenorth,beetz}@cs.tum.edu

Abstract. In this paper, we describe representations and inference techniques that are used in the RoboEarth system for the web-based exchange of information between robots. We present novel representations for environment maps that combine expressive semantic environment models with techniques for selecting suitable maps from the web-based RoboEarth knowledge base. We further propose techniques for improving class-level object models with additional information as needed for distributed learning of object properties. In an integrated experiment, we show that the system enables robots to perform mobile manipulation tasks including the retrieval of suitable environment maps and the estimation and exchange of object property information.

1 Introduction

The RoboEarth system [1] is designed as a web community by robots for robots: Similar to community-driven web pages for humans, such as Wikipedia, which enable humans to exchange knowledge among each other, RoboEarth allows knowledge exchange between robots. When one robot has learned how to perform a task or has created an object model, it can upload this information to RoboEarth. Other robots can later download it and do not have to learn everything on their own again.

Since the platform is intended to serve for exchanging various kinds of information among heterogeneous robots with different capabilities, the development of appropriate techniques for finding relevant information becomes a challenging problem. When humans search for information on the Web, they can easily distinguish relevant from irrelevant information (e.g. search results caused by ambiguous meanings of the search terms) and filter out information they cannot make use of (e.g. text written in a language they do not understand). Realizing these capabilities on a robot requires powerful knowledge representation and reasoning techniques.

A central part of the RoboEarth project is thus the development of an expressive, formal representation language to encode the exchanged information. It helps the robot answer questions such as: Which descriptions exist for the task to be performed? Which of them can be used given the robot's hardware and software configuration? Are all required capabilities available? Which object models

are needed to perform the task? Are there maps that describe the environment the robot is to operate in, which kinds of information do they provide?

Autonomously taking these decisions requires access to meta-information that describes which kind of knowledge is encoded in which format, including for instance which coordinate systems are used for spatial information or which capabilities are needed to perform a task. This requires more comprehensive representations than commonly used for describing actions, objects, or environment structures in robotics. Related approaches usually focus on the representation of the information content, e.g. the spatial information in a map, but do not describe meta-information, e.g. *which* environment is actually described in that map. Hierarchical Task Networks (HTN [2]) and related plan languages are similar to the action representation used in RoboEarth but focus on the description of the task itself, i.e. its sub-actions, goals, and ordering constraints. XABSL [3], mainly used in the RoboCup soccer context, describes actions in terms of hierarchical finite state machines. AutomationML [4] is a standard for describing task information and spatial configurations, mainly used in industrial applications. The FIPA [5] standard primary deals with the definition of communication standards for software agents. Object description formats like the proprietary DXF [6] or the open Collada [7] standard describe objects using meshes and textures, but without further specifying semantic properties.

In [8], we gave a general overview of the representation language used in RoboEarth. In this paper, we will first describe the work-flow for downloading information from RoboEarth to show how the reasoning processes are integrated into the overall system. Then, we will focus on two specific reasoning problems, namely the representation and discovery of map information in RoboEarth, and the retrieval, improvement and sharing of object models.

The main contributions of this paper are the following: We present novel representations for environment maps that combine the expressive semantic environment models introduced in [9] with explicit descriptions of the environment that is described in the map, and with techniques for selecting suitable maps from a web-based database. We further present novel techniques for improving class-level object models with additional information as needed for distributed learning of object properties.

2 Retrieving Information Needed for a Task

A common scenario in RoboEarth is that a robot is given a task that it does not have a plan for. This means the robot needs to obtain an action plan and identify and retrieve missing components such that it, in the end, has all information it needs to successfully accomplish the task.

We assume that the robot has initial knowledge about the environment it is operating in, for example the address of the room (see Section 3.3), and a semantic model of itself, including its hardware and software components and abstract capabilities. This self-model needs to be created only once for each kind of robot and is described using the Semantic Robot Description Language

(SRDL [10]). When the robot acquires additional capabilities, e.g. by downloading an object model from RoboEarth, it adds them to its self-model so that they are available for future queries.

Figure 1 outlines the main inference steps involved in the retrieval of information. These inferences are performed before the robot starts to execute the task and ensure the availability of all components that are required for this task. First, the robot checks whether it has a map of the environment at hand and, if not, downloads one from RoboEarth to obtain information about free space and obstacles as well as about objects in the environment. The next step is to search for task descriptions, called "action recipes", that describe on the one hand which actions need to be performed, and, on the other hand, list dependencies the task has in terms of components and capabilities. The robot matches this specification against its self-model and iterates over all missing components to check whether they can be obtained from RoboEarth. If all missing components can be provided, the system converts the action recipe into an executable action plan in the CRAM plan language (CPL, [11]) and sends it to the CRAM executive.

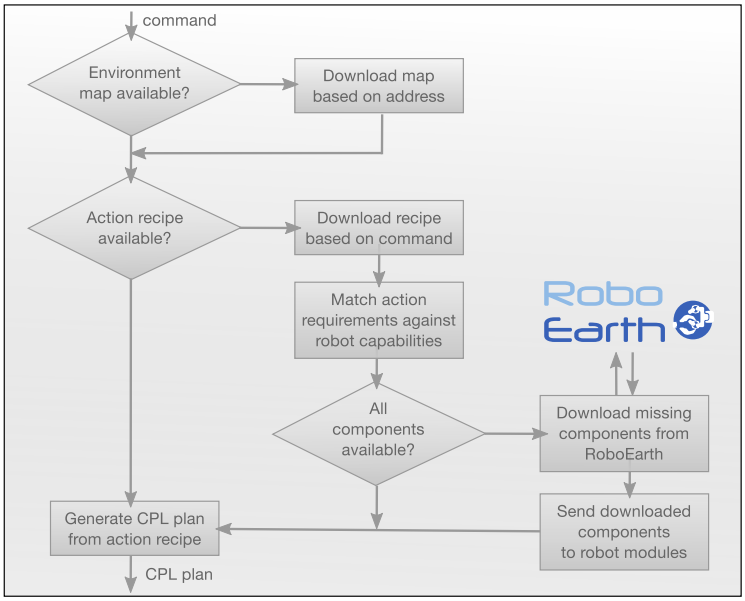

Fig. 1. Flowchart describing the process for downloading information from RoboEarth. Based on the command to perform a task, the robot determines which information it needs and downloads the required descriptions from the RoboEarth knowledge base.

The representation language [8] represents knowledge using the Web Ontology Language (OWL) as an extension of the KnowRob knowledge base [12].

The ontologies and software modules are available online as open source software.[1]

3 Representation of Environment Information

Figure 2 visualizes different kinds of environment maps that can be described and exchanged via RoboEarth. Their descriptions consist of two parts: the representation of the map's content, described in Section 3.1, and of the environment that is described in this map, which will be introduced in Section 3.3.

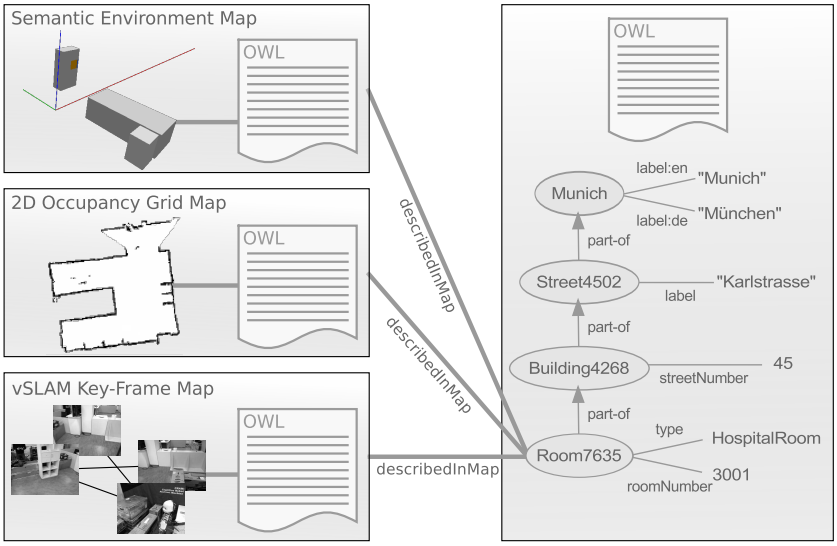

Fig. 2. Representations of different kinds of environment maps. Each map consists of an OWL description of its type and properties and optionally a binary file. In the case of semantic maps, the OWL description also describes the objects that are part of the map. The map is linked to a description of the environment it describes via the *describedInMap* relation.

3.1 Types of environment maps

For all tasks that include navigation or interaction with objects, a robot needs an environment model to plan its actions. Depending on the action to be performed, different kinds of maps are needed: Occupancy grid maps discriminate between free space and obstacles for localization, navigation and obstacle avoidance. Semantic environment maps contain localized object instances and can be used for grounding abstract object descriptions contained in an action recipe.

[1] Online: http://ros.org/wiki/knowrob, http://ros.org/wiki/re_ontology

Maps are described by an OWL specification of its type and properties that can optionally be linked to binary files. After download, the OWL description is parsed, which creates an instance of the respective type of map in the knowledge base such that the robot 'knows' that the map is available. Depending on the type of the map, the system can decide how to proceed after download: For example, a serialized occupancy grid is sent to the localization and navigation routines.

3.2 Spatio-temporal Object Pose Representation

Semantic environment maps consist of a set of typed, localized object instances. Based on the type, one can retrieve semantic information about the objects that is described at the class level, for example descriptions of physical parts, articulation properties, or CAD models to be used for recognition, spatial reasoning and visualization. Section 4 describes in detail how these properties are described for object classes.

To account for the dynamics in the environment and in order to be able to describe changes in the world, introduced by the robot itself or external effects, the representation of object instances can store and reason about changes in object poses. Multiple detections at different poses can be attached to an object instance; each of them is annotated with a time point and the perception method that has been used. This spatio-temporal object representation can be generated automatically from perceptual information and provides a flexible way of integrating novel information about objects into the knowledge base. It is described in more detail in [8].

3.3 Discovery of suitable maps for the robot's environment

In order to exchange maps with other agents, a robot needs to be able to describe which environment the maps belong to. This information is very important in the RoboEarth setup as it allows to specify which maps are desired when sending a request to the knowledge base. Since environments can be described in many different ways, the representation should support a spatial hierarchy (city – street – building), a semantic hierarchy (room – kitchen), and different labels for the same room.

We have developed a flexible approach that is similar to the human way of describing an address: Maps are annotated with data like the city, street, building, floor, room number or room type they describe, as shown in the right part of Figure 2. These components are linked by a transitive part-of relation. This allows to query for combinations of these levels, e.g. to search for all maps of a kitchen in Karlstrasse in Munich, or for all rooms on the third floor of Karlstrasse 45. It further allows to combine labels, such as room numbers, with types of rooms (private homes usually do not have room numbers) and to attach multiple labels to the same physical entity (Karlsplatz and Stachus are two names for the same square in Munich).

The representation can be combined with other techniques for searching for environment maps: Searching by GPS coordinate would be useful for field robotics, while there is normally no GPS signal available indoors. Alternatively, one can also assume that the robot is continuously localized (quantitatively or qualitatively) in a hierarchy of metric and topological maps, and that it can detect when it runs out of one map. In this case, it can search for an extension map describing the new area. This method, however, requires modification of the lower-level map management infrastructure.

4 Retrieving and Improving Object Models

While the information a robot perceives is at first environment-specific, there are often generic pieces of information that can be extracted from it. For example, if the robot estimates the articulation properties of an object [13], this information is a priori specific to the object instance the robot was interacting with. If the object, however, is an instance of a common type, it may make sense to extract the generic information and share it with other robots. For this reason, we have developed methods to generate class-level object models, taking a specific object instance as an example.

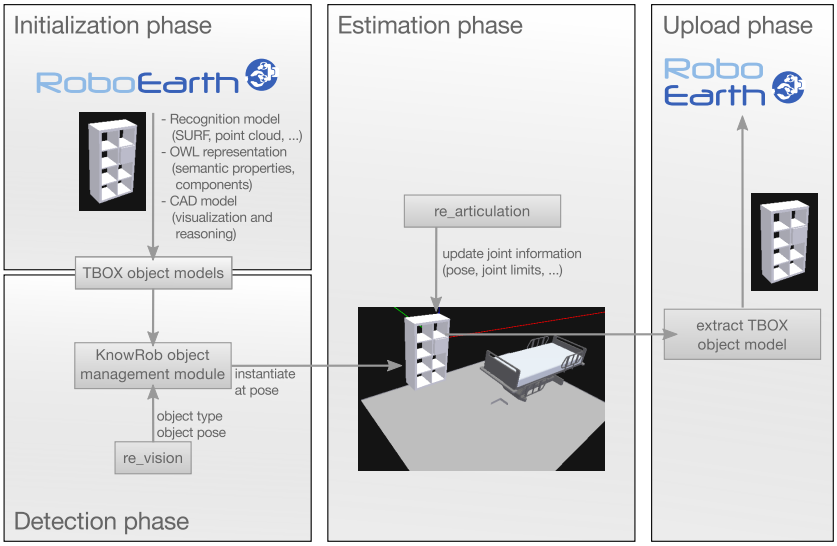

Fig. 3. Schematic overview of the object model up- and download. Robots can retrieve an object model from RoboEarth, instantiate the model if the object has been perceived in the environment, estimate additional properties, and upload a model generated from this object instance to share the new information with other robots.

Figure 3 shows the life cycle of an object model: During the initialization phase, shown in Figure 1, the model is downloaded from RoboEarth. If the

robot detects the object in the environment, the abstract class-level model is instantiated at the respective position, recursively translating all object-intrinsic coordinates into global map coordinates. During interaction, the robot estimates object properties like the positions and types of joints and adds them to the object instance. After finishing the task, it checks which pieces of information have been updated and for which sharing makes sense.

For these objects, it extracts a class-level (TBOX) model and uploads it to the RoboEarth knowledge base. The RoboEarth language supports the explicit specification of coordinate frames (e.g. map-global, relative to other objects or other components of the same objects, or qualified using frames in the ROS tf system[2]), and provides methods for the conversion between coordinates in different frames. By recursively converting the coordinates and translating instance-level descriptions into class restrictions, the system can create an object model that includes the new information.

5 Experiments

In a recent experiment, we have investigated how these methods can enable robots to perform a drink serving task in a previously unknown environment. The experiment has been done in two locations with two different robot platforms, a PR2 and the Amigo robot. Both robots were only equipped with the command and the address of the environment they were operating in and successfully downloaded all information required for the task. Compared to prior experiments [8], the robots thus had to operate with less initial information and perform more inference to find, select and apply information from the RoboEarth knowledge base.

The upper part of Figure 4 shows the environment maps that have been downloaded from RoboEarth. RoboEarth offers a SerQL [14] query interface.[3] The following query has been used to download the map information:

```
select source from context source {R}
  kr:describedInMap {S} ;
  kr:roomNumber {N}
  where N like "3001"
  using namespace
  kr=<http://ias.cs.tum.edu/kb/knowrob.owl#>;
```

Based on this map, the robots (Figure 4 bottom) could navigate to the appropriate positions and locate the objects required for the task. The action recipe to be used was selected using the following query:

```
select source from context source {A}
  rdfs:label {"serve a drink"^^xsd:string}
  using namespace
  rdfs=<http://www.w3.org/2000/01/rdf-schema#>
```

The action recipe was then matched against the robots' capabilities with the result that all required capabilities were available, but some recognition models

[2] http://ros.org/wiki/tf
[3] http://api.roboearth.org

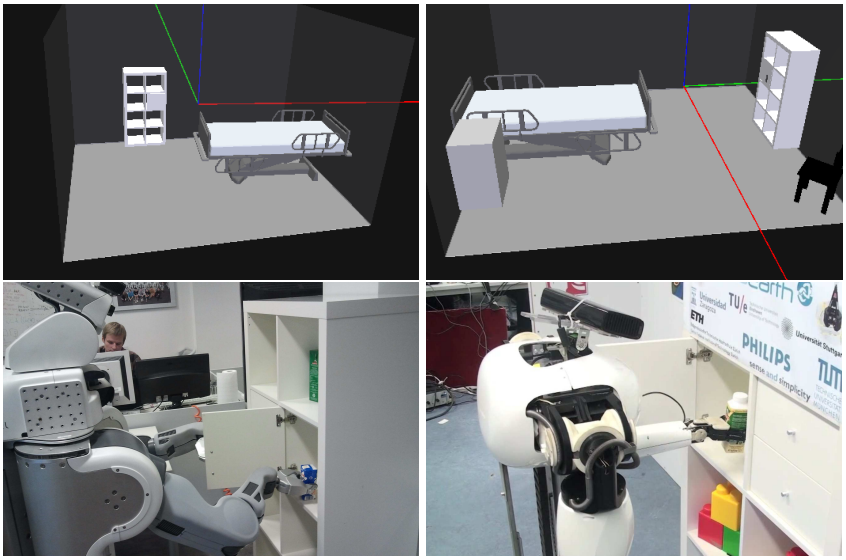

Fig. 4. Top: Semantic environment maps of the two hospital rooms, downloaded from RoboEarth based on the address and room number. Bottom: PR2 and Amigo robots opening the cabinet and picking up the drink to be served.

for some of the objects mentioned in the task were missing (namely the bottle and the bed), which have been downloaded, including the CAD models shown in Figure 4. The following CPL plan was then generated from the action recipe:

```
(def-top-level-plan serve-a-drink ()
(with-designators (
      (bottle1 (object '((name bottle1)
                         (type drinking-bottle))))
      (bed1 (object     '((name bed1)
                         (type bed_piece-of-furniture))))
      (hand-pose-handover1  (location '((on ,bed1))))
      (robot-pose-handover1 (location '((to reach)
                                        (side :right)
                                        (loc ,hand-pose-handover1))))
      (arms-at101 (action '((type trajectory)
                            (pose ,hand-pose-handover1)
                            (side :right))))
      (unhand-action102 (action '((type open-gripper)
                                  (side :right))))
)

(achieve '(object-in-hand ,bottle1 :right))
(at-location (robot-pose-handover1))
(achieve '(arms-at ,arms-at101))
(achieve '(arms-at ,unhand-action102))))
```

This experiment shows that the representation and reasoning mechanisms in RoboEarth can be used to represent and select information about human environments and mobile manipulation tasks. They have successfully been used by two heterogeneous robots in different environments who both performed the same mobile manipulation task.

6 Discussion and Conclusions

In this paper, we presented novel methods for exchanging information between robots with a focus on two applications: the exchange of environment maps and of object models. Regarding maps, we discussed the representation of the different kinds of maps and the relation between the maps themselves and the environments they describe. These techniques allow easy discovery of information in the database using only information that can realistically be assumed to be available on the robot. Object models are described by a combination of a semantic description in OWL with (binary) recognition models and support the representation of a component hierarchy as well as articulation properties. We showed that object class descriptions can be downloaded and be instantiated once the corresponding object has been detected in the environment. Based on these instances, the robot can estimate additional information, attach this information to the object, and share the newly acquired information with other robots. The presented methods significantly raise the level of semantics and are a big step towards robots that autonomously exchange information. Challenges to be addressed in the future include the question of which pieces of information are actually worth being exchanged.

Acknowledgments

This work is supported in part by the EU FP7 Project *RoboEarth* (grant number 248942).

References

1. M. Waibel, M. Beetz, R. D'Andrea, R. Janssen, M. Tenorth, J. Civera, J. Elfring, D. Gálvez-López, K. Häussermann, J. Montiel, A. Perzylo, B. Schießle, O. Zweigle, and R. van de Molengraft, "RoboEarth - A World Wide Web for Robots," *Robotics & Automation Magazine*, vol. 18, no. 2, pp. 69–82, 2011.
2. K. Erol, J. Hendler, and D. Nau, "Htn planning: Complexity and expressivity," in *Proceedings of the National Conference on Artificial Intelligence*. John Wiley & Sons LTD, 1994, pp. 1123–1123.
3. M. Loetzsch, M. Risler, and M. Jüngel, "XABSL-a pragmatic approach to behavior engineering," in *Proceedings of IEEE/RSJ International Conference of Intelligent Robots and Systems (IROS)*. Citeseer, 2006, pp. 5124–5129.
4. R. Drath, A. Luder, J. Peschke, and L. Hundt, "AutomationML-the glue for seamless automation engineering," in *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*. IEEE, 2008, pp. 616–623.
5. P. O'Brien and R. Nicol, "FIPA – towards a standard for software agents," *BT Technology Journal*, vol. 16, no. 3, pp. 51–59, 1998.
6. D. Rudolph, T. Stürznickel, and L. Weissenberger, *Der DXF-Standard*. Rossipaul, 1993.
7. R. Arnaud and M. Barnes, *COLLADA: sailing the gulf of 3D digital content creation*. AK Peters, Ltd., 2006.

8. M. Tenorth, A. C. Perzylo, R. Lafrenz, and M. Beetz, "The roboearth language: Representing and exchanging knowledge about actions, objects, and environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, MN, USA, May 14–18 2012, accepted for publication.
9. M. Tenorth, L. Kunze, D. Jain, and M. Beetz, "KNOWROB-MAP – Knowledge-Linked Semantic Object Maps," in *10th IEEE-RAS International Conference on Humanoid Robots*, Nashville, TN, USA, December 6-8 2010, pp. 430–435.
10. L. Kunze, T. Roehm, and M. Beetz, "Towards semantic robot description languages," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May, 9–13 2011.
11. M. Beetz, L. Mösenlechner, and M. Tenorth, "CRAM – A Cognitive Robot Abstract Machine for Everyday Manipulation in Human Environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, October 18-22 2010.
12. M. Tenorth and M. Beetz, "KnowRob – Knowledge Processing for Autonomous Personal Robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 4261–4266.
13. J. Sturm, C. Stachniss, and W. Burgard, "Learning kinematic models for articulated objects," *Journal on Artificial Intelligence Research (JAIR)*, 2011.
14. J. Broekstra and A. Kampman, "Serql: A second generation rdf query language," in *Proc. SWAD-Europe Workshop on Semantic Web Storage and Retrieval*, 2003, pp. 13–14.