Probabilistic Relational Reasoning in Semantic Robot Navigation

Walter Mayor Toro[1], Fabio G. Cozman[1], Kate Revoredo[2], and Anna Helena Reali Costa[1]

[1] Escola Politécnica, Univ. de São Paulo
Av. Prof. Luciano Gualberto, trav.3, 158, 05508-970 São Paulo, SP, Brazil
{walter.mayortoro,fgcozman,anna.reali}@usp.br
[2] Depto. de Informática Aplicada, Univ. Federal do Estado do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
katerevoredo@uniriotec.br

Abstract. We examine the use of semantic web resources in robot navigation; more specifically, in qualitative navigation where uncertain reasoning plays a significant role. We propose a framework for robot navigation that connects existing semantic web resources based on probabilistic description logics, with probabilistic relational learning and planning. We show the benefits of this framework in a real robot, presenting a case study on how semantic web resources can be used to face sensor and mapping uncertainty in a practical problem.

Keywords: Semantic robotics, KnowRob system, probabilistic description logics, Bayesian networks.

1 Introduction

Recent experience has shown that applications in robotics can benefit from semantic information carrying commonsense facts [1, 3, 12, 13] One particular example of semantic knowledge system for robotics is the KnowRob package (Knowledge Processing for Autonomous Personal Robots) [15, 16]. KnowRob operates on ontology databases such as OMICS (indoor common-sense knowledge database) [4], mixing description logics [2] and Bayesian networks [11].

However, it is not always easy to effectively bring these semantic web resources into practical use, as it is necessary to combine semantic information and low-level data, and to handle uncertain sensors and incomplete maps. In this paper we propose a framework for qualitative robot navigation that uses the probabilistic description logic knowlege base in KnowRob to learn and reason at a relational level. We explore a scheme where higher level descriptions are used to reason at an abstract level. This has important advantages. First, it saves computation as it handles sparser representations. Second, it is a perfect match to the level at which knowledge is stored (that is, relations are used throughout). Third, the use of information in a higher level of abstraction allows

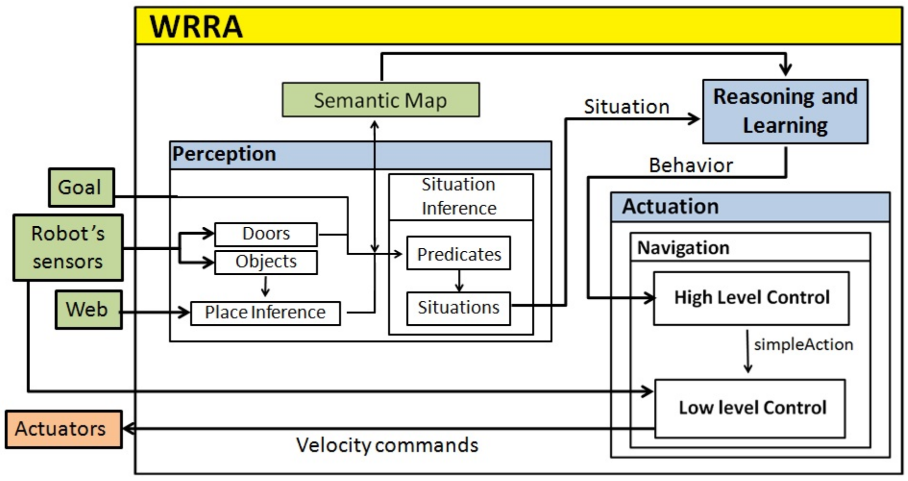

Fig. 1. Overview of the Web-based Relational Robotic Architecture.

knowledge to be generalized and transferred to other agents or used in similar tasks.

We also describe an implementation with a real robot that demonstrates how semantic web resources can work within applications that demand substantial uncertain reasoning. We present our knowledge representation strategy, our sensor gathering blocks, and our ground and abstract reasoning modules. Overall, our goal is to contribute with a case study on how semantic web resources can be used in practice. The paper is organized as follows. In Section 2 we give an overview of the different subsystems that our proposal combines. Section 3 presents the implementation of the system and our experiments. Section 4 concludes the paper.

2 A Framework for Robot Navigation

We consider a robot with three important sources of information. First, sensors that detect objects in the environment. Second, a map with semantic information, that is updated during navigation. Third, and most important, a web database with commonsense knowledge (for example, likely location of objects in rooms). In our framework, information processing flows through three major modules: Perception, Reasoning and Learning, and Actuation. We call the whole architecture by *Web-based Relational Robotic Architecture* (WRRA), as depicted in Figure 1. From the perspective of uncertain reasoning with semantic web resources, the Perception module (Section 2.1) is the most significant contribution of this paper. The other two modules are only briefly described as relevant information can be found in previous publications.

2.1 Perception: semantic information and probabilistic reasoning

This module receives a description of the *goal*. In the case study of interest here, the goal is to find a *target room* inside a house. The Perception module receives sensory information (detected objects), and must abstract the data into compact symbolic representations. Upon receiving data, the robot accesses its Semantic Web resources to determine the most likely room that generated the data, as described in this section.

Unlike most existing robotic systems, we pursue reasoning at a high level of abstraction, employing concepts, roles and relations between then as expressed within KnowRob. It is due to this decision that we can effectively employ semantic web resources. The *situation* of the robot is described in terms of a relational representation that not only allows for abstraction of metric and sensory details, but also enables knowledge to be generalized and reused in new tasks. During navigation, the robot uses sensor data to build a relational representation of the environment (the semantic map).

The output of the Perception module is a description of the robot's situation, which is specified by a conjunction of active predicates (with truth value TRUE) such as: `seeDoor`() that indicates that the robot sees one or more doors in the room; `seeNonVisitedDoor`($\mathtt{d}_1$), meaning that the robot sees door $\mathtt{d}_1$ that has not yet been visited; `inTargetRoom`(), which indicates that the target room is where the robot is; `nonTargetRoom`($\mathtt{p}_1$), meaning that the robot is in $\mathtt{p}_1$ and it is not the target room; `inRoom`($\mathtt{p}_1$) that indicates that `p` is the most likely room where the robot is; and others. The truth value of `inRoom`(`p`) is computed by *Place Inference* block, as we explain now.

The Perception module is heavily based on reasoning facilities available in the KnowRob package. The knowledge base in KnowRob uses rdf triples to represent a large ontology, with relationships between objects such as Drawer, a subclass of StorageConstruct, or Refrigerator – Freezer, a subclass of FurniturePiece [16]. Additionally, sentences in OWL indicate relationships between objects. Sentences are attached to probabilities, and for inference they are grounded into Bayesian networks using facilities in the ProbCog system [5].

Just as an example of rdf triple in the knowledge base, consider the fact, contained in the OMICS database, that a kitchen contains a refrigerator (`XXX` denotes the string `http://ias.cs.tum.edu/kb/knowrob.owl`):

```
<rdf:Description rdf:about="XXX#OmicsLocations-1">
    <ns1:object rdf:resource="XXX#Kitchen"></ns1:object>
    <ns1:subject rdf:resource="XXX#Refrigerator"></ns1:subject>
    <rdf:type rdf:resource="XXX#OmicsLocations"></rdf:type>
</rdf:Description>
```

The Perception module queries KnowRob, which returns, for each observed object, the probability that the location is each possible room, given the observed object. Queries are sent to KnowRob through Prolog sentences via function calls in the Python language; as an example, consider (a complete query is given in Section 3):

```
for obj in DetectedObjects:
    q="bayes_probability_given(knowrob:'OmicsLocations',
          Room,knowrob:'"+obj+"',Pr)"
    query = prolog.query(q)
```

Such a query returns probabilities such as

```
Room = 'knowrob.owl#Kitchen'
Pr = 0.1031101853182014 ;
```

That is, given a perceived object o_i, KnowRob uses inference with its probabilistic description logic [8, 7] to return $P(r_j|o_i)$ for each room r_j. The problem now is to combine these pieces of information into a probability that the robot is in room r_j, given *all* detected objects $o_1, \ldots, o_n$. We have:

$$\begin{aligned} P(r_j|o_1,\ldots,o_n) &= \frac{P(o_1,\ldots,o_n|r_j)P(r_j)}{P(o_1,\ldots,o_n)} \\ &= \frac{P(o_1|r_j,o_2,\ldots,o_n)P(o_2|r_j,o_3,\ldots,o_n)\ldots P(o_1|r_j)P(r_j)}{P(o_1,\ldots,o_n)}. \end{aligned}$$

We now assume that, given r_j, an observation (of an object) is independent of other observations (of other objects in the same room). Hence:

$$\begin{aligned} P(r_j|o_1,\ldots,o_n) &= \frac{P(o_1|r_j)P(o_2|r_j)\ldots P(o_1|r_j)P(r_j)}{P(o_1,\ldots,o_n)} \\ &= \frac{(P(r_j|o_1)P(o_1)/P(r_j))\ldots(P(r_j|o_n)P(o_n)/P(r_j))P(r_j)}{P(o_1,\ldots,o_n)} \\ &= \left(\prod_{i=1}^{n} P(r_j|o_i)\right)\frac{\prod_{i=1}^{n} P(o_i)}{P(o_1,\ldots,o_n)(P(r_j))^{n-1}}. \end{aligned}$$

We now introduce a substantive assumption, namely, that every room has identical a priori probability $P(r_j)$. So, $P(r_j|o_1,\ldots,o_n)$ is proportional to $\prod_{i=1}^{n} P(r_j|o_i)$. Once the Perception module gets, for each room, each term of this product from KnowRob, it compares each room with respect to this product, setting the truth value of `inRoom(p)` as TRUE for: $\mathtt{p} = \arg\max_{r_j} \prod_{i=1}^{n} P(r_j|o_i)$, and FALSE otherwise.

During navigation, a semantic map of the environment is created. Each visited room and each observed object are represented as vertices of a graph that describes the topological map (left side of Figure 2). Connectivity between rooms is represented by graph edges, which are defined through doors conecting the rooms. While this topological map is built, edges are created by connecting vertices of the topological map to vertices of the conceptual map (right side of Figure 2). Unlike other approaches [1, 3], our map does not involve metric representation of the environment. Still, our semantic map inserts probabilistic information in the representation. Every inference and reasoning in WRRA occurs at the level of objects, rooms and relationships and properties thereof.

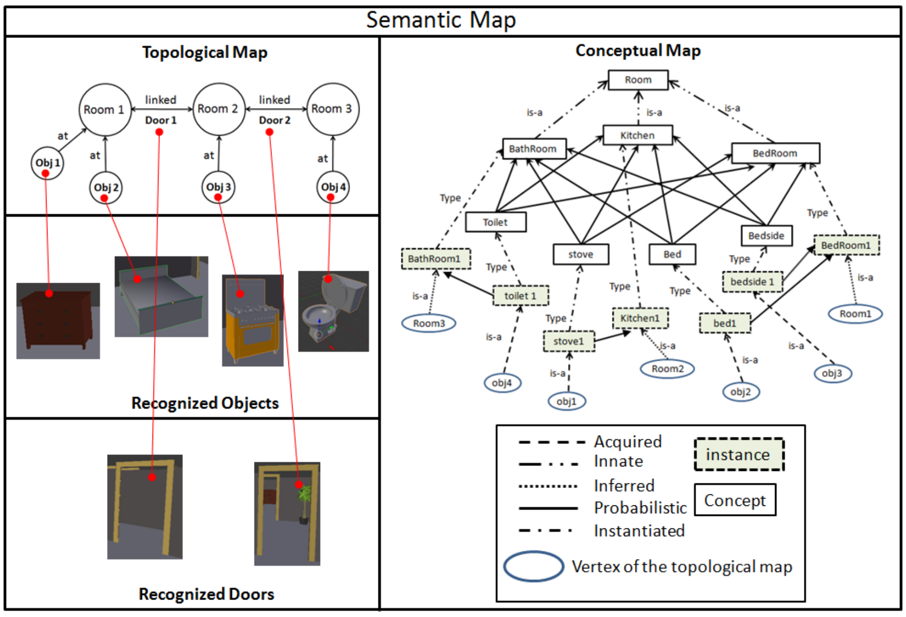

Fig. 2. The semantic map built by WRRA.

2.2 Reasoning and Learning, and Actuation

The WRRA uses reinforcement learning (RL) to refine behavior through interactions with the environment [14]. Typical RL solutions learn from scratch; we instead employ two levels of RL [6], where an abstract and a ground policy are learned simultaneously. The stochastic abstract policy learned in a source task is then used in new similar tasks. Our robot navigation problem is modeled as a Relational Markov Decision Process (RMDP) [10], in which situations $s \in \mathcal{S}$ are represented as a conjunction of predicates describing properties of and relations among objects, such as: $s_1 = \texttt{inRoom}(\texttt{livingroom}) \wedge \texttt{nonTargetRoom}(\texttt{livingroom}) \wedge \texttt{seeNoDoors}() \wedge \texttt{notAllDoorsVisited}()$. Other formalisms are possible to represent decisions and transitions [9].

A conjunction is a *ground conjunction* if it contains only ground atoms (such as s_1 given in the example). In our discussion each variable in a conjunction is implicitly assumed to be existentially quantified. An *abstract situation* σ (and *abstract behavior* α) is a conjunction with no ground atom. A relational representation enables us to aggregate situations and behaviors by using variables instead of constants in the predicate terms. For example, ground situation s_1 is covered by abstract situation σ by replacing *livingroom* with variable X; in this case, other situation could also be covered by σ, e.g., $s_1 = \texttt{inRoom}(\texttt{kitchen}) \wedge \texttt{nonTargetRoom}(\texttt{kitchen}) \wedge \texttt{seeNoDoors}() \wedge \texttt{notAllDoorsVisited}()$.

Denote by $\mathcal{S}_\sigma$ the set of ground situations $s \in \mathcal{S}$ covered by abstract situation σ. We assume that each ground situation s is abstracted to only one abstract situation σ. Similarly, we define $\mathcal{A}_\alpha(s)$ as the set of all ground behaviors $a \in \mathcal{A}$ covered by an abstract behavior α in ground situation s. We also define $\mathcal{S}_{ab}$ and $\mathcal{A}_{ab}$ as the set of all abstract situations and the set of all abstract behaviors in an RMDP, respectively. To simplify notation, here we use the assumption that if an atom does not appear in a ground sentence, the negated atom is assumed.

To solve an RMDP is to find an *optimal policy* π^* that maximizes a function R_t of future rewards. In RL tasks the agent does not know the dynamics of the process and a series of RL algorithms can be used to find a policy [14]. To translate from ground to abstract level, we define two operations: *abstraction* and *grounding*. Abstraction is the translation from the ground level (perceived by the robot's sensors) to the abstract level by replacing constants with variables, $\phi_s : \mathcal{S} \to \mathcal{S}_{ab}$. For a ground situation s, the corresponding abstract situation σ is given by $\phi_s(s) = \sigma$ so that $s \in \mathcal{S}_\sigma$. Grounding is the translation from the abstract level to the ground level, $a = \mathit{grounding}(\alpha, s)$. Clearly only ground states are sensed and visited by the robot, and only ground actions can be actually applied. Laerning and reasoning must proceed by processing, at time t, the (ground) experience $\langle s_t, a_t, r_t, s_{t+1}, a_{t+1}\rangle$, which is related to the tuple $\langle \sigma_t, \alpha_t, r_t, \sigma_{t+1}, \alpha_{t+1}\rangle$.

We propose the following scheme to apply an abstract policy in a ground problem. Consider a stochastic abstract policy defined as $\pi_{ab} : \mathcal{S}_{ab} \times \mathcal{A}_{ab} \to [0, 1]$. After the abstract situation $\sigma = \phi_s(s)$ is derived from the observed ground situation s, a transferred abstract policy (learned from source tasks) yields probabilities $\pi_{ab}(\sigma, \alpha_k) = P(\alpha_k|\sigma)$ for all $\alpha_k \in \mathcal{A}_{ab}$. We select an abstract behavior $\alpha_k \in \mathcal{A}_{ab}$ according to these probabilities. Then the process remains the same, with $a = \mathit{grounding}(\alpha_k, s)$.

Thus, in our system, the robot initially receives an abstract policy and applies it. As its knowledge about the new environment increases, due to its perception and action in the environment, the robot creates and improves a semantic map, which places restrictions on the actions defined by the policy initially received, adapting it to the new environment and to the new task. For example, consider the robot identifies it is in the living room, which is not the target room, and the living room has two doors, d_1 and d_2. The abstract policy indicates that it can randomly choose any one of the two doors and go through it, hoping to reach the target room. Assume the robot circulates in other rooms, after going through the chosen door, say d_1, and represents what is discovered about the environment in a semantic map. Upon returning to the living room without having reached the target room, the reasoning process now indicates that it should choose another door (d_2).

Finally, the Actuation module is divided into a High Level Control (HLC) and Low Level Control (LLC). HLC receives a behavior selected by the Reasoning and Learning module. The behavior is divided into simple actions that can be executed by specific hardware modules. Each simple action is sent to LLC, to be actually executed. Low-level commands are issued by the Actuation module.

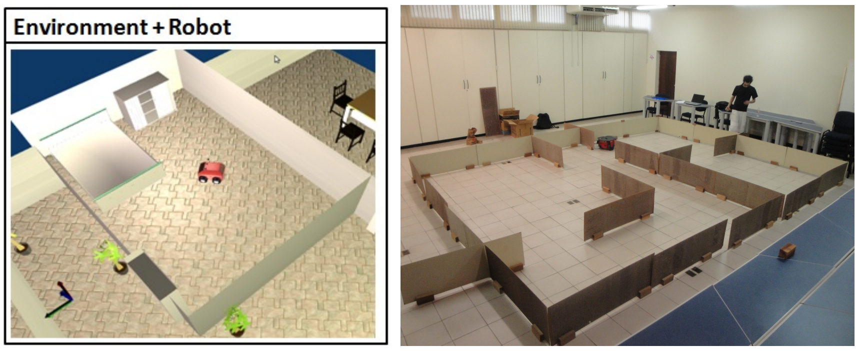

Fig. 3. *Left*: Simulated house. *Right*: Experimental setup with real robot.

3 Implementation and Discussion

We now describe our implementation and experiments. The robot we consider is a wheeled base equipped with 5 sensors: 3 semantic cameras, 1 odometer and 1 laser range scanner. We run tests both in a simulated environment and with the real robot. The simulated scenario (see Figure 3-*Left*) was designed with the open source tool for 3D creation, BLENDER[3], with which we have created a 3D CAD representation of the house and the objects it contains, including the robot (an ATRV 4-wheeled base); the representation was integrated into the MORSE simulator[4] and the Robot Operating System (ROS)[5]. The environment is a house that has eight types of rooms: 1 hallway (with some potted plants), 1 kitchen (with 1 stove, 1 fridge, and a dishwasher), 1 living room (with 1 sofa and 2 armchairs), 1 bathroom (with 1 toilet and 1 sink), 3 bedrooms (with 1 bed and 1 bedside), and 1 dining room (with 1 table and 6 chairs). The real robot is a Pioneer 2DX, and with the real robot we used QR codes to identify doors and objects, so as to obtain functionality similar to a semantic camera.

The semantic camera is, in essence, a sensor that allows to recognize objects that the robot sees and the relative position between the robot and objects viewed. The robot was equipped with two semantic cameras that recognize general objects and one semantic camera that recognizes only doors. The odometer and the laser scanner are used in the Navigation module.

For a better understanding of how the architecture WRRA works, how is its integration with the information of the semantic web and the technologies employed, we describe a simple case study executed in the simulated scenario, where we have great flexibility in defining tasks and measuring behavior. WRRA was implemented using the Python programming language and was integrated with the ROS framework. Initially, the robot knows nothing about the house

[3] http://www.blender.org/
[4] http://www.openrobots.org/wiki/morse/
[5] http://wiki.ros.org/

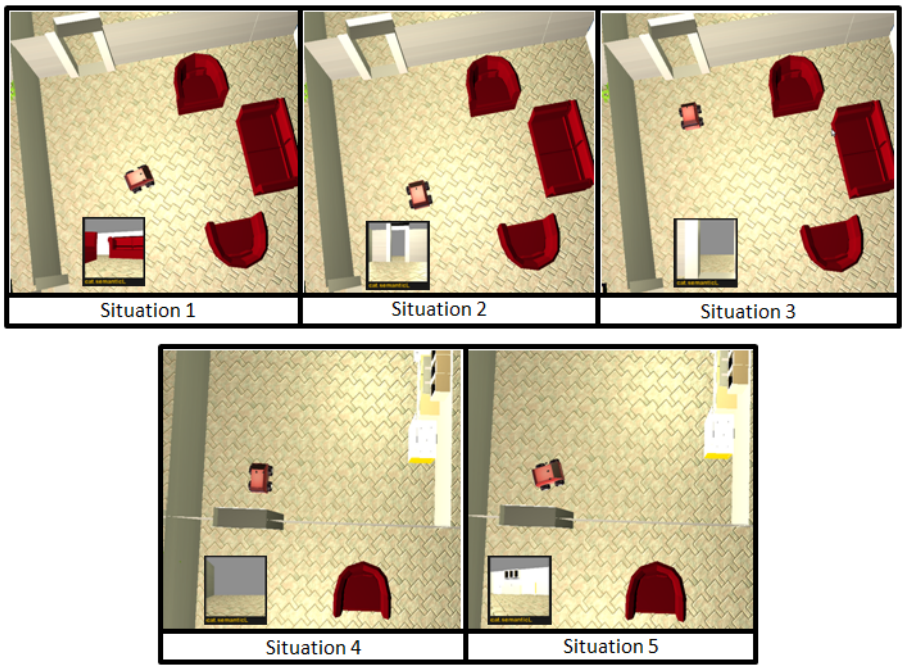

Fig. 4. Sequence of situations faced by the mobile robot in case study.

and has only a generic abstract policy that defines mappings from abstract situations into abstract behaviors, such as $\pi_{abs}(\texttt{inRoom(X)} \wedge \texttt{nonTargetRoom(X)} \wedge \texttt{seeNoDoors}() \wedge \texttt{notAllDoorsVisited}()) = \texttt{findDoor}()$, Indicating that if the robot is in a certain room that is not the target room and it did not detect any door in this room and the list of doors visited by it is empty, then the robot must find a door. The robot is given the goal of reaching the home kitchen. Then, the robot perceives situations, and selects the appropriate behavior for each situation and performs it, until the target room is reached. Figure 4 describes a sequence of five situations faced by the robot.

Situation 1: The Perception module collects information from the environment using the robot semantic cameras. From the position where the robot is, two objects are detected: $\texttt{obj}_1 = \texttt{sofa}$ and $\texttt{obj}_2 = \texttt{armchair}$. Then the Place Inference submodule performs a query to the integrated ROS library KnowRob-OMICS, which estimates the most likely room where the robot is taking into account the objects detected by the robot:

```
prolog = json_prolog.Prolog()
for obj in objets:
    q="bayes_probability_given(knowrob:'OmicsLocations',
    Room,knowrob:'"+obj+"',Pr)"
    query = prolog.query(q)
    for solution in query.solutions():
        room=str(solution['Room'])[37::]
        places.place[room].append(solution['Pr'])
query.finish()
```

As the queries are implemented using Python and the KnowRob-OMICS is implemented using the PROLOG logic programming language, the WRRA uses the ROS library JSON-PROLOG [6] to send the queries from Python code to PROLOG. When the KnowRob-OMICS is queried, it returns the following information for each recognized object:

```
Room = 'knowrob.owl#Kitchen'
Pr = 0.1031101853182014 ;
Room = 'knowrob.owl#DiningRoom'
Pr = 0.12185749173969258 ;
Room = 'knowrob.owl#BedRoom'
Pr = 0.12185749173969258 ;
Room = 'knowrob.owl#LivingRoom'
Pr = 0.3655724752190777 ;
Room = 'knowrob.owl#Hallway'
Pr = 0.14893693434851316 ;
Room = 'knowrob.owl#BathRoom'
Pr = 0.1386654216348226 ;
```

This information gives the probability that each room is the current location of the robot, given only one recognized object. Figure 5 shows some probabilities that a room type has certain object type. These probabilities come from the ROS library KnowRob-OMICS that uses the Lidstone's law, which redistributes the probability mass assigned to the seen tuples to the unseen tuples like (bed,bathroom). The Lidstone's law uses a parameter $\lambda < 1$ and when the parameter $\lambda \to 1$ much probability is distributed to unseen tuples [4]. In our experiments, we set $\lambda = 0.5$.

Then the Place Inference submodule uses the probabilistic reasoning process explained in Section 2.1 to infer the most likely place where the robot is by taking into account all objects recognized by the robot. In $situation_1$ of the case study reported here, the place inferred as the most likely place is $\mathtt{p}_1 = \mathtt{livingroom}$.

When objects and doors are detected and the place is inferred, the semantic map is updated. For this instance the map is updated with the objects $\mathtt{obj}_1$ and $\mathtt{obj}_2$ and the place $\mathtt{p}_1$ by building the relations $\mathtt{obj}_1$ at $\mathtt{p}_1$ and $\mathtt{obj}_2$ at $\mathtt{p}_1$. Next, the Inference Situation submodule receives information about detected doors, the inferred place and the updated semantic map. With this information, the truth values of the predicates are calculated and the conjunction of predicates with truth value TRUE forms a situation description using the SMACH library [7],

[6] http://wiki.ros.org/json_prolog
[7] http://wiki.ros.org/smach

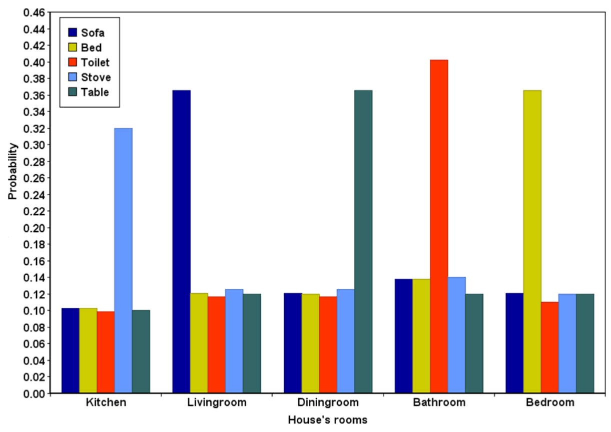

Fig. 5. Probabilities of room given observed object, by KnowRob-OMICS.

which is a ROS-independent Python library to build hierarchical state machines. Finally that inferred situation is the output of the Perception module that for this case is the $situation_1 = \texttt{inRoom(livingroom)} \wedge \texttt{nonTargetRoom(livingroom)} \wedge \texttt{seeNoDoors}() \wedge \texttt{notAllDoorsVisited}()$.

Then $situation_1$ is sent as input to the Reasoning and Learning module, and it is transformed in its corresponding abstract situation. The abstract policy is then used to define the output to the Actuation module: $\pi_{abs}(\texttt{inRoom(X)} \wedge \texttt{nonTargetRoom(X)} \wedge \texttt{seeNoDoors}() \wedge \texttt{notAllDoorsVisited}()) = \texttt{findDoor}()$ (see subsection 2.2). In the current version the output of this module is one of four behaviors: $\texttt{goToDoor(d)}$ meaning that the robot should go to $\texttt{d}$; $\texttt{goToNextRoom(p)}$ meaning that the robot should go into the next place $\texttt{p}$; $\texttt{findDoor}()$ meaning that the robot should search for doors in the room; $\texttt{exploration}()$ meaning that the robot should search for objects in the room.

Finally, in the Actuation module, the HLC submodule uses the ActionLIB ROS library to decompose each behavior into a sequence of simple actions, which are in turn translated by the LLC submodule to respective velocities of translation and rotation by using the Move Base ROS library [8], which allows the robot to reach a certain target pose. The Move Base library in turn uses other ROS libraries to avoid obstacles during navigation and to build a local path for the execution of each simple action sent by the HLC. Usually the Move Base library works with a static metric map of the environment, but in our case the Move Base library was set to work without it, since WRRA only reasons in relational level. The robot, controlled by the actuation module, search for a door in the environment. At the moment its semantics camera detects a door, a new situation is defined.

Situation 2: When the robot perceives a door (in this case, it sees door $\texttt{d}_1$), the Perception module updates the semantic map with the door $\texttt{d}_1$ and the

[8] http://wiki.ros.org/move_base

relation $\mathtt{d}_1$ at $\mathtt{p}_1$. So a new ground situation is determined by the Perception module: $situation_2 = \mathtt{inRoom(livingroom)} \land \mathtt{nonTargetRoom(livingroom)} \land \mathtt{seeDoors()} \land \mathtt{notAllDoorsVisited()} \land \mathtt{seeNonVisitedDoor(d_1)}$. This situation is turned into its corresponding abstract situation in the Reasoning and Learning module, and the abstract policy gives: $\pi_{abs}(\mathtt{inRoom(X)} \land \mathtt{nonTargetRoom(X)} \land \mathtt{seeDoors()} \land \mathtt{notAllDoorsVisited()} \land \mathtt{seeNonVisitedDoor(Y)}) = \mathtt{goToDoor(Y)}$. In this case, the grounding of behavior $\mathtt{goToDoor(Y)}$ gives $\mathtt{goToDoor(d_1)}$. Then, the Navigation module operates properly, using sensory information that gives the relative position of the robot to $\mathtt{d}_1$ and to obstacles, in order to drive the robot to the front door.

Situation 3: In this situation the robot knows it still is in the living room, it still sees the door $\mathtt{d}_1$ that has not been visited, and now it can see the adjacent room $\mathtt{p}_2$ through door $\mathtt{d}_1$. The map is updated by building the relation $\mathtt{p}_2$ connected to $\mathtt{p}_1$ through $\mathtt{d}_1$. This situation is:
$situation_3 = \mathtt{inRoom(livingroom)} \land \mathtt{nonTargetRoom(livingroom)} \land \mathtt{seeDoors()} \land \mathtt{notAllDoorsVisited()} \land \mathtt{seeNonVisitedDoor(d_1)} \land \mathtt{seeNextPlace(p_2)}$.
Given the abstraction of $situation_3$, the behavior indicated by the abstract policy is $goToNextPlace(Z)$ and the grounding of it results in $\mathtt{goToNextPlace(p_2)}$. In this case, the Navigation module drives the robot through the door and the robot reaches the adjacent room $\mathtt{p}_2$.

Situation 4: As the robot has not observed any object in this new room, then $\mathtt{p}_2 = \mathtt{unknown}$. In this case, the map does not need to be updated and the only predicate with truth-value TRUE is $\mathtt{inUnknownRoom()}$. Then the Reasoning and Learning module outputs the behavior $\mathtt{exploration()}$, meaning that the robot must explore the room, looking for objects.

Situation 5: Finally, the robot observes objects $obj_3 = \mathtt{oven}$ and $obj_4 = \mathtt{freezer}$ that allow inferring that it is in the $\mathtt{p}_2 = \mathtt{kitchen}$. Then the map is updated by bulding the relations obj_3 at $\mathtt{p}_2$, obj_4 at $\mathtt{p}_2$. As the kitchen is the target room, the episode ends and the task is fulfilled.

4 Conclusion

In this paper we have explored the use of semantic web resources to conduct probabilistic relational learning and reasoning in robot navigation. We have presented an architecture (WRRA) for robot navigation that employs the KnowRob system and its knowledge base of probabilistic description logic sentences, together with relational reinforcement learning. The resulting framework shows how to use, in practice, semantic web resources that can deal with uncertainty.

We have implemented the WRRA, first in a simulated environment, then in a real robot. The fact that the WRRA operates with abstract semantic information, both in its knowledge base, and in its inputs and outputs, simplifies the whole process and leads to effective qualitative navigation. Moreover, the acquired abstract knowledge base can be transferred to other scenarios. Our experiments indicate that indoor navigation can actually benefit from such a framework.

Several avenues are open to future work. Additional predicates can be tested to infer the robot location, and web resources can be mixed with human intervention. Furthermore, we intend to include measures of uncertainty about the situation of the robot, by associating probabilities with predicates. We also plan to conduct more extensive tests with the real robot.

Acknowledgments. The authors are partially supported by the National Council for Scientific and Technological Development (CNPq), Brazil.

References

1. Aydemir, A., Pronobis, A., Gobelbecker, M., Jensfelt, P.: Active visual object search in unknown environments using uncertain semantics. Robotics, IEEE Transactions on, 29(4), 986–1002 (2013)
2. Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., Patel-Schneider, P. F.: Description Logic Handbook, Cambridge University Press (2002)
3. Galindo, C., Saffiotti, A.: Inferring robot goals from violations of semantic knowledge. Robotics and Autonomous Systems, 61(10), 1131–1143 (2013)
4. Gupta, Rakesh, Kochenderfer, Mykel J. : Commonsense data acquisition for indoor mobile robots. In: 19th National Conference on Artificial Intelligence (AAAI-04), pp. 605–610. AAAI Press (2004)
5. Jain, D., Waldherr, S., and Beetz, M.: Bayesian Logic Networks. Technical report, IAS Group, Fakultat fur Informatik, Technische Universitat Munchen (2009)
6. Koga, M.L., Silva, V.F.d., Cozman, F.G., Costa, A.H.R.: Speeding-up reinforcement learning through abstraction and transfer learning. In: Conf. Autonomous Agents and Multiagent Systems, pp.119–126 (2013)
7. Lukasiewicz, T.: Expressive probabilistic description logics. Artificial Intelligence, 172(6-7), 852–883 (2008)
8. Lukasiewicz, T., Straccia, U.: Managing Uncertainty and Vagueness in Description Logics for the Semantic Web. Journal of Web Semantics, 6, 291–308 (2008)
9. Mateus, P., Pacheco, A., Pinto, J., Sernadas, A.:Probabilistic Situation Calculus. Annals of Mathematics and Artificial Intelligence, 32, 393–431 (2001)
10. Otterlo,M. V.: The Logic of Adaptative Behaviour. IOS Press, Amsterdam (2009)
11. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann (1988)
12. Riazuelo, L., Tenorth, M., Di Marco, D., Salas, M., Msenlechner, L., Kunze, L., Montiel, J. M. M.: RoboEarth Web-Enabled and Knowledge-Based Active Perception. In: IROS Workshop on AI-based Robotics (2013)
13. Saito, M., Chen, H., Okada, K., Inaba, M., Kunze, L.,Beetz, M.: Semantic object search in large-scale indoor environments. In: IROS Workshop on active Semantic Perception and Object Search in the Real World (2011)
14. Sutton, Richard S., Barto, Andrew G.: Introduction to Reinforcement Learning, MIT Press, Cambridge, MA (1998)
15. Tenorth, M., Klank, U., Pangercic, D., Beetz, M.: Web-enabled robots. Robotics & Automation Magazine, IEEE, 18(2), 58–68 (2011)
16. Tenorth, M., Beetz, M.: KnowRob – A Knowledge Processing Infrastructure for Cognition-enabled Robots. Part 1: The KnowRob System. International Journal of Robotics Research (IJRR) 32(5), 566–590 (2013)