Ontology-Based Unified Robot Knowledge for Service Robots in Indoor Environments

Gi Hyun Lim, *Student Member, IEEE*, Il Hong Suh, *Senior Member, IEEE*, and Hyowon Suh

***Abstract*—A significant obstacle for service robots is the execution of complex tasks in real environments. For example, it is not easy for service robots to find objects that are partially observable and are located at a place which is not identical but near the place where the robots saw them previously. To overcome the challenge effectively, robot knowledge represented as a semantic network can be extremely useful. This paper presents an ontology-based unified robot knowledge framework that integrates low-level data with high-level knowledge for robot intelligence. This framework consists of two sections: knowledge description and knowledge association. Knowledge description includes comprehensively integrated robot knowledge derived from low-level knowledge regarding perceptual features, part objects, metric maps, and primitive behaviors, as well as high-level knowledge about perceptual concepts, objects, semantic maps, tasks, and contexts. Knowledge association uses logical inference with both unidirectional and bidirectional rules. This characteristic enables reasoning to be performed even when only a partial information is available. The experimental results that demonstrate the advantages of using the proposed knowledge framework are also presented.**

***Index Terms*—Intelligent service robot, knowledge association, knowledge description, ontology-based knowledge, unified robot knowledge.**

I. Introduction

To PERFORM the service tasks effectively, service robots must handle not only the low-level sensory-motor data but also the high-level semantic information. These data and information are bidirectionally linked, with the low-level data passed upwards and the high-level information returned downwards using semantic relationships and hierarchy. In this paper, these data and information are defined as robot knowledge, and they must be mutually associated to be integrated into the unified robot knowledge for the service robots.

Manuscript received February 24, 2009; revised August 4, 2009; accepted October 28, 2009. Date of publication October 17, 2010; date of current version April 15, 2011. This work was supported in part by the Technology Innovation Program of the Korean Ministry of Knowledge and Economy/Korea Evaluation Institute of Industrial Technology under Grant 2008-F-038-01 (Development of Context Adaptive Cognition Technology) and in part by the Intelligent Robotics Development Program, which is one of the 21st Century Frontier R&D Programs funded by the Korean Ministry of Knowledge and Economy. This paper was recommended by Associate Editor K. Hirota.

G. H. Lim and I. H. Suh are with the Department of Electronics and Computer Engineering, Hanyang University, Seoul 133-791, Korea (e-mail: ihsuh@hanyang.ac.kr).

H. Suh is with the Department of Industrial Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-338, Korea.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCA.2010.2076404

A. Motivation

Robot knowledge is a key tool that enables the service robots to successfully complete the service tasks in real environments. This knowledge can be represented as a semantic network or graph, both of which are capable of representing semantic relations between concepts. Additionally, the service robots are designed to complete the service tasks semi- or fully automatically in a specific environment [1]. Robots must also interact with humans and must understand human intentions to enable social collaboration [2]. By using its sensors and actuators, the service robots must be able to perceive features, model environments with objects, and spaces and must carry out tasks using primitive robot behaviors and low-level sensor data. For example, the service robots must carry out high-level tasks such as "get me my cup." To complete this task, the service robot must know what the object is (e.g., cup) and where it is located. It must then move to the item's probable location and must find the object. In addition, the robots must combine tasks involving their primitive behaviors to accomplish movement or manipulation tasks such as "goto," "turn," and "grip," as well as perceptual tasks such as "extract features" and "match features" [3].

To compensate for an incomplete information, the service robots must also be able to use knowledge through relations represented by links between information. Currently, the use of the robot knowledge is gaining popularity in robotic fields such as mapping and localization, cognitive vision, and human–robot interaction in intelligent robots [4], [5]. For these reasons, different forms of robot knowledge must be fully integrated. The different forms of robot knowledge range from low-level robot data about perception and action to high-level information about the immediate environment from models that form the robots' internal representation of the real world.

B. Related Works

The Good Old Fashioned AI and Robotics system (GOFAIR) [6] is a classical symbol manipulation approach that is consist of three modules for perception, reasoning, and action. This system, however, is problematic when implemented in robots operating in real-world environments [7]. To address these problem areas in GOFAIR, behavior-based architectures were developed, in which perception and action are tightly coupled, allowing the robot to react to the perception of any stimulus. The most typical behavior-based robotic architecture is the subsumption architecture [8], in which higher layers of activity subsume lower layers in a behavioral hierarchy. Horizontal architectures have been used to demonstrate the complex tasks involving interactions of simple reactive behaviors [9]. However, the use of behavior-based architectures to complete complicated service tasks can frequently be problematic. In addition, replacing or changing the predesigned perception action pairs can be difficult. The integration of the robot knowledge from low to high levels could prove to be the best solution for robots to perform complicated service tasks in real environments.

Previous studies have investigated the use of the robot knowledge in specific fields in order to connect the existing low-level data with the high-level information between virtual and physical environments [10], and wireless sensor networks [11]. This can be classified into the following categories: object recognition and categorization [12], [13], context modeling [14] and context reasoning [15], [16], task planning [17], space representation and navigation [18]–[20], locomotion [21], and other specific applications such as scientific experimentation [22]. However, since a specific knowledge is integrated as a unified robot knowledge, this knowledge is likely to be utilized more frequently not only in its own field but also in other fields. For object recognition and categorization, a visual concept ontology that is composed of spatial concepts, spatial relations, color concepts, and texture concepts can all be used as an intermediate layer between domain knowledge and image processing procedures during the knowledge acquisition phase. Algorithms have also been developed for visual concept learning, feature selection, and training [12]. In addition, symbol grounding, which occurs between sensory features and their symbolic representations [13], requires robust object recognition methods [23], combining many local features with several other visual features such as shape, color, and texture in a probabilistic and/or ontological approach [12], [24]–[26].

For context modeling in ubiquitous systems and robot environments, devices, space, people, and artifacts are modeled using ontology, and rules are used as criteria in matching the contextual concept [14], [21], [22]. Furthermore, knowledge representation can be utilized in intricate task planning at both abstract and semantic levels, as well as for motion and manipulation planning at a geometric level [17]. To collaborate with humans, on the other hand, semantic representation of space is required [27]–[29]. Ontology can be used for spatial representation and navigation through a spatial semantic hierarchy (SSH) [18] and route graphs (RG) [19] representing the general concepts of space. SSH and RG can be deconstructed into navigation tasks by applying a formal ontology [20]. Compound action specifications that model the relationship between actions and conditions are inferred from knowledge of linguistic conditional phrases and from spatial actions and local configuration through route instructions [30]. A rule-based dialog system was proposed to extract the route instructions using an intermediate semantic representation [31]. Semantic relationships that classify the meanings of words in terms of spatial routes are script that is composed from a set of primitive operations on occupancy grids using a context-dependent natural language [32].

For surface locomotion, the ontology of robot description, terrain features, and locomotion capability can all be used to evaluate the ground robots [21]. An ontology-driven approach can also be utilized in specific applications such as scientific experimentation [22] to represent all of the relevant data and metadata. These approaches share a limited focus on specific tasks, but the service robots must carry out tasks that are more comprehensive. The service robots typically utilize the low-level sensory-motor data observed through the sensors when performing the commands and the high-level knowledge for reasoning and generation of action sequences [24]. These two different types of data (low-level sensory-motor data and high-level semantic information) must be integrated into a single framework to allow the service robots to meet the challenges of performing tasks in continuously changing environments.

Ontology [33] can be defined as "a formal and explicit specification of shared conceptualization" [34], [35], i.e., ontology is a description of the concepts and relationships for knowledge sharing across many domains. In this paper, robot knowledge is represented using ontology for the intelligent service robots to specify the knowledge that is shareable across all robotic fields.

The service robot is expected to perform the tasks of perception, planning, and action using an integration of the robot's sensory data and contextual understanding. However, previous approaches either have employed a partial information such as sensory data and context understanding or have developed specific individual tasks using both sets of data. Therefore, the proposed approach builds a unified robot knowledge framework from partial and specific knowledge such as object recognition and categorization, context modeling, task planning, space representation, and navigation. In this paper, robot knowledge is described within a concept hierarchy through the use of ontology. This ontology-based approach provides an appropriate framework for integrating knowledge with sensory data and context information. With this increased knowledge and extended relations, the robot is more likely to complete its tasks successfully. As a specific knowledge is integrated into the unified robot knowledge, knowledge from both the robot's own field and other fields is likely to be utilized more frequently.

C. Scope and Organization

This paper is organized as follows. Section II describes the approaches and architecture of the proposed robot knowledge framework. Section III presents the descriptions of the robot knowledge as knowledge classes. Section IV discusses the reasoning mechanisms as one type of knowledge association. Section V provides practical examples to demonstrate the validity of the proposed knowledge framework. Finally, Section VI presents some concluding remarks.

II. Service Robot Knowledge Framework

A. Approach

Neuroscientists have recently been investigating how the brain works, especially within the neocortex, which is a distinctive part of the brain found in mammals. As a result of their studies, they have discovered that an understanding of the principles of operation of the neocortex is necessary for robot intelligence. The main characteristic found in neocortex operation is the bidirectional information flow, where information in the neocortical columns is passed up and down a hierarchy for recognition and disambiguation and is propagated forward to predict the next information input [36]. By making the neocortex as an inspiration, this characteristic was used as a source for the proposed modeling robot knowledge framework. Here, individual robot knowledge is described within a hierarchy and is processed bidirectionally through their associations with each other.

Specifically, this approach builds up rich robot knowledge for service robots by unifying features, objects, spaces, contexts, and actions from specific applications such as object recognition and categorization, context modeling, task planning, space representation, and navigation. Moreover, action knowledge is coupled with all other concepts of model and feature knowledge to represent the world environments using features such as sensory-motor coordination [7], object action complex [37], and affordance, which rates the potentiality of objects and/or environments to allow robots to perform actions without planning [9]. This action-coupled knowledge could make it easier for robots to reactively select an action and not to deliberately explore all available knowledge. Since the unified knowledge contains bidirectional links between the low-level data and the high-level information, more opportunities are available to complete not only the specific applications but also the whole missions, even when only a partial information is available. In the case of object recognition failure due to partial occlusion, previous spatial relations between the object and its neighbors can be used to enable the robot to turn around to find related objects, which will increase the probability of finding the target object.

B. Robot Knowledge Framework

Previous studies have also developed robot knowledge in other areas: ontology-based robot knowledge [4], [25], context understanding [26], [38], abstraction of world elements and tasks [39], heuristics for the suggestion of alternative tasks [40], and semantic localization [41], [42]. These can be integrated into a unified knowledge framework that is used to describe robot knowledge from the low-level data to the high-level knowledge and to accommodate the aforementioned service robot functions.

This paper introduces an ontology-based unified robot knowledge (OUR-K) framework for service robots, which includes knowledge description and knowledge association, as shown in Fig. 1. Knowledge description defines the robot data and environment in five knowledge classes and includes comprehensively integrated robot knowledge ranging from the low-level knowledge of perception and action found by the robot itself to the high-level knowledge about the world model, including objects, spaces, and contexts.

Knowledge association, which creates and describes the relationships among knowledge descriptions, can make use of several methods such as logical inference, Bayesian inference, and heuristics to form relations. A logical inference method specifies or infers the relationships based on the ontological properties of the knowledge classes. A Bayesian inference method infers the unknown random variables for the knowledge classes with conditional independence. The heuristic algorithms can search for information about the knowledge classes with informed knowledge. With these features, an ontological system for robot knowledge description and knowledge association will allow the service robot knowledge to be managed comprehensively and synthetically, enabling the robots to recognize objects, to avoid obstacles, to understand contexts, and to process common concepts including facts, knowledge, and functions by employing their own sensors and behaviors. The service robot ontology will be based on the robot visual input, using the robot's perception of the environment through their own sensors to perform the required tasks.

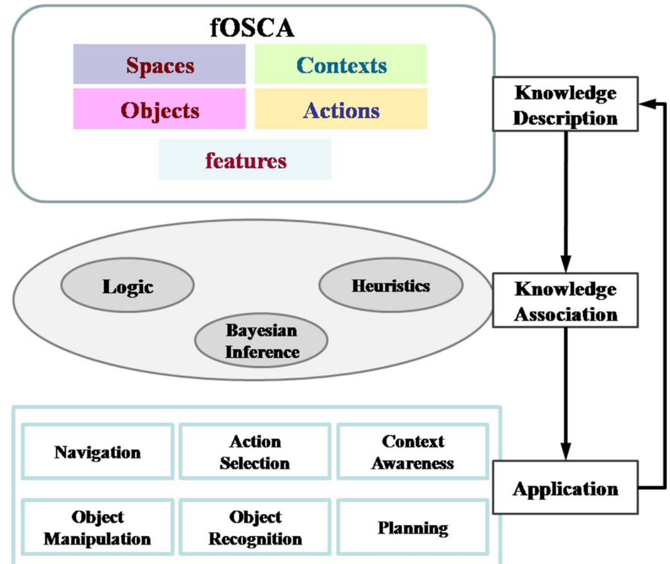


Fig. 1. OUR-K framework that is consist of three parts: knowledge description, knowledge association, and application.

OUR-K is a service robot knowledge framework that enables the robot to complete tasks even if it is given an incomplete information. The framework incorporates the knowledge description about the robot and its environment and the knowledge association algorithms, including logic, Bayesian inference, and heuristics. There are two types of robot knowledge: robot-dependent and robot-independent knowledge. A robot perceives and acts with its own sensors and motors but commonly models the world and plans a sequence of actions to achieve goals using perceived objects, spaces, and contexts. Contexts are abstract, and they share mutual relationships between objects and spaces. In particular, this framework categorizes the robot knowledge description into five classes to describe the world where the robot exists. Each knowledge class consists of two or three levels of knowledge, with each level containing three ontological layers. Low-level knowledge is associated with high-level knowledge through the intervention of mid-level knowledge, as shown in Fig. 2. A service robot perceives its environment, identifies features with its own sensors, models the environment internally with objects and spaces, plans a sequence of tasks, performs these tasks using its own primitive behaviors, and continuously repeats these processes [9]. Cognitive capabilities are essential for the completion of these processes. Context includes the characteristics of the particular

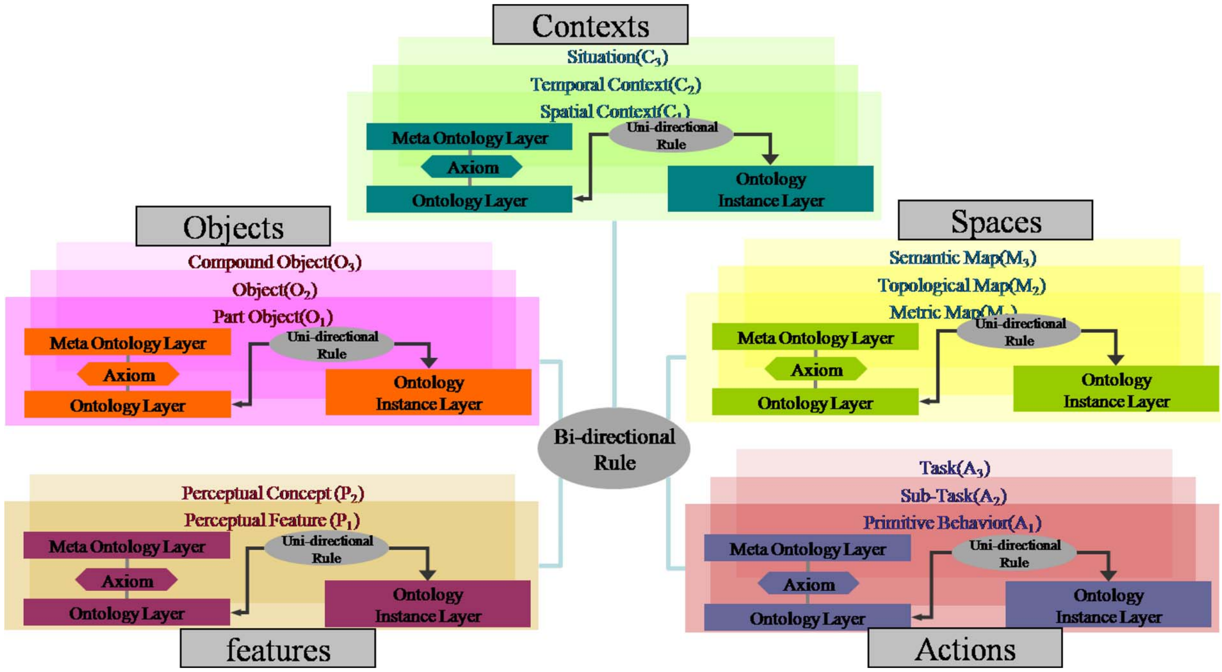


Fig. 2. Robot knowledge classes and association rules. OUR-K has five knowledge classes: feature, object, space, action, and context. Each knowledge class has three knowledge levels: high, middle, and low. Each knowledge level also has three ontology layers: metaontology, ontology, and ontology instance layers. Within these, there are axioms and rules. Concept hierarchies and axioms are represented with DL, while the unidirectional and bidirectional rules are represented with Horn logic.

environmental situation around the robot, which provides clues for the robot's appropriate action selection mechanism.

There have been many suggestions for formal representations, such as Web Ontology Language (OWL) and Resource Description Framework (RDF) [33]. The formal representations for OUR-K are defined in the Appendix. The framework's notation and semantics are based on the ontological structure found in the Karlsruhe Ontology (KAON) [43], in addition to previous research on ontology-based context understanding [25] and robot ontology [4], [38], [40].

III. Description of Knowledge Classes

Robot knowledge is designed for use in service robots to provide a well-defined robot software architecture and systematic and comprehensive integration of robot components. One of the objectives for this research is the improvement of a service robot for the elderly, called T-Rot, at the Center for Intelligent Service Robotics (CIR), which is managed by the Intelligent Robotics Development Program [44]. The hybrid architecture for this robot is a multilayered system that is comprised of a top deliberative layer and a lower reactive layer, which are joined through the interface or middle layer, linking rapid reaction and long-range deliberation [9], [45]–[47]. The role of this research lies in knowledge representation and reasoning for service robots.

World modeling is represented by objects and spaces where objects are located. Furthermore, contexts are used to help reduce the search space to complete a task. These are abstract representations of the spatial and temporal relationships between objects and spaces, and these data are integrated into unified robot knowledge using five knowledge classes: perception, object, space, action, and context. Data are not restricted to a particular knowledge class, and they can be applied to other knowledge classes. For example, an object can be recognized not only by its object properties but also by spatial contexts and/or its location in space.

Specifically, this tripodal schematic control architecture is used with a layered functionality diagram composed of three layers: deliberative, sequencing, and reactive layers [44]. The layered functionality diagram shows the connectivity and information flows between components, allowing the development of an efficient and well-designed control architecture. For this tripodal schematic concept, three knowledge levels for each knowledge class are specified: high-, mid-, and low-level knowledge. The main purpose of the high-level knowledge is to interface with the low-level knowledge in the upper knowledge classes, while the mid-level knowledge is used to sequence the low- and high-level knowledge residing in the same knowledge classes. The proposed robot knowledge is currently dedicated to T-Rot. Since the robot knowledge is represented by ontology, it is easy to share concepts and relations, allowing the robot knowledge to be easily extended to other service robots.

Ontology is used to represent the robot knowledge because it is shareable and exchangeable [36]. Existing knowledge concepts can be represented, as well as the properties and relations between concepts that provide an additional information to aid problem solving. Each knowledge level has three ontological layers (OLayer): a metaontology layer for generic knowledge, an ontology schema layer for domain knowledge, and an ontology instance layer to store the knowledge instance. Because ontology is an object-oriented and frame-based language that is using generic and individual frames, the metaontology and ontology layers can describe the conceptual relationships between knowledge classes with generic frames, while the ontology

TABLE I
OBJECT CLASS

Tuple	Examples
$OLayer_{211}$ (meta ontology layer) for $KLevel_{21}$ (low-level : part object) $OLayer_{211} := \langle Cp_{211}, Rel_{211}, H_{211}{}^{C}, H_{211}{}^{R}, A_{211}{}^{0} \rangle$	
Cp_{211}	{ObjectPart, PerceptualPart,FunctionalPart}
Rel_{211}	{hasPerceptualConcept, hasSIFTFeature, hasTextureFeature, hasFunction}
$H_{211}{}^{C}$	H^C(PerceptualPart :ObjectPart), H^C(FunctionalPart :ObjectPart)
$H_{211}{}^{R}$	H^R(hasSIFTFeature: hasPerceptualConcept), H^R(hasTextureFeature: hasPerceptualConcept)
$A_{211}{}^{0}$	$\forall x$, Part(x) $\wedge$ PerceptualConcept(y) $\wedge$ hasPerceptualConcept (x, y) $\rightarrow$ PerceptualPart (x) $\forall x$, Part(x) $\wedge$ Function (y)$\wedge$ hasFunction (x, y) $\rightarrow$ FunctionalPart (x)
$OLayer_{221}$ (meta ontology layer) for $KLevel_{22}$ (middle-level : object) $OLayer_{221} := \langle Cp_{221}, Rel_{221}, H_{221}{}^{C}, H_{221}{}^{R}, A_{221}{}^{0} \rangle$	
Cp_{221}	{Object, Container,Cup, Saucer}
Rel_{221}	{hasPosition, hasSize, hasPart, hasPerceptualPart, hasFunctionalPart}
$H_{221}{}^{C}$	H^C(Container :Object) , H^C(Cup :Container) , H^C(Saucer :Object)
$H_{221}{}^{R}$	H^R(hasPerceptualPart: hasPart), H^R(hasFunctionalPart: hasPart)
$A_{221}{}^{0}$	$\exists x$, Object(x) $\wedge$ PerceptualPart (y) $\wedge$ FunctionalPart (z) $\wedge$hasFunctionalPart (x, y) $\wedge$ hasPerceptualPart (x, z)
$OLayer_{231}$ (meta ontology layer) for $KLevel_{23}$ (high-level : compound object) $OLayer_{231} := \langle Cp_{231}, Rel_{231}, H_{231}{}^{C}, H_{231}{}^{R}, A_{231}{}^{0} \rangle$	
Cp_{231}	{CompoundObject, CupandSaucer}
Rel_{231}	{hasCompoundRelation, hasFunctionalRelation, hasLocationalRelation}
$H_{231}{}^{C}$	H^C(CupandSaucer : CompoundObject)
$H_{231}{}^{R}$	H^R(hasFunctionalRelation: hasCompoundRelation), H^R(hasLocationalRelation: hasCompoundRelation)
$A_{231}{}^{0}$	$\exists x$, CompoundObject(x) $\wedge$Object(y) $\wedge$ hasCompoundRelation (x,y)

Examples of classes begin with an uppercase, while those describing relations begin with a lowercase. A factor of class in parentheses is an instance of that class, while two factors of relation in parentheses are subject and object of relation respectively. "$\forall$" is a logical quantifier meaning "for all" and "$\exists$" is a logical quantifier meaning "exists."

All classes have a super-class as a concept hierarchy ($H_{ijk}{}^{C}$), with the ancestor of all classes being "Thing". All relations do not have an ancestor within the relation hierarchy ($H_{ijk}{}^{R}$).

instance layer can describe the real environments as individual frames. In particular, a metaontology can provide a template for an ontology schema layer focused on a building terminology, i.e., concept definitions. While building this ontology, the definition of any new concept is checked for consistency with the metaontology and other previously defined concepts through ontology reasoning. When in execution, ontology instances are propagated to reflect the real environment in the ontology instance layer, while coherence is checked according to the knowledge classes in the ontology layer.

A. Knowledge Class—Object

Table I lists the example object knowledge classes $KClass_2$, which has three knowledge levels: O_1, O_2, and O_3. O_1 is the part-object level which includes the functional and perceptual parts. An object can be divided into parts based on the function. For example, a `cup` is composed of a `body` and a `handle`, each with its own function, i.e., containing and grasping, respectively. In addition, an object can also be divided based on its perceptual features such as `color`, `texture`, and `scale invariant feature transform` (`SIFT`) features. The perceptual parts are clustered by their feature points, and the perceptual features are coupled within the perceptual concept level $(f_3 : KLevel_{13})$ in the feature knowledge class $(KClass_1)$. O_2 is the object level that includes the object name and functionality. O_3 is the compound level of the object class, where objects that are closely related (e.g., a `cup` and a `saucer`) are generally called for and utilized together.

To clarify the knowledge description of an object knowledge class $(KClass_2)$, a `cup` is used as an example. A `cup` is a kind of a `container` with perceptual parts, such as `color` and `SIFTS` features, and functional parts, such as a `body` and a `handle`. Therefore, the description logic (DL) representation of the object "`cup`" can be shown as

$$\begin{aligned} Cup := Object \\ &\wedge \exists hasSuperClass(Cup, Container) \\ &\wedge \exists hasPerceptualPart(Cup, Color) \\ &\wedge \exists hasPerceptualPart(Cup, SIFT) \\ &\wedge \exists hasFunctionalPart(Cup, Body) \\ &\wedge \exists hasFunctionalPart(Cup, Handle). \end{aligned}$$

These perceptual parts are grounded with the perceptual concepts $(f_3 : KLevel_{13})$ in the feature $KClass_1$. For robust symbol grounding [12], [23], [48] between the perceptual features $(f_1 : KLevel_{11})$ in the feature KClass and the perceptual parts $(O_1 : KLevel_{21})$ in the object KClass, general object recognition algorithms using many local features and/or color features are used in conjunction with the high-level knowledge, including the spatial information such as object and robot locations.

B. Knowledge Class—Space

The space knowledge class $(KClass_3)$ contains three knowledge levels: S_1, S_2, and S_3. Table II lists the examples of these space classes. The space KClass includes three types of maps: metric, topological, and semantic. A metric map (S_1) consists of empty and occupied areas built using the line and point features in a perceptual concept level $(f_3 : KLevel_{13})$ in the feature KClass. The topological map (S_2) includes `nodes` and `nodeEdges`, which are linked to the `objects` using `objectEdges` that are observed at each node on the semantic map (S_3). The following shows an example of an `Office`:

$$\begin{aligned} Office := \exists hasSuperClass(Office, Room) \\ &\wedge \exists hasNode(Office, Node) \\ &\wedge \exists hasObject(Office, Object). \end{aligned}$$

`Nodes` and `nodeEdges` are utilized to plan a sequence of activities to navigate within the given space.

C. Knowledge Class—Feature

The feature knowledge class $(KClass_1)$ has two KLevels: f_1 and f_3. Table III lists the examples of these feature classes. f_1 is the perceptual feature level that includes a set of perceptual features (SIFT) [49], the dominant color and texture for the object part $(O_1 : KLevel_{21})$ in $KClass_2$, and a metric map $(S_1 : KLevel_{31})$ in $KClass_3$. The perceptual features are produced by the robot's sensors and data processing algorithms. In addition, each perceptual feature has the property of the `hasFeatureAlgorithm`, including texture, color, and SIFT extractors. This property enables a vision module to activate the appropriate algorithms to identify the perceptual features. f_3 is the perceptual concept level that is grounded to f_1 by a

TABLE II
SPACE CLASS

Tuple	Examples
OLayer$_{311}$ (meta ontology layer) for KLevel$_{31}$ (low-level : metric map) $OLayer_{311} := \langle Cp_{311}, Rel_{311}, H_{311}^C, H_{311}^R, A_{311}^0 \rangle$	
Cp_{311}	*{MetricMap, Area, EmptyArea, OccupiedArea}*
Rel_{311}	*{hasMetricMap, hasArea, hasEmptyArea, hasOccupiedArea}*
H_{311}^C	*H^C(EmptyArea: Area), H^C(OccupiedArea: Area)*
H_{311}^R	*H^R(hasEmptyArea: hasArea), H^R(hasOccupiedArea: hasArea)*
A_{311}^0	
OLayer$_{321}$ (meta ontology layer) for KLevel$_{32}$ (middle-level : topological map) $OLayer_{321} := \langle Cp_{321}, Rel_{321}, H_{321}^C, H_{321}^R, A_{321}^0 \rangle$	
Cp_{321}	*{TopologicalMap, Node, Edge, NodeEdge, ObjectEdge}*
Rel_{321}	*{hasTopologicalMap, hasNode, hasEdge, hasNodeEdge, hasObjectEdge}*
H_{321}^C	*H^C(NodeEdge: Edge), H^C(ObjectEdge: Edge)*
H_{321}^R	*H^R(hasNodeEdge: hasEdge), H^R(hasObjectEdge: hasEdge)*
A_{321}^0	$\exists x$, Node(x) $\wedge$ Edge(y) $\wedge$ hasEdge(x,y) $\forall x$, Edge (x) $\wedge$ Node (y) $\wedge$ Node (z) $\wedge$ hasEdge (y, x) $\wedge$ hasEdge (z, x) $\rightarrow$ NodeEdge (x) $\forall x$, Edge (x) $\wedge$ Node (y) $\wedge$ Object (z) $\wedge$ hasEdge (y, x) $\wedge$ hasEdge (z, x) $\rightarrow$ ObjectEdge (x)
OLayer$_{331}$ (meta ontology layer) for KLevel$_{33}$ (high-level : semantic map) $OLayer_{331} := \langle Cp_{331}, Rel_{331}, H_{331}^C, H_{331}^R, A_{331}^0 \rangle$	
Cp_{331}	*{SemanticMap, Building, Room, Office, Kitchen}*
Rel_{331}	*{hasSemanticMap, hasObject}*
H_{331}^C	*H^C(Building: SemanticMap), H^C(Room: Building), H^C(Office: Room), H^C(Kitchen: Room)*
H_{331}^R	
A_{331}^0	$\exists x$, SemanticMap(x) $\wedge$ Node(y) $\wedge$ hasNode(x,y)

TABLE III
FEATURE CLASS

Tuple	Examples
OLayer$_{111}$ (meta ontology layer) for KLevel$_{11}$ (low-level : perceptual feature) $OLayer_{111} := \langle Cp_{111}, Rel_{111}, H_{111}^C, H_{111}^R, A_{111}^0 \rangle$	
Cp_{111}	*{PerceptualFeature, SIFTFeature, TextureFeature, ColorFeature, EdgeFeature}*
Rel_{111}	*{hasPerceptualFeature, hasTextureFeature, hasSIFTFeature, hasColorFeature, hasEdgeFeature, hasFeatureAlgorithm}*
H_{111}^C	*H^C(SIFTFeature: PerceptualFeature), H^C(TextureFeature: PerceptualFeature), H^C(ColorFeature: PerceptualFeature), H^C(EdgeFeature: PerceptualFeature)*
H_{111}^R	*H^R(hasTextureFeature: hasPerceptualFeature), H^R(hasSIFTFeature: hasPerceptualFeature), H^R(hasColorFeature: hasPerceptualFeature), H^R(EdgeFeature: hasPerceptualFeature)*
A_{111}^0	$\exists x$, PerceptualFeature (x) $\wedge$ Algorithm (y) $\wedge$ hasAlgorithm(x, y)
OLayer$_{131}$ (meta ontology layer) for KLevel$_{13}$ (high-level : perceptual concept) $OLayer_{331} := \langle Cp_{131}, Rel_{131}, H_{131}^C, H_{131}^R, A_{131}^0 \rangle$	
Cp_{131}	*{PerceptualConcept, Color, Texture, SIFT, Point, Line}*
Rel_{131}	*{hasPerceptualConcept, hasColor, hasTexture, hasPoint, hasLine, hasMatchAlgorithm }*
H_{131}^C	*H^C(Color: PerceptualConcept), H^C(Texture: PerceptualConcept), H^C(SIFT: PerceptualConcept), H^C(Point: PerceptualConcept), H^C(Line: PerceptualConcept)*
H_{131}^R	*H^R(hasColor: hasPerceptualConcept), H^R(hasTexture: hasPerceptualConcept), H^R(hasPoint: hasPerceptualConcept), ,H^R(hasLine: hasPerceptualConcept)*
A_{131}^0	$\exists x$, PerceptualConcept (x) $\wedge$ PerceptualFeature (y) $\wedge$ hasPerceptualFeature (x,y)

matching algorithm, negating the need for an intermediate level (f_2 : KLevel$_{12}$) to connect the perceptual feature and perceptual concept levels. The following is an example of a color feature `GreenishBlue`:

$$\begin{aligned} &GreenishBlue: \\ &\quad = \exists hasSuperClass(GreenishBlue, ColorDescriptor) \\ &\qquad \wedge \exists hasAlgorithm(GreenishBlue, extractColor) \\ &\qquad \wedge \exists hasColor(GreenishBlue, ColorValue). \end{aligned}$$

TABLE IV
CONTEXT CLASS

Tuple	Examples
OLayer$_{411}$ (meta ontology layer) for KLevel$_{41}$ (low-level : spatial context) $OLayer_{411} := \langle Cp_{411}, Rel_{411}, H_{411}^C, H_{411}^R, A_{411}^0 \rangle$	
Cp_{411}	*{SpatialContext, On,In, Left, Right}*
Rel_{411}	*{hasSpatialContext, hasSubjective, hasObjective}*
H_{411}^C	*H^C(On: SpatialContext), H^C(In: SpatialContext), H^C(Left: SpatialContext), H^C(Right: SpatialContext)*
H_{411}^R	*H^R(hasSubjective: hasSpatialContext), H^R(hasObjective: hasSpatialContext)*
A_{411}^0	$\exists x$, SpatialContext (x) $\wedge$ Object (y) $\wedge$ hasSpatialContext (y,x)
OLayer$_{421}$ (meta ontology layer) for KLevel$_{42}$ (middle-level : temporal context) $OLayer_{421} := \langle Cp_{421}, Rel_{421}, H_{421}^C, H_{421}^R, A_{421}^0 \rangle$	
Cp_{421}	*{TemporalContext, Interval, Before, After, Met-by, Overlap, Meet}*
Rel_{421}	*{hasTemporalContext, hasSubjective, hasObjective}*
H_{421}^C	*H^C(Interval: TemporalContext), H^C(Before: TemporalContext), H^C(After: TemporalContext), H^C(Met-by: TemporalContext), H^C(Overlap: TemporalContext),H^C(Meet: TemporalContext)*
H_{421}^R	*H^R(hasSubjective: hasTemporalContext), H^R(hasObjective: hasTemporalContext)*
A_{421}^0	$\forall x$, Object (x) $\wedge$ SpatialContext(y) $\wedge$ SpatialContext(z) $\wedge$ hasSpatialContext (x,y) $\wedge$ hasSpatialContext (x,z) $\wedge \neg$equal (y,z) $\rightarrow$ TemporalContext (c) $\wedge$ hasTemporalContext (x,c)
OLayer$_{431}$ (meta ontology layer) for KLevel$_{43}$ (high-level : situation) $OLayer_{431} := \langle Cp_{431}, Rel_{431}, H_{431}^C, H_{431}^R, A_{431}^0 \rangle$	
Cp_{431}	*{Situation, Crowd}*
Rel_{431}	*{hasSituation, hasSubjective, hasObjective}*
H_{431}^C	*H^C(Crowd: Situation)*
H_{431}^R	*H^R(hasSubjective: hasSituation), H^R(hasObjective: hasSituation)*
A_{431}^0	$\exists x$, Situation (x) $\wedge$ TemporalContext (y) $\wedge$ hasTemporalContext (x,y)

When an object recognition is requested, the acquired image is segmented roughly according to points of inflection in the edge histogram. For each block of segments, the values of the numerical descriptors and visual features are obtained using the recommended visual feature extractors. For example, the visual concepts of color and SIFT may be recommended when processing a segmented block, yielding the corresponding properties of `GreenishBlue` and a number of matched SIFT descriptors.

D. Knowledge Class—Context

The context knowledge class (KClass_4) has three KLevels: C_1, C_2, and C_3. Table IV lists the examples of these context classes. C_1 is the spatial context level that contains the spatial relationships such as `on`, `in`, `left`, and `right` [50]. The spatial relationships are inferred based on the instances and values of properties contained in the object and space levels in KClass_2. C_2 is the temporal context level based on the temporal concepts defined by Allen [51]. Temporal contexts, including `before`, `after`, `met` $-$ `by`, `overlap`, and `meet`, can be inferred using the instances of model and spatial context classes. C_3 is the situation level (e.g., `crowd`, which means that the robot will encounter some obstacles), which can be useful for the development of navigation stratagems. The context does not only contain data such as a list of objects or their locations, but it also includes abstract and characteristic situations that can be represented by relationships between objects and object properties. Low-level spatial (C_1) and temporal contexts (C_2) are populated using generic reasoning. C_1 and C_2 are inferred through the objects and their locations, and the time intervals of the instantiation of the spatial contexts, respectively. However,

TABLE V
ACTION CLASS

Tuple	Examples
OLayer_{511} (meta ontology layer) for KLevel_{51} (low-level : primitive behavior) $OLayer_{511} := \langle Cp_{511}, Rel_{511}, H_{511}{}^{C}, H_{511}{}^{R}, A_{511}{}^{0} \rangle$	
Cp_{511}	*{PrimitiveBehavior, PerceptualBehavior, MotionBehavior, Manipulation-Behavior, Turn, Goto, ExtractColor, ExtractSIFT, ExtractLine}*
Rel_{511}	*{hasBehavior, hasBehaviorTarget, hasParameter}*
$H_{511}{}^{C}$	H^C*(PerceptualBehavior: PrimitiveBehavior),* H^C*(MotionBehavior: Primitive-Behavior),* H^C*(ManipulatingBehavior: PrimitiveBehavior),* H^C*(Turn: MotionBehavior),* H^C*(Goto: MotionBehavior),* H^C*(ExtractColor: PerceptualBehavior),* H^C*(ExtractSIFT: PerceptualBehavior),* H^C*(ExtractLine: PerceptualBehavior)*
$H_{511}{}^{R}$	
$A_{511}{}^{0}$	$\exists x$, *PrimitiveBehavior (x)* $\wedge$ *Object (y)* $\wedge$ *hasBehaviorTarget(x, y)* $\exists x$, *PrimitiveBehavior (x)* $\wedge$ *value (y)* $\wedge$ *hasParameter(x, y)*
OLayer_{521} (meta ontology layer) for KLevel_{52} (middle-level : sub-task) $OLayer_{521} := \langle Cp_{521}, Rel_{521}, H_{521}{}^{C}, H_{521}{}^{R}, A_{521}{}^{0} \rangle$	
Cp_{521}	*{SubTask, PerceptualSubTask, MotionSubTask, ManipulationSubTask, RecognizeObject}*
Rel_{521}	*{hasSubTask, hasSubTaskTarget, hasSubTaskParameter, hasSequence}*
$H_{521}{}^{C}$	H^C*(PerceptualSubTask: SubTask),* H^C*(MotionSubTask: SubTask),* H^C*(ManipulationSubTask: SubTask),* H^C*(RecognizeObject: PerceptualSubTask)*
$H_{521}{}^{R}$	H^R*(hasBehavior: hasSubTask)*
$A_{521}{}^{0}$	$\exists x$, *Subtask (x)* $\wedge$ *Behavior(y)* $\wedge$ *hasBehavior (x,y)* $\wedge$ *hasSequence(x, z)*
OLayer_{531} (meta ontology layer) for KLevel_{53} (high-level : task) $OLayer_{531} := \langle Cp_{531}, Rel_{531}, H_{531}{}^{C}, H_{531}{}^{R}, A_{531}{}^{0} \rangle$	
Cp_{531}	*{Task, Delivery, Navigation, FindObject, GenerateContext}*
Rel_{531}	*{hasTask, hasTaskTarget, hasTaskParameter}*
$H_{531}{}^{C}$	H^C*(Delivery: Task),* H^C*(Navigation: Task),* H^C*(FindObject: Task),* H^C*(GenerateContext: Task)*
$H_{531}{}^{R}$	H^R*(hasSubTask: hasTask)*
$A_{531}{}^{0}$	$\exists x$, *Task (x)* $\wedge$ *Subtask (y)* $\wedge$ *hasSubTask (x,y)*

the high-level contexts (C_3) are populated with domain-specific inference rules. The situation context "crowd" is extracted and used in the navigation domain. The following example shows the spatial relation on:

$$On := \exists hasSuperClass(On, RelativeSpatialContext)$$
$$\wedge\ \forall hasSubjective(On, Object)$$
$$\wedge\ \forall hasObjective(On, Object).$$

Spatial contexts such as on and left are instantiated from the position of the objects, specified from the object's hasPosition property in the model KClass. These spatial contexts have two properties: hasSubjective and hasObjective. The range of these two properties is found within the object level in the model KClass. For example, the spatial context on has a property hasSubjective with the value cup and a property hasObjective with the value table. The high-level context of crowd can then be inferred from the spatial and/or temporal contexts when a given object is too big to pass without impediment.

E. Knowledge Class—Action

The action knowledge class (KClass_5) also contains three knowledge levels: A_1, A_2, and A_3. Table V lists the examples of these action classes. A_1 is the primitive behavior level, including the robot's atomic functions. These include motion for metric preconditions and effects such as goto and turn, as well as feature extraction algorithms for perception such as extractColor, extractSIFT, and extractLine. A_3 is the task level describing the long-term goals for the symbolic preconditions and effects including delivery, navigation, findObject, and generateContext. A_2 is the subtask level with functions such as gotoSpace, localization, and recognizeObject. These can be defined as a short-term sequence of behaviors required to complete the subgoals, which are determined by clustering similar features of preconditions and effects [52]. A task can be decomposed into subtasks, which can be simultaneously decomposed into primitive behaviors. For example, RecognizeObject is the task of finding an object within an image. The following is an example of the delivery task:

$$Delivery$$
$$:= \exists hasSuperClass(Delivery, Task)$$
$$\wedge \forall hasTaskTarget(Delivery, Object)$$
$$\wedge \exists hasSubTask(Delivery, PerceptualSubTask)$$
$$\wedge \forall hasSubTask(Delivery, MovingSubTask).$$

Each ontology schema layer within the action class is instantiated by a planner. When requested to plan, it first gathers all the ontological instances in the topological map level $(\mathrm{S}_2 : \mathrm{KLevel}_{32})$, in the semantic map level $(\mathrm{S}_3 : \mathrm{KLevel}_{33})$, and in the instances containing spatially relevant objects $(\mathrm{O}_2 : \mathrm{KLevel}_{22})$ and compound objects $(\mathrm{O}_3 : \mathrm{KLevel}_{23})$, which are linked through the ObjectEdge parameter. Following this, the planner identifies the instances in the task knowledge level (A_3) and their sequences, which can also be found as values within the property hasSequence located in the subtask level (A_2). Finally, the planner generates the instances in the primitive behavior level (A_1) and produces the sequences for each task instance. In this paper, an abductive event calculus planner was used for this planning and instantiation process [53].

For the subtask recognizeObject, the primitive behaviors extractSIFT and extractColor can be performed. This subtask comprises one part of the delivery task whose target object (hasTaskTarget) is cup.

IV. KNOWLEDGE ASSOCIATION

Logical inference, Bayesian inference, and heuristics can be used to apply knowledge association among different OLayers, KLevels, and/or KClasses. Logical inference specifies the relations with logical rigidity, while the use of Bayesian inference with conditional independence resolves uncertainties, and the heuristic algorithms incorporating knowledge instances can be helpful in problem solving. For the purposes of this paper, logical inference and formal representation are the only methods used. Logical inference enables the robots to use the expressive inference mechanisms that identify hidden knowledge using well-defined rules [54], [55]. These rules represent the relationships between knowledge levels and/or ontological layers. In addition, it is necessary to define the axioms that specify the semantics of the concepts and relational constraints at each OLayer—these are generally accepted as valid without proof. Rules are utilized using logical inference, which is one type of knowledge association, as shown in Fig. 1, by inferring facts using ontological schema and ontology instances occurring at KClasses and KLevels. The general form taken by these rules is

"IF–THEN" using the Horn clause, with the rules for reasoning considered as functions with the ability to transform a set of facts into new facts. Rules are defined as relationships between different ontological layers, knowledge levels, or knowledge classes, whereas the simple relationships between different KClasses are represented by the ontological properties.

A. Axioms and Two Types of Rules

In the proposed knowledge framework, the axioms are defined using DL in the ontology schema layer ($\mathrm{OLayer_{ij1}}$ and $\mathrm{OLayer_{ij2}}$) and are used to evaluate whether the generated instances ($\mathrm{OLayer_{ij3}}$) are consistent. Moreover, rules are semantically categorized into two types: the $\mathrm{R^U}$ rules govern the unidirectional reasoning within each knowledge class ($\mathrm{KClass_i}$), while the $\mathrm{R^B}$ rules control the bidirectional reasoning between different knowledge classes. Unidirectional reasoning can be inferred using ontological classes, properties, and instances within the same knowledge level or between knowledge levels ($\mathrm{KLevel_{ij}}$). Where perceptions and actions exist, the axioms are applied to verify whether these data could be knowledge instances that accord with the ontology schema or ontology instances that have been previously asserted. If the data are new, the knowledge instances can then be created. Otherwise, if the data correspond with the knowledge instances with changed properties, the knowledge instances are updated, but if there are no perceptions and actions matched with a priori knowledge instances, those instances are deleted. Any changes in the knowledge instances trigger the application of unidirectional rules using forward chaining. In the proposed rule inference, the novel data drive this forward chaining inference, allowing more data to be extracted in the knowledge classes. On the other hand, unidirectional rule inference is restricted within the same knowledge classes. When a query is presented, a list of goals is generated, and the system works in reverse using backward chaining inference. This begins at the consequent, denoted by the "THEN" part of a bidirectional rule, and works to the antecedent that corresponds to the "IF" part of the rule to find the available data to support any of these consequents. If the goal cannot be matched, it is divided into subgoals, and the system recursively searches. The bidirectional rules are not restricted within a single knowledge class, solely returning the results of the query.

For example, the dominant color of a perceptual concept layer in the feature knowledge class is identified using the dissimilarity between its color descriptor and the observed color descriptor. Adjacent knowledge levels are associated with each other through unidirectional reasoning. In contrast, bidirectional reasoning is inferred across several knowledge classes ($\mathrm{KClass_i}$) or knowledge levels ($\mathrm{KLevel_{ij}}$). When object recognition fails after being successful, the bidirectional reasoning rules utilize not only the perceptual feature ($\mathrm{KLevel_{11}}$) and perceptual concept ($\mathrm{KLevel_{13}}$) levels in the feature knowledge class ($\mathrm{KClass_1}$) but also the following: 1) object ($\mathrm{KLevel_{22}}$) and compound object ($\mathrm{KLevel_{23}}$) levels in the object knowledge class; 2) topological map level ($\mathrm{KLevel_{32}}$) in the space knowledge class; and 3) spatial context ($\mathrm{KLevel_{41}}$) and temporal context ($\mathrm{KLevel_{42}}$) levels in the context knowledge class ($\mathrm{KClass_4}$). Through bidirectional reasoning, the highly sophisticated knowledge is represented across several knowledge classes and levels. In this way, the reasoning of the comprehensive tasks can be achieved, even with partial data that may arise from incomplete robot sensors or occluded scenes.

TABLE VI
EXAMPLES OF RULES

Class Relation	Meaning	First Order Logic (FOL) Representation
Matching of Dominant Color		
Uni-directional Feature: PerceptualConcept (f_3) PerceptualFeature (f_1)	IF Dissimilarity between D1of Color and D2 is less than Threshold THEN D2 is Color	∀c,d1,d2 Color (c) ∧ ColorDescriptor (d1) ∧ ColorDescriptor (d2) ∧ valueOfDissimilarity (d1, d2) < c.colorThreshold ⇒ Color (c, d2)
Monitoring for Navigation		
Bi-directional Space: TopologicalMap (S_3) Object: Object (O_2) Action: Task (A_3)	IF Robot has to go Target_Node AND A is object AND A is linked to Target_Node AND A is recognized THEN Navigation successes	∀o,n,e Object (o) ∧ Node (n) ∧ Navigation (n) ∧ hasObjectEdge (n,o) ∧ RecognizeObject (o) ⇒ Navigation (n)
Missed Object Recognition		
Bi-directional Object: Object (O_2) Context SpatialContext (C_1) Action Sub-Task (A_2)	IF Object Recognition fails AND A is instance of context with Target AND A is recognized THEN Approach A AND recognize Object Target	∀x,h,k,m,o Object (x) ∧ Object (y) ∧ ¬recognizeObject (x) ∧ hasSpatialContext (x, y) ∧ RecognizeObject (y) ⇒ Approach (y) ∧ RecognizeObject (x)

B. Role and Interaction Between Unidirectional and Bidirectional Reasonings

By allowing two types of rules to be applied in sequence, the search space for the inference can be reduced. When a query is presented at a knowledge class, the unidirectional rules within that knowledge class are applied first to infer the correct answers using facts confined within the knowledge class. If these unidirectional rules fail to answer the query, the bidirectional rules are then applied across the different knowledge classes to find the answer. In addition, the instances generated by the original unidirectional rules can be simultaneously employed within the bidirectional rules. For example, the object location in the model class can be used not only as an *a priori* knowledge for object recognition but also as a fact for space classification. Moreover, the contextual information inferred in the model class can simultaneously be used for object recognition in the perception class. Table VI lists some examples of the rules for navigation monitoring and missing object recognition tasks using several knowledge classes.

As far as monotonicity is concerned, if new facts or rules are expected not to change in the knowledge base, the system is monotonic. Otherwise, if the axioms or rules are allowed to add or delete facts in the knowledge base, the behavior of the system is nonmonotonic [56], [57]. In OUR-K, the consequence of reasoning using bidirectional rules is monotonic since new facts are not created and since only the queries are answered. However, the unidirectional rules for spatial reasoning are nonmonotonic operators, which reflect the changes in object location since the new spatial relationships are inferred between newly observed objects, necessitating the update of the previous spatial relationships. It is noted that, as in previous researches [56], [57], spatial reasoning, if produced by the update and delete rules, can be interpreted within a closed world assumption (CWA), whereby anything not currently known to be true is assumed false. The following update and delete rules for spatial reasoning are designed under CWA to make the knowledge consistent.

When a new object instance A1 is observed at node N1, an object edge marking the domain object as observable at the node is created:

If A1 *is not linked with* N1
And A1 *is observed at* N1
Then create object edge *between* A1 *and* N1.

When an object instance A1 is observed at node N1, where A1 was previously observed at another node N2, the object edge linking A1 and N2 is removed:

If A1 *was previously linked with* N1
And A1 *is observed at* N2
And N1 *is not equal to* N2
Then delete object edge *between* A1 *and* N1.

When an object instance A1 is observed at node N1 and when A1 co-occurs with another object instance A2, a new spatial relation is created:

If A1 *was not linked with* N1
And A1 *is observed at* N1
And A2 *is linked with* N1
And A1 *and* A2 *co-occur*
And there is no instance of spatial context
between A1 *and* A2
Then create spatial context between A1 *and* A2.

Once created, a spatial context such as on, in, left, and right, which are child classes of the spatial context, is specified based on the node view point.

When an object instance A1 is observed at node N1 and when A1 was previously observed at another node N2, the spatial contexts of object A2 at node 2 are removed:

If A1 *was previously linked with* N1
And A1 *is observed at* N2
And A2 *is linked with* N1
And N1 *is not equal to* N2
And there are instances
of spatial context of A1 *at node* N1
Then delete spatial context of A1 *at Node* N1.

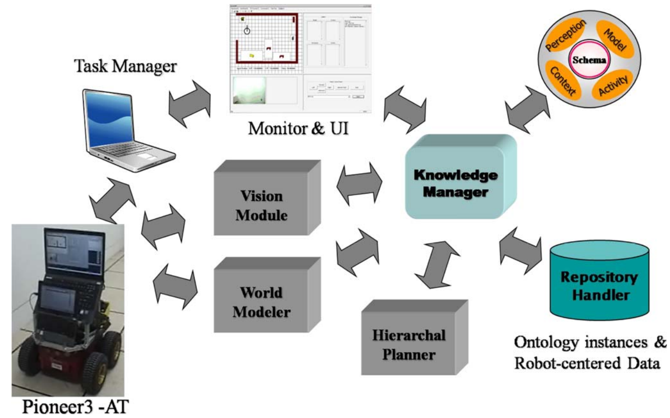


Fig. 3. System configuration of the experiments.

Finally, when an object instance A1 is observed at node N1 where A1 was previously observed, the lower classes of the spatial relation for A1 are flagged for change and are then updated.

As the objects are observed and as the robot is localized, the world model reflects the real environments by using the spatial contexts and the semantic map as the consequences of the update and delete rules.

V. Experiment

A. Implementation System Architecture

The proposed framework for robot knowledge can be verified with the experimental task of a reception service using a cup, which is one of the challenges laid out by CIR. Fig. 3 shows the configuration of this experimental system. The architecture of the CIR robot is hybrid as it integrates reactive control with deliberation [44], [46]. A task manager (TM) and a knowledge manager (KM) are deployed at the deliberative layer to perform computations such as planning, world modeling, and high-level reasoning. Task management is implemented as a belief, desire, and intention agent [47] with goals, plans, and action selection, and KM is set as a service agent with a world model, contextual information, and knowledge base. They interact using predefined goal-based interaction. KM handles requests about ontology, including ontology creation, retrieval, and update/deletion. It then makes appropriate queries to the robot-centered ontology schema and ontology repository handler. A pioneer robot was used in our experiments, as shown in Fig. 3, and since this robot lacked arms, the delivery task was replaced with "find object" and "generate context about that object" tasks. Robot ontology and metalevel ontology schemas were designed using OWL (with the Protégé ontology editor), and the ontology instances were accumulated in the repository handler. The hierarchical planner generated plans and ontology instances for each layer within the action KClass_5 (A). The World Modeler was used to build the metric, topological, and semantic maps and to generate the space ontology instances in the space KClass_3, as well as to locate the robot on the maps. A camera was fixed on top of the robot for object recognition by the vision module. For this experiment, the Evolution Robotics Software Platform [58]

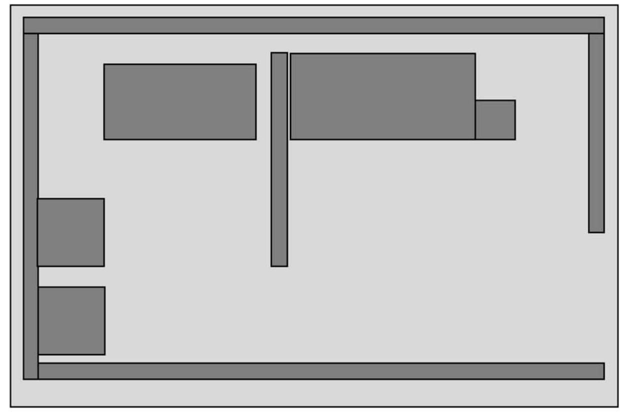
Fig. 4. Experimental environments.

was used for object recognition and localization. Photographs taken from the camera at any given moment are processed using perception algorithms in the vision module, generating ontology instances within the feature level $(\mathrm{f} : \mathrm{KClass}_1)$ after perceptual processing. The perceived features (OLayer_{113}) are then grounded with the ontological instances of the perceptual concepts (OLayer_{133}). The perceptual concepts become the ontological instances of either object features (OLayer_{213}) and objects (OLayer_{223}) in the object class (KClass_2) or metric map (OLayer_{313}) and topological map (OLayer_{323}) in the space class (KClass_3). The ontological instances of the spatial contexts (OLayer_{413}) and temporal contexts (OLayer_{423}) in the context class (KClass_4) are inferred in the KM.

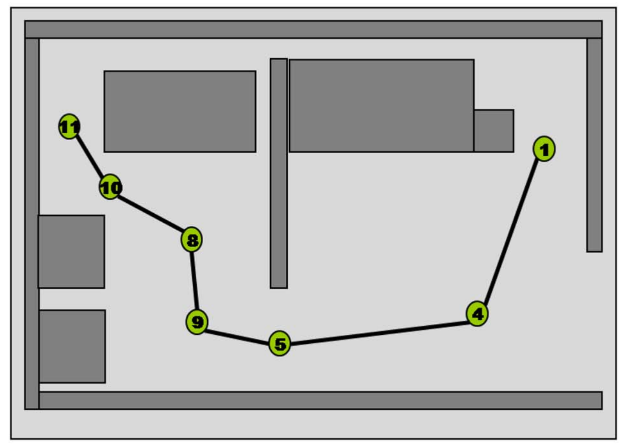
B. Experimental Environments and Ontology Instantiation of the Space KClass

The photographs in Fig. 4 show the experimental environments used for the OUR-K knowledge description (KDescription). One room is an `office`, which is an instance (OLayer_{333}) in the semantic map level (KLevel_{33}) and which includes three topological nodes: `nodes 1`, `4`, and `5` at the topological map level (KLevel_{32}). The other room is a `kitchen`, and it includes five nodes. `Node 5` is included in both rooms, making it a connecting node between the `kitchen` and the `office`.

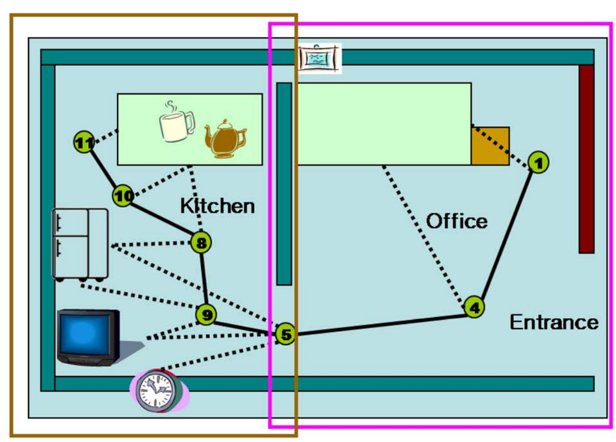
Fig. 5 shows the metric map (KLevel_{31}), the topological map (KLevel_{32}), and the semantic map (KLevel_{33}), which are included in the space knowledge class (KClass_3). Fig. 6 shows the ontology schema (OLayer_{312}) and the ontology instances (OLayer_{313}) for the metric map level (KLevel_{31}) based on the results found using a laser range finder for the experimental space shown in Fig. 5(a). It includes two types of area: occupied and empty. Fig. 7 shows the ontology schema (OLayer_{322}) and the ontology instance (OLayer_{323}) for the topological map level (KLevel_{32}) shown in Fig. 5(b). It includes `nodes` and `nodeEdges` generated using a Voronoi diagram. These can also be added upon the user request to be used by another module. The presence of `nodeEdges` between the `nodes` indicates that both `nodes` are linked and are navigated by the robot. Fig. 8 shows the ontology schema (OLayer_{332}) and the ontology instance (OLayer_{333}) layers within the semantic map level (KLevel_{33}) for the experimental space shown in Fig. 5(c). This includes `objects` and `objectEdges`. If a certain `object` is recognized as not movable, that object is added as a landmark, and its `objectEdge` links it to the `nodes` from where it is observable. In the case of movable objects, the spatial contexts $(\mathrm{C}_1 : \mathrm{KLevel}_{41})$ are first generated with immovable objects,



(a)



(b)



(c)

Fig. 5. Three types of map for the experimental environments. (a) Metric map. (b) Topological map. (c) Semantic map.

followed by other movable objects. In addition, all of these objects are located in the occupied area of the metric map $(\mathrm{S}_1 : \mathrm{KLevel}_{31})$. As a result, the three maps are cross related, and updating one map causes all other maps to also update with the relevant properties.

C. Experimental Results

Fig. 9 shows an experimental scenario that is designed to clarify how a service robot can use the symbolic knowledge to solve a specific task using only a partial information, where the

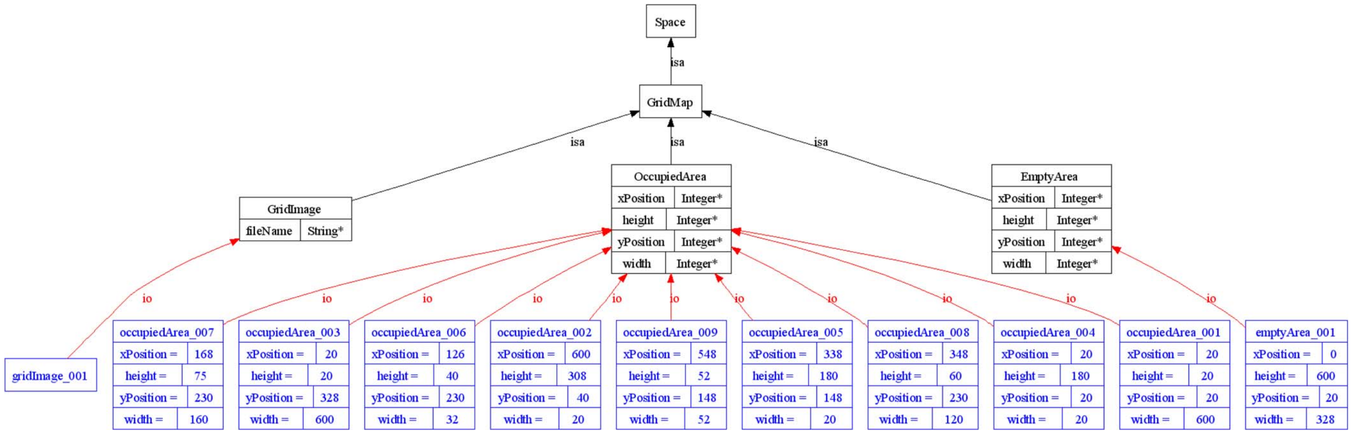


Fig. 6. Ontology schema and the metric map instance (KLevel_{31}).

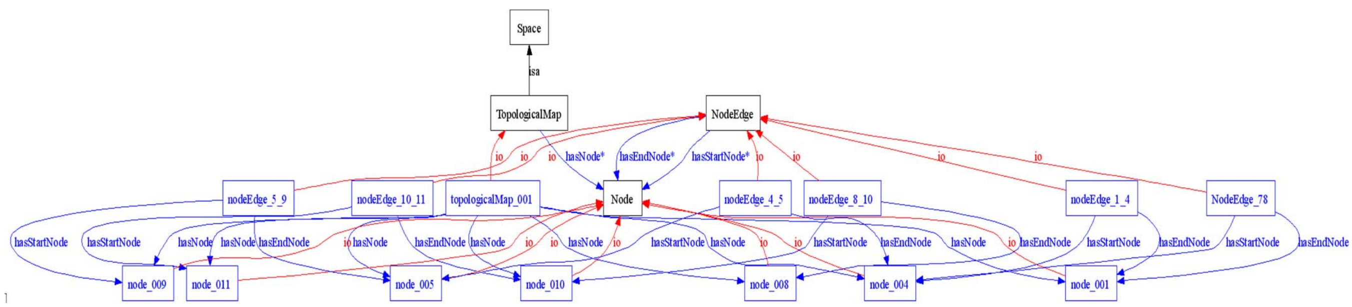


Fig. 7. Ontology schema and the instance of the topological map (KLevel_{32}).

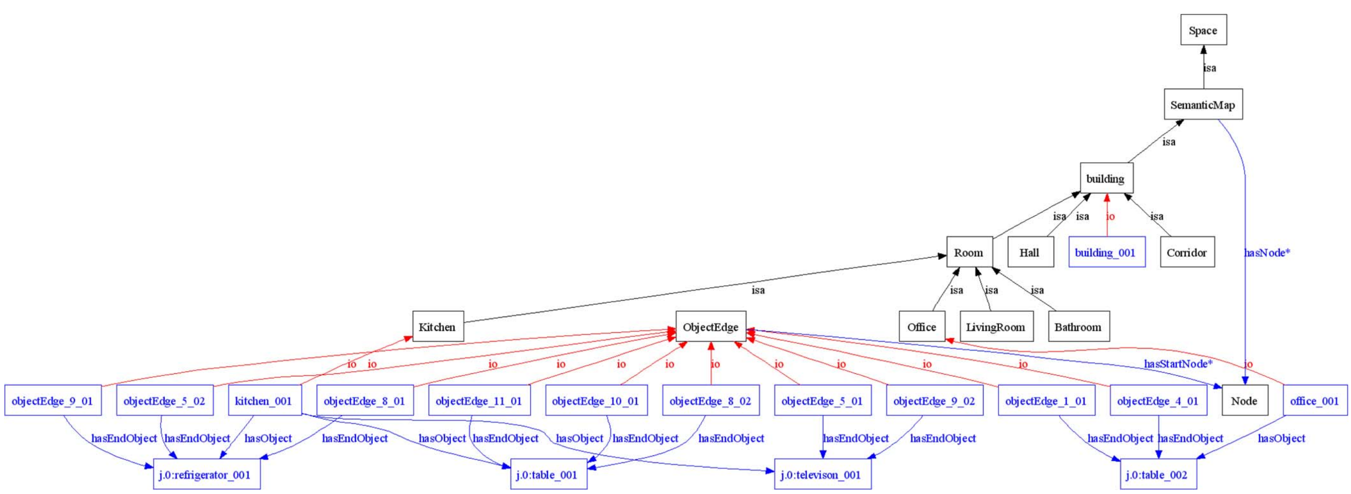


Fig. 8. Ontology schema and the instance of the semantic map (KLevel_{33}).

robot's perception is incomplete and subjected to change. The robot was given a map of the experimental space in the form of instances in the ontology schema of the space KClass_3 (S), as shown in Fig. 9. In the previous step, the robot explored the experimental environment and generated the ontology instances of the space KClass_3 and the object KClass_2. When the robot recognized a `cup` and a `kettle` in the `kitchen`, it generated their spatial context: the `cup` is at the `left` of the `kettle`. The robot was also assumed to have instantiated ontological instances of the object in the object KClass $(\mathrm{O}_2 : \mathrm{KLevel}_{22})$ in the space during the previous step, as shown in Fig. 10. The semantic map $(\mathrm{S}_3 : \mathrm{KLevel}_{33})$ contains `kitchen_001` and `office_001`, with `kitchen_001` containing the objects `cup_001` and `table_001`.

Next, the robot was ordered to "find a cup," so it moved to the `kitchen` where the `cup` was last found. To find `cup_001`, the robot infers that it must move from `office_001` to `kitchen_001` via `node_008`. In the context ontology instance, `cup_001` is on `table_001`. `Table_001` is linked to `node_008`, and `node_008` is included in `kitchen_001`. Therefore, the robot determines that it should proceed to `kitchen_001 to` find the `cup`, as shown in Fig. 11. The next stage included three knowledge reasoning steps. As the robot moved to the `kitchen`, it monitored its location to ensure that it was moving

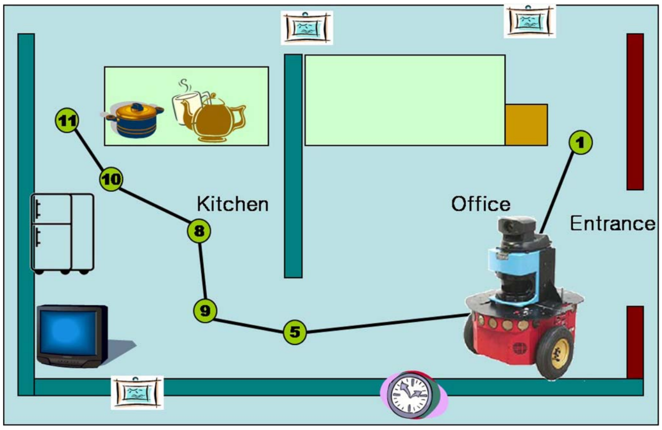


Fig. 9. Experimental scenario.

(a) (b)

Fig. 10. (a) Previous episode in the ontology instance. (b) Current situation where the cup is partially occluded. This is unknown to the robot. (a) Previous step. (b) Current situation.

towards the `kitchen` (step 1). When the robot arrived at the `kitchen`, it attempted to identify the `cup` but failed because the `cup` was occluded by the `kettle`, as shown in Fig. 10(b). The robot then queried what action was required to find the `cup` (step 2). Finally, the robot found the `cup` and updated its spatial and temporal contexts (step 3).

Fig. 12 shows the action class ontology and the ontological instances for the `delivery` task (OLayer_{533}) using an abductive event calculus planner that plans by a hierarchical abstraction of the space elements (KClass_3) and behaviors (joint of action hierarchy: A_1, A_2, and A_3) [39]. The high-level task of `delivery` ($\mathrm{A}_3 : \mathrm{KLevel}_{53}$) is composed of instances (OLayer_{523}) of the middle-level subtasks ($\mathrm{A}_2 : \mathrm{KLevel}_{52}$) such as `gotoGoalSpace`, `findObject`, and `generateContext`. Each task is also planned as a sequence of primitive behavior instances (OLayer_{513}) such as `gotoNode` and `recognizeObject` ($\mathrm{A}_3 : \mathrm{KLevel}_{51}$). To find `cup_001` in the `kitchen`, the TM [46] requested that the hierarchical planner creates a plan. These sequences of actions become the ontological instances of action. At the task level, the delivery task for `cup_001` is instantiated with three sequenced subtasks: `gotoGoalSpace`, `findObject`, and `generateContext`. The `findObject` subtask is, in turn, composed of two further subtasks: `gotoNode`, which tells the robot to proceed to `node_009`, and `recognizeObject` to find `cup_001`. These subtasks are also composed of primitive robot behaviors such as `turn`, `goto`, and `takePicture`, as shown in Fig. 13.

D. Examples of Knowledge Reasoning

1) Step 1: Is It on the Way to the Kitchen?: En route to the kitchen, the TM queries, "is the robot on the way to the kitchen?" To validate that the robot is on the correct route, the KM infers the landmarks in the ontology instance layer of the topological map (S_2). The KM recommends that the TM recognize `refrigerator_001` and `television_001` based on the rule, "**if *A* exists, if *A* is linked to the *Target_node*, and if *A* is recognized, then the robot is going the right way toward the *Target_node*.**" Additionally, `refrigerator_001` and `television_001` are linked to `Node_009`, which is included within the kitchen. Once `refrigerator_001` or `television_001` is recognized, the TM can then confirm that the robot is heading towards `kitchen_001`, as shown in Fig. 14.

2) Step 2: When No Cup Is Recognized, What Action Is Required to Find the Cup?: When the robot arrives at the target `node` in the `kitchen`, the TM performs the next task: recognize a `cup` in the environment. As shown in Fig. 10, in this case, the `cup` is partially obscured, and the robot fails to recognize it. Following this, the TM queries the KM "what action is required to find the `cup`?" Based on the rule "**if *A* is an instance of the spatial context with *target*, then recognize *A***" and based on the facts that `cup_001` is to the `left` of the `kettle_001`, the KM recommends that the TM should recognize `kettle_001`. Although the goal is for the robot "to find a `cup`," this rule recommends that the robot recognizes the `kettle` rather than the `cup`, offering an opportunity to complete the task. Also, the `table` is another candidate that would help in identifying the `cup`, as the unidirectional rules can be used alongside the spatial context instances of the fact that the "`cup` is on the `table`." The spatial context implies a tendency of co-occurrence between two objects, whose relationship is not a random distribution [50]. In this experiment, there are tendencies of co-occurrence between the `cup`, `table`, and/or `kettle`. The results of the object recognition for both the table and/or `kettle` provide a cue to find the `cup` in spite of occlusion. Once the KM has given its recommendation, the TM requests a vision module to recognize `kettle_001`. If the recognition is successful, then the KM recommends that the robot approaches `kettle_001` instead of the target object, using the rule: "**if *A* is the instance of the spatial context with the *target*, then recognize *A*; if *A* is recognized, then approach *A*, and recognize the *target*.**" The TM can then approve the recommendations and can replan to achieve the overall goal by adding new tasks. Fig. 13 shows that the updated action ontology instances for `approachObject` regarding `kettle_001` and `recognizeObject` for `cup_001` have been added to the instances within the subtask level ($\mathrm{A}_2 : \mathrm{KLevel}_{52}$) in the action KClass. Due to these new instances, the robot approached the `kettle` rather than the `cup`, and it tries again to recognize the `cup`. Finally, `cup_001` is successfully recognized even when it is partially occluded, as shown in Fig. 15.

The procedures set out in step 2 reveal that the objects can be recognized using the bidirectional reasoning rules. The unidirectional reasoning rules are also utilized at every object recognition step. When a "`Find cup`" task is requested, the vision module first asks the KM to obtain a `visual concept` of the `cup` and to use an `extractor algorithm`, which includes the properties from the perceptual feature level ($\mathrm{f}_1 : \mathrm{KLevel}_{13}$) found in the feature knowledge class (KClass_1). For the purposes of matching, a hue of `GreenishBlue` and more than

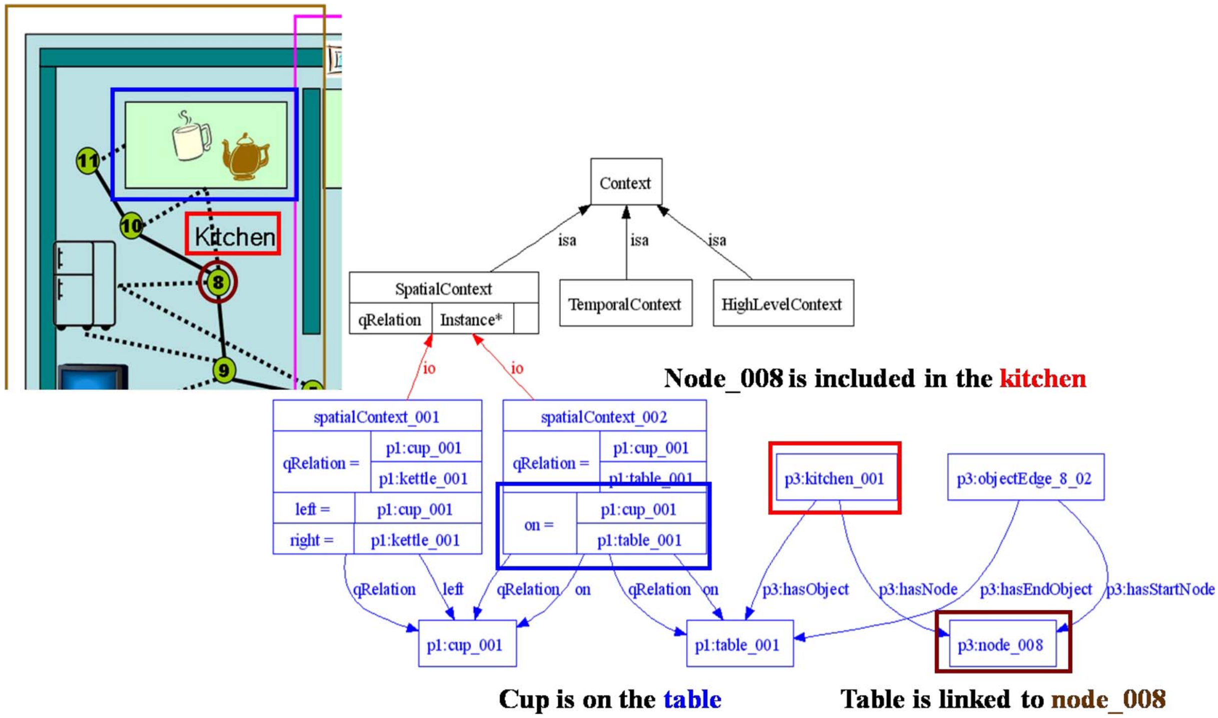


Fig. 11. Ontology schema and the instance defining where to go to find the cup.

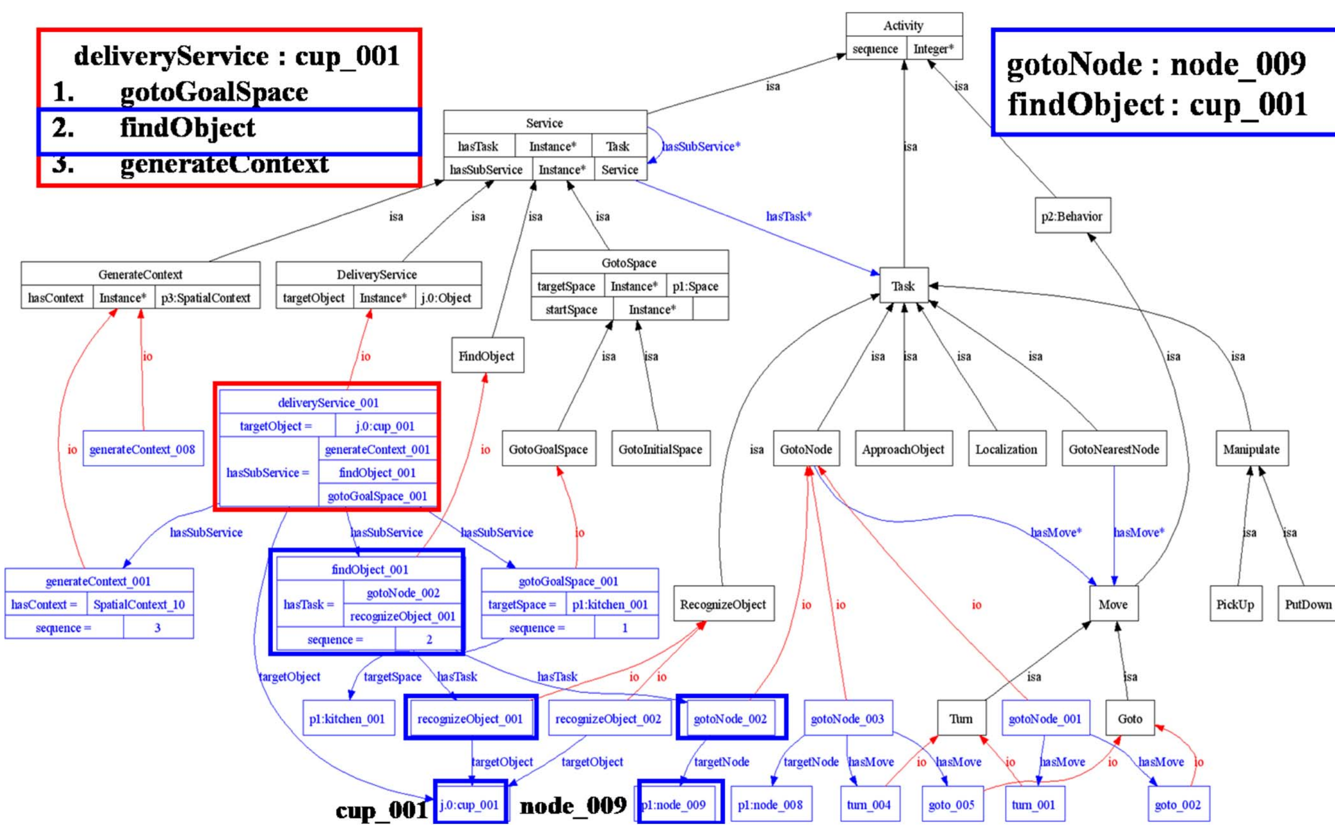


Fig. 12. Ontology schema and the instance for the delivery service.

five identical SIFT features must be identified according to the properties contained in the perceptual concept (f_3 : KLevel) of the cup. Then, the `hueExtractor` and `SIFT extractor` commands are executed to identify such `visual features` for the candidate segment blocks found in the current visual scene. If these visual features are identified in a segment, then the segment is registered as a `cup` by the matching rule using a set of unidirectional reasoning rules.

If these visual features are only partially rather than completely identified, then the vision module asks the KM to

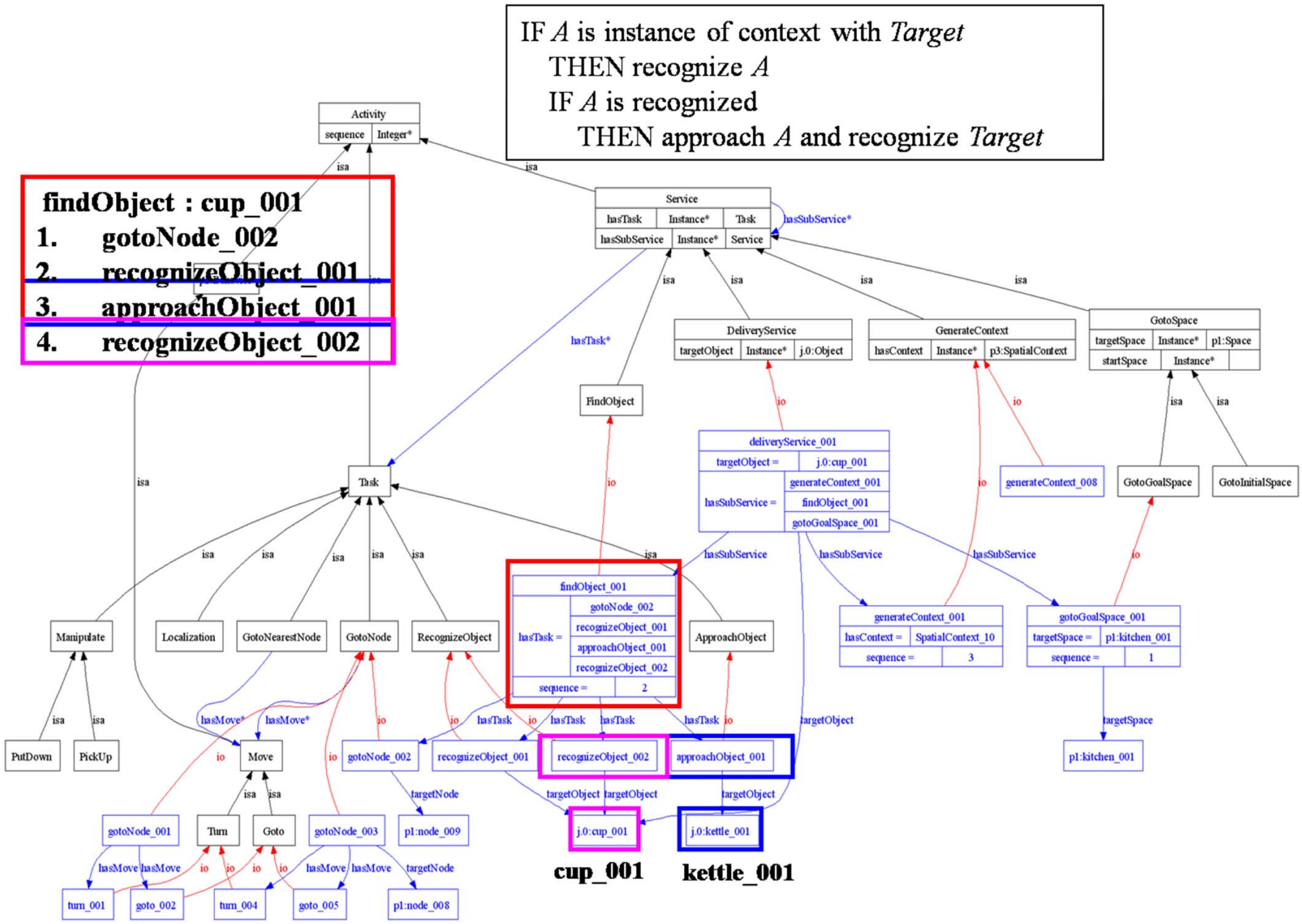


Fig. 13. Ontology schema, instance, and rule for adding new tasks to find the cup near the kettle.

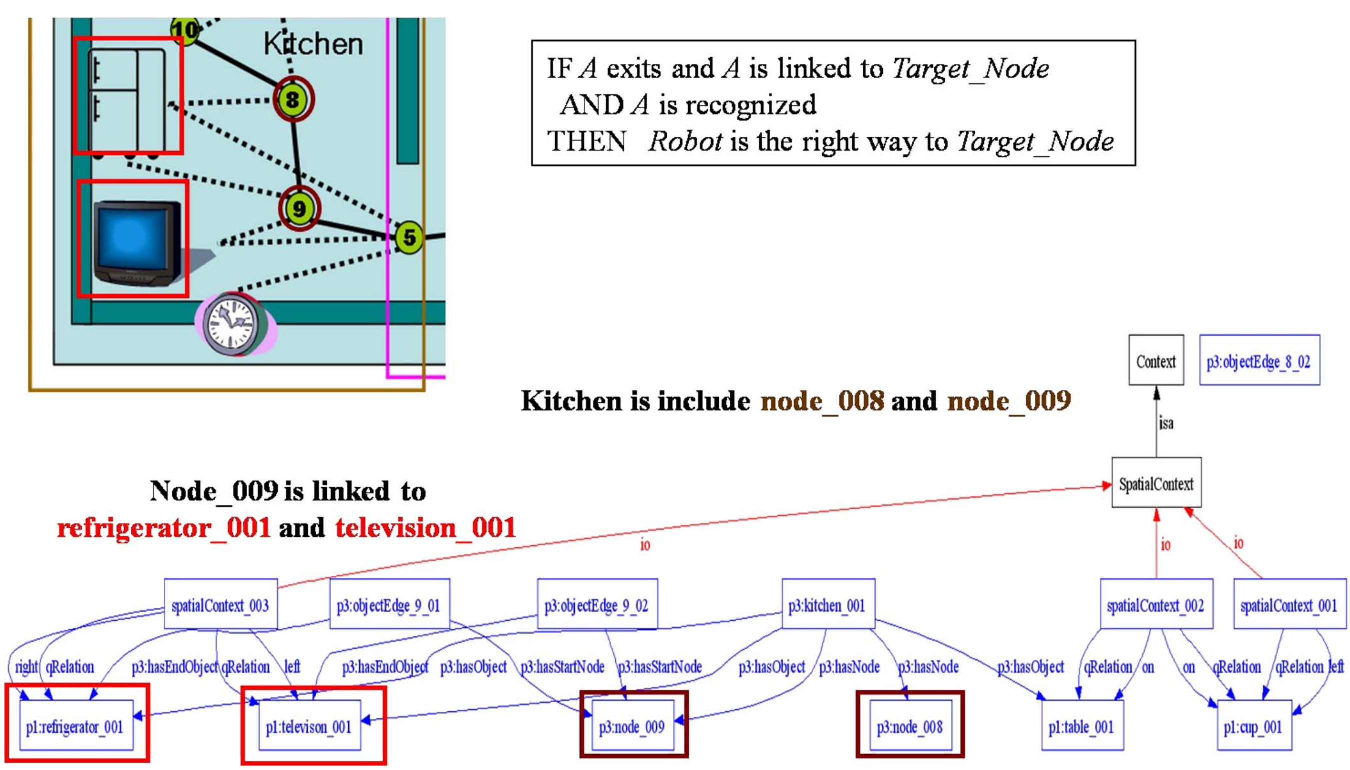
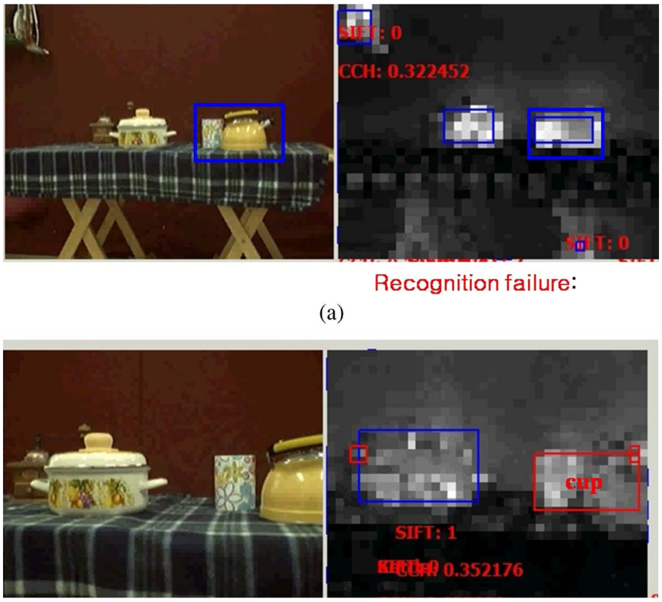


Fig. 14. Ontology schema, instance, and rule for the case en route to the kitchen.



(a)

(b)

Fig. 15. Recognition results. (a) Recognition failure: cup is partially occluded. (b) Recognition success: approach kettle, and recognize cup again.

search for additional relevant facts that can be inferred using the bidirectional rules $(\mathrm{R}^{\mathrm{L}})$ that have conditions in terms of space (S:KClass$_3$) and context (C:KClass$_4$) knowledge classes. Consequently, the bidirectional reasoning rules infer that the `kettle` is an object that is relevant to the `cup`, and the KM requests that the vision module recognizes the `kettle` in the current visual scene. In addition, the KM also requests that the vision module recognizes the `table` in the current visual scene. This is because the `table` is another candidate for `cup` identification using the unidirectional rules, as the spatial context instances include the fact that the "`cup` is `on` the `table`."

3) Step 3: Context Generation: As a result of the "`Find cup`" task, the instances of the spatial and temporal contexts are generated. The results are as follows: in the instance found in the spatial context level, `Cup_001` is to the `rear` of `kettle_001`, and in the temporal context level, `kettle_001` is `dynamic`, i.e., `kettle_001` has moved because, in the current instance, the spatial context between `kettle_001` and `cup_001` differs from that of the previous spatial context instance.

These experimental results show that the service robot utilized the ontology-based robot knowledge to successfully complete the tasks.

VI. Conclusion

In this paper, an OUR-K system has been developed, which can be used by service robots in real environments. The proposed knowledge framework describes the service robot knowledge that is required to integrate the low-level knowledge about the perceptual features, part objects, metric maps, and primitive behaviors with high-level knowledge about the perceptual concepts, objects, semantic maps, tasks, and contexts, and it associates with these robot knowledge classes. This robot knowledge framework enabled our robot to complete a "find cup" task in spite of hidden or partial data. Moreover, the proposed knowledge framework allowed the robots to solve most queries by applying the simple unidirectional rules within the same knowledge classes and the bidirectional rules among different knowledge classes. The experimental results reveal that the service robot exhibited a great ability to complete the complex tasks successfully.

Appendix

This section defines a formal representation of OUR-K. In this paper, the parentheses [i.e., ()] denote functions, properties, or relations; the curly braces (i.e., { }) denote sets; and the angle braces (i.e. $\langle\rangle$) denote n-tuples.

Knowledge description and knowledge associations are two core components of OUR-K, as shown in Definition 1.

Definition 1: Ontology-based unified knowledge framework:

$$OUR\text{-}K := \langle KDescription, KAssociations\rangle.$$

Such that the robot knowledge is described into KDescription and KAssociations that are association methods within KDescription.

Definition 2 outlines the knowledge description. KDescription consists of five classes of knowledge, including feature, object, space, action, and context, as shown in the following discussion.

Definition 2: KDescription consists of five knowledge classes:

$$KDescription = \{KClasses_i | \, 1 \leq i \leq 5\}.$$

We define a knowledge class $KClass_i$ for $i \in N$ (set of natural numbers), $1 \leq i \leq 5$, where index i is for five knowledge classes.

$KClass_1$ is a class of knowledge for the feature (f);
$KClass_2$ is a class of knowledge for the object (O);
$KClass_3$ is a class of knowledge for the space (S);
$KClass_4$ is a class of knowledge for the context (C);
$KClass_5$ is a class of knowledge for the action (A).

As shown in Definition 3, OUR-K includes three knowledge association methods: logical inference, Bayesian inference, and heuristics.

Definition 3: Methods of knowledge associations:

$$KAssociations = \{Logical\ inference, Bayesian\ inference, Heuristics\}.$$

Definitions 4 and 5 define a set of knowledge levels (KLevel) and ontological layers (OLayer). Each knowledge class consists of nine ontological layers (3 KLevel $*$ 3 OLayer). Three are comprised of low-level knowledge, three are comprised of middle-level knowledge, and three are comprised of high-level knowledge. Each KLevel has three ontological layers: a metaontology layer, an ontology schema layer, and an ontology instance layer.

Definition 4: A set of a level of knowledge consists of three knowledge layers, except the middle-level knowledge of the feature class:

$$KClass_i = \{KLevel_{ij} | 1 \leq i \leq 5, 1 \leq j \leq 3, \backslash KLevel_{12}\}.$$

We define a knowledge level for i, $j \in N$ (set of natural numbers), $1 \leq i \leq 5$, $1 \leq j \leq 3$, and $\backslash ij = 12$, where index j is for three knowledge layers.

$KLevel_{i1}$ is a knowledge level for the low-level knowledge (f_1, O_1, S_1, C_1, A_1);

$KLevel_{i2}$ is a knowledge level for the middle-level knowledge, except the feature class (O_2, S_2, C_2, A_2);

$KLevel_{i3}$ is a knowledge level for the high-level knowledge (f_3, O_3, S_3, C_3, A_3).

Definition 5: A set of ontology layers consists of three ontology layers:

$$KLevel_{ij} = \{OLayer_{ijk} | 1 \leq i \leq 5, 1 \leq j \leq 3, 1 \leq k \leq 3, \backslash OLayer_{12k}\}.$$

We define a knowledge layer for i, j, $k \in N$ (set of natural numbers), $1 \leq i \leq 5$, $1 \leq j \leq 3$, $1 \leq k \leq 3$, and $\backslash ijk = 12k$, where index k is for three ontology layers.

$OLayer_{ij1}$ is an ontology layer for the metaontology layer, except the middle-level knowledge of the feature class ($f_{j1}, O_{j1}, S_{j1}, C_{j1}, A_{j1}$);

$OLayer_{ij2}$ is an ontology layer for the ontology layer, except the middle-level knowledge of the feature class ($f_{j2}, O_{j2}, S_{j2}, C_{j2}, A_{j2}$);

$OLayer_{ij3}$ is an ontology layer for the ontology instance layer, except the middle-level knowledge of the feature class ($f_{j3}, O_{j3}, S_{j3}, C_{j3}, A_{j3}$).

Definition 6 represents the basic ontological elements of the knowledge class structure OLayer$_{ijk}$. These are composed of five tuples: concept, relation, concept hierarchy, relation hierarchy, and axiom. The level of knowledge for each layer can be organized using these five tuples.

Definition 6: The ijkth ontology layer in OUR-K consists of five tuples:

$$OLayer_{ijk} = \left\langle Cp_{ijk}, Rel_{ijk}, H^C_{ijk}, H^R_{ijk}, A^0_{ikj} \right\rangle$$

for $1 \leq i \leq 5$, $1 \leq j \leq 3$, and $1 \leq k \leq 3$;

Cp_{ijk} is a set of concepts in $OLayer_{ijk}$;

Rel_{ijk} is a set of relation in $OLayer_{ijk}$;

H^C_{ijk} is a set of concept hierarchies in $OLayer_{ijk}$;

H^R_{ijk} is a set of relation hierarchies in $OLayer_{ijk}$;

A^0_{ijk} is a set of axioms.

Definitions 7 and 8 describe the axioms that clearly specify the concepts and relations. Here, the axioms are the relationships between the concepts in the same OLayer$_{ijk}$.

Definition 7: The set of axioms of each OLayer is a set of sentences Λ that follows the representation of the logical language. Λ is represented by the structures of $OLayer_{ijk}$ $(f_{ijk}, O_{ijk}, S_{ijk}, C_{ijk}, A_{ijk})$ (elements of Cp_{ijk} and Rel_{ijk} of $OLayer_{ijk}$) for $1 \leq i \leq 5$, $1 \leq j \leq 3$, and $1 \leq k \leq 3$, which are the elements in the same $OLayer_{ijk}$. Also, a sentence of Λ specifies the meaning of the elements by describing the relationship of the elements in an $OLayer_{ijk}$. Any sentence in Λ cannot be entailed by other sentences in Λ.

Definition 8: Axioms are a structure of three tuples:

$$A^{KA} := \langle AI, \Lambda, \alpha \rangle$$

AI is a set of axiom identifiers;

Λ is a set of logical sentences;

α is a set of axiom mapping functions: $\alpha : AI \Rightarrow \Lambda$.

Definitions 9–13 define the rules for reasoning between OLayer$_{ijk}$ in the same knowledge level and in different knowledge levels as a knowledge association method. While axioms are defined in one OLayer$_{ijk}$, the rules are defined between various OLayer$_{ijk}$. These rules can be used to infer the relationships between elements that exist in several OLayer$_{ijk}$. Definition 13 presents a metarule that serves as a template form for new rules.

Definition 9: A structure of knowledge association rules for the logical inference within knowledge descriptions consists of two sets of rules:

$$R^{KA} = \{R^U, R^B\}.$$

R^U is the rules between the hierarchical layers of $KLevel_{ij}$ in the same knowledge classes for unidirection;

R^B is the rules among the hierarchical layer of $KLevel_{ijk}$ in the different knowledge classes for bidirection.

Definition 10: The set of layer rules is a set of sentences $\mathcal{N}$ that follows the representation of the logical language. $\mathcal{N}$ is represented by two tuples of $OLayer_{ijk}$ (elements of C_{ijk} and Rel_{ijk} of $OLayer_{ijk}$). A sentence of $\mathcal{N}$ represents the relationship between the two elements of $OLayer_{ij}$ (C_{ijk} and Rel_{ijk}), and it is used to entail other concepts or relations. The rule should include at least two elements: one from an $OLayer_{ijk}$ and the other from another $OLayer_{ijk}$.

Definition 11: The unidirectional rules between the hierarchical layers are a structure of three tuples:

$$R^U := \langle RI^U, \mathcal{N}^U, \beta^U \rangle$$

RI^U is a set of layer rule identifiers for unidirection;

$\mathcal{N}^U$ is a set of logical sentences for the layer rules for unidirection;

β^U is a set of layer rule mapping functions for unidirection:

$$\beta^U : RI^U \Rightarrow \mathcal{N}^U.$$

Definition 12: The bidirectional rules between levels of knowledge are a structure of three tuples:

$$R^B := \langle RI^B, \mathcal{N}^B, \beta^B \rangle$$

RI^B is a set of association rule identifiers for bidirection;

$\mathcal{N}^B$ is a set of logical sentences for the association rules for bidirection;

β^B is a set of association rule mapping functions for bidirection:

$$\beta^B : RI^B \Rightarrow \mathcal{N}^B.$$

Definition 13: A metarule is a rule template in the form of

$$P1 \wedge P2 \wedge \cdots \wedge Pm \wedge U \Rightarrow Q1 \wedge Q2 \wedge \cdots \wedge Qn$$

where Pi (for $i = 1, \ldots, m$) and Qi (for $i = 1, \ldots, n$) are either concepts or relations defined in the metaontology layer ($OLayer_{ijk}$, where $i = 1, 2, 3, 4, 5$ and $j = 1, 2, 3$) and U presents the additional user-defined predicates.

REFERENCES

[1] K. Kawamura, R. T. Pack, and M. Iskarous, "Design philosophy for service robots," *Robot. Auton. Syst.*, vol. 18, no. 1/2, pp. 109–116, Jul. 1996.

[2] K. Severinson-Eklundh, A. Green, and H. Hüttenrauch, "Social and collaborative aspects of interaction with a service robot," *Robot. Auton. Syst.*, vol. 42, no. 3, pp. 223–234, Mar. 2003.

[3] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2003, ch. 2.

[4] I. H. Suh, G. H. Lim, W. Hwang, H. Suh, J. H. Choi, and Y. T. Park, "Ontology-based multi-layered robot knowledge framework (OMRKF) for robot intelligence," in *Proc. IEEE/RSJ Int. Conf. IROS*, San Diego, CA, 2007, pp. 429–436.

[5] J. Hertzberg and A. Saffiotti, "Using semantic knowledge in robotics," *Robot. Auton. Syst.*, vol. 56, pp. 875–877, Nov. 2008.

[6] A. K. Mackworth, "On seeing robots," in *Computer Vision: Systems, Theory, and Applications*, A. Basu and X. Li, Eds. Singapore: World Scientific, 1993, pp. 261–268.

[7] R. Pfeifer and C. Scheier, *Understanding Intelligence*. Cambridge, MA: MIT Press, 2001, ch. 3.

[8] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Trans. Robot. Autom.*, vol. RA-2, no. 1, pp. 14–23, Mar. 1986.

[9] R. C. Arkin, *Behavior-Based Robotics*. Cambridge, MA: MIT Press, 1998.

[10] L. Rai and S. J. Kang, "Knowledge-based integration between virtual and physical prototyping for identifying behavioral constraints of embedded real-time systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 4, pp. 754–769, Jul. 2009.

[11] C. Y. Chang, J. P. Sheu, Y. C. Chen, and S. W. Chang, "An obstacle-free and power-efficient deployment algorithm for wireless sensor networks," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 4, pp. 795–806, Jul. 2009.

[12] N. Maillot, M. Thonnat, and A. Boucher, "Towards ontology-based cognitive vision," *Mach. Vis. Appl.*, vol. 16, no. 1, pp. 33–40, Dec. 2004.

[13] R. Mendoza and M. Williams, "Ontology based object categorisation for robots," in *Proc. Conf. Res. Practice Inf. Technol. Ser.*, 2005, vol. 58, pp. 65–80.

[14] Y. C. Go and J. Sohn, "Context modeling for intelligent robot services using rule and ontology," in *Proc. 7th ICACT*, 2005, vol. 2, pp. 813–816.

[15] C. Anagnostopoulos and S. Hadjiefthymiades, "Advanced inference in situation-aware computing," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 5, pp. 1108–1115, Sep. 2009.

[16] W. Dargie, "Adaptive audio-based context recognition," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 4, pp. 715–725, Jul. 2009.

[17] S. Cambon, R. Alami, and F. Gravot, "A hybrid approach to intricate motion, manipulation and task planning," *Int. J. Robot. Res.*, vol. 28, no. 1, pp. 104–126, Jan. 2009.

[18] E. Remolina and B. Kuipers, "Towards a formalization of the spatial semantic hierarchy," in *Proc. 4th Symp. Logical Formalizations Commonsense Reason.*, London, U.K., Jan. 1998.

[19] B. Krieg-Bruckner, U. Frese, K. Luttich, C. Mandel, T. Mossakowski, and R. J. Ross, "Specification of an ontology for route graphs," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2005, pp. 390–412.

[20] J. Bateman and S. Farrar, "Modelling models of robot navigation using formal spatial ontology," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2004, pp. 366–389.

[21] R. Chatterjee, I. Takao, F. Matsuno, and S. Tadokoro, "Robot description ontology and bases for surface locomotion evaluation," in *Proc. IEEE Int. Workshop Safety, Security Rescue Robot.*, Kobe, Japan, Jun. 2005, pp. 242–247.

[22] L. N. Soldatova, A. Clare, A. Sparkes, and R. D. King, "An ontology for a robot scientist," *Bioinformatics*, vol. 22, no. 14, pp. e464–e471, Jul. 2006.

[23] S. Vasudevan, S. Gachter, M. Berger, and R. Siegwart, "Cognitive maps for mobile robots—An object based approach," in *Proc. IEEE/RSJ Int. Conf. IROS*, Beijing, China, 2006, pp. 359–371.

[24] S. Coradeschi and A. Saffiotti, "Perceptual anchoring of symbols for action," in *Proc. 17th IJCAI Conf.*, Seattle, WA, 2001, pp. 407–412.

[25] W. Hwang, J. Park, H. Suh, H. Kim, and I. H. Suh, "Ontology-based framework of robot context modeling and reasoning for object recognition," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2006, pp. 596–606.

[26] E. Wang, Y. S. Kim, H. S. Kim, J. H. Son, S. Lee, and I. H. Suh, "Ontology modeling and storage system for robot context understanding," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2005, pp. 922–929.

[27] E. A. Topp, H. Huttenrauch, H. Christensen, and K. Severinson Eklundh, "Bringing together human and robotic environment representations—A pilot study," in *Proc. IEEE/RSJ Int. Conf. IROS*, Beijing, China, 2006, pp. 4946–4952.

[28] O. Martínez Mozos, R. Triebel, P. Jensfelt, A. Rottmann, and W. Burgard, "Supervised semantic labeling of places using information extracted from sensor data," *Robot. Auton. Syst.*, vol. 55, no. 5, pp. 391–402, May 2007.

[29] A. Poncela, C. Urdiales, E. J. Pérez, and F. Sandoval, "A new efficiency-weighted strategy for continuous human/robot cooperation in navigation," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 3, pp. 486–500, May 2009.

[30] M. MacMahon, B. Stankiewicz, and B. Kuipers, "Walk the talk: Connecting language, knowledge, and action in route instructions," in *Proc. Nat. Conf. AAAI*, 2006, pp. 1475–1482.

[31] S. Lauria, T. Kyriacou, G. Bugmann, J. Bos, and E. Klein, "Converting natural language route instructions into robot executable procedures," in *Proc. IEEE Int. Workshop Roman*, Berlin, Germany, 2002, pp. 223–228.

[32] S. Tellex and D. Roy, "Spatial routines for a simulated speech-controlled vehicle," in *Proc. 1st ACM SIGCHI/SIGART Conf. Human–Robot Interaction*, 2006, pp. 156–163.

[33] F. Baader, D. Calvanese, D. Mcfuinness, D. Nardi, and P. Patel-schneider, *The Description Logic Handbook*. Cambridge, U.K.: Cambridge Univ. Press, 2003, ch. 2.

[34] J. Schoening, *IEEE P1600.1 Standard Upper Ontology Working Group*. [Online]. Available: http://suo.ieee.org

[35] T. Gruber, *What is an Ontology*, 2001. [Online]. Available: http://www-ksl.stanford.edu/kst/what-is-an-ontology.html

[36] D. George, "How the brain might work: A hierarchical and temporal model for learning and recognition," Ph.D. dissertation, Dept. Elect. Eng., Stanford Univ., Stanford, CA, 2008.

[37] C. Geib, K. Mourao, R. Petrick, N. Pugeault, M. Steedman, N. Krueger, and F. Wörgötter, "Object action complexes as an interface for planning and robot control," in *Proc. IEEE RAS Int Conf. Humanoid Robots*, Genova, Italy, 2006.

[38] G. H. Lim, J. L. Chung, G. G. Ryu, J. B. Kim, S. H. Lee, S. Lee, I. H. Suh, J. H. Choi, and Y. T. Park, "Ontological representation of vision-based 3D spatio-temporal context for mobile robot applications," in *Proc. 12th Int. Symp. AROB*, Beppu, Japan, 2007, pp. 25–27.

[39] Y. Park, I. H. Suh, and B. Choi, "Hierarchical abstraction of world elements and behaviors for efficient task planning of a mobile robot," in *Proc. IEEE/RSJ Int. Conf. IROS*, Nice, France, 2008, pp. 2585–2592.

[40] G. H. Lim and I. H. Suh, "Weighted action-coupled semantic network (wASN) for robot intelligence," in *Proc. IEEE/RSJ Int. Conf. IROS*, Nice, France, 2008, pp. 2035–2040.

[41] C. Yi, I. H. Suh, G. H. Lim, and B. Choi, "Bayesian robot localization using spatial object contexts," in *Proc. IEEE/RSJ Int. Conf. IROS*, St. Louis, MO, 2009, pp. 3467–3473.

[42] C. Yi, I. H. Suh, G. H. Lim, and B. Choi, "Active-semantic localization with a single consumer-grade camera," in *Proc. IEEE Int. Conf. SMC*, San Antonio, TX, 2009, pp. 2161–2166.

[43] E. Bozsak, M. Ehrig, S. Handschuh, A. Hotho, A. Maedce, B. Motik, D. Oberle, C. Schmitz, S. Staab, and L. Stojanovic, "KAON—Towards a large scale semantic web," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2002, pp. 304–313.

[44] M. Kim, S. Kim, S. Park, M. T. Choi, M. Kim, and H. Gomaa, "Service robot for the elderly," *IEEE Robot. Autom. Mag.*, vol. 16, no. 1, pp. 34–45, Mar. 2009.

[45] R. T. Pack, D. M. Wilkes, and K. Kawamura, "A software architecture for integrated service robot development," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 1997, vol. 4, pp. 3774–3779.

[46] J. Lee and B. Kwak, "A task management architecture for control of intelligent robots," in *Lecture Notes in Computer Science*, vol. 4088. Berlin, Germany: Springer-Verlag, Aug. 2006, pp. 59–70.

[47] M Huber, "JAM: A BDI-theoretic mobile agent," in *Proc. 3rd Int. Conf. Agents*, Seattle, WA, 1999, pp. 236–243.
[48] J. Melchert, S. Coradeschi, and A. Loutfi, "Knowledge representation and reasoning for perceptual anchoring," in *Proc. 19th IEEE ICTAI*, Patras, Greece, Oct. 2007, pp. 129–136.
[49] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.
[50] J. L. Davenport and M. C. Potter, "Scene consistency in object and background perception," *Psychol. Sci.*, vol. 15, no. 8, pp. 559–564, Aug. 2004.
[51] J. F. Allen, "Planning as temporal reasoning," in *Proc. 2nd Int. Conf. Principles Knowl. Representation Reason.*, Cambridge, MA, Apr. 1991.
[52] S. H. Lee and I. H. Suh, "Bayesian network-based behavior control for Skilligent robot," in *Proc. IEEE/RSJ ICRA*, Kobe, Japan, 2009, pp. 1448–1454.
[53] K. Eshghi, "Abductive planning with event calculus," in *Proc. 5th Int. Conf. Logic Program.*, 1988, pp. 562–579.
[54] I. Bratko, *Prolog Programming for Artificial Intelligence*, 3rd ed. London, U.K.: Pearson Education, 2001, p. 57.
[55] R. J. Brachman and H. J. Levesque, *Knowledge Representation and Reasoning*. San Mateo, CA: Morgan Kaufmann, 2004.
[56] M. Esposito, "An ontological and non-monotonic rule-based approach to label medical images," in *Proc. IEEE 3rd SITIS*, Shanghai, China, Dec. 2007, pp. 603–611.
[57] M. Kifer, "Nonmonotonic reasoning in FLORA-2," in *Lecture Notes in Computer Science*, vol. 3662. Berlin, Germany: Springer-Verlag, 2005, pp. 390–412.
[58] M. E. Munich, J. Ostrowski, and P. Pirjanian, "ERSP: A software platform and architecture for the service robotics industry," in *Proc. IEEE/RSJ Int. Conf. IROS*, Edmonton, AB, Canada, 2005, pp. 460–467.

Gi Hyun Lim (S'07) received the B.S. degree in metallurgical engineering and the M.S. degree in electronics and computer engineering from Hanyang University, Seoul, Korea, in 1997 and 2007, respectively, where he is currently working toward the Ph.D. degree in the Intelligence and Communications for Robot Laboratory.

From 1997 to 1998, he was an Engineer with the Dongbu Electronics, Ltd., Chungcheongbuk-do, Korea, where he was involved in research on semiconductor factory automation. From 1999 to 2005, he was a Senior Research Engineer at a venture business, where he was involved in research on real-time operating systems and embedded systems.

Il Hong Suh (M'82–SM'03) received the B.S. degree in electronics engineering from the Seoul National University, Seoul, Korea, in 1977 and the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology, Seoul, in 1979 and 1982, respectively.

From 1985 to 2000, he was with the Department of Electrical Engineering and Computer Science, Hanyang University, Ansan, Korea, where he was a Full Professor. Since 2000, he has been with the Department of Electronics and Computer Engineering, Hanyang University, Seoul, where he is a Full Professor. From August 2004 to July 2006, he also served as the Dean of the College of Information and Communications, Hanyang University. He is also the Director of a National Robotics Education Center (Robotics Specialized Education Consortium, which is sponsored by the Korean Ministry of Knowledge and Economy). He has published more than 170 contributions in robotics and control. His research interests lie in the area of communications and intelligence for robots, including web-based control of robots, ontology-based robot intelligence, context-adaptive action-coupled perception and learning, and robot software platforms.

Dr. Suh served as the President of the Systems and Control Society of the Korea Institute of Telecommunications Engineers in 2005–2007, and he also served as the President of the Korea Robotics Society in 2008. He served as the Chair of the steering committee for the Robot Unified Platform Initiative organized by the former Korean Ministry of Information and Communications for 2006–2007. He was listed in Who's Who in the World in 2008 and 2009. He has been involved in a number of Korean National Projects such as the Intelligent Robotics Frontier Research Program for the 21st Century and the Ubiquitous Robotics Companion.

Hyowon Suh received the B.S. degree in mechanical engineering from Yonsei University, Seoul, Korea, in 1981, the M.S. degree in mechanical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Seoul, in 1983, and the Ph.D. degree in industrial engineering from West Virginia University, Morgantown, in 1991.

He is currently a Professor with the Department of Industrial Engineering, KAIST, Daejeon, Korea. From 1983 to 1987, he was an Assistant Research Engineer with the Central Research Center, Daewoo Heavy Industries, Ltd., Inchon, Korea. From 1992 to 1995, he was a Principal Research Engineer with the Production System Center, Korea Institute of Industrial Technology, Korea. He has published several papers in ontology applications. His research interest is ontology applications in engineering.