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

Improvisational goal-oriented action recommendation under Incomplete Knowledge Base

Gi Hyun Lim, *Member, IEEE,* Il Hong Suh, *Senior Member, IEEE,*

***Abstract*— Robots need to have knowledge of their environment to be able to successfully complete service tasks. Most knowledge inference mechanisms assume complete and correct knowledge about the environment. Real world environments are often uncertain and only partially observable. Thus, intelligent service robots may have an incomplete knowledge base which includes true positives as well as false negatives and false positives. False negatives and false positives can prevent service robots from completing their service tasks. In the field of logical inference, false positives are a more significant problem compared to false negatives. A weighted ontology and association mechanism was proposed in previous research which recommended improvisational goal-oriented actions that could be applied in the case of false negatives. However, false positives are not usually matched in the existing ontological semantic network. Consequently, the association mechanism does not work. To deal with false positives, the weighted ontology and association mechanism were extended by adding additional nodes which are associated with epistemic actions. The proposed method was successfully evaluated and verified through experiments; results show that almost all problems associated with false positives and false negatives were resolved.**

I. Introduction

To perform service tasks, intelligent agents such as service robots utilize knowledge to understand user intentions, perceive environments and build world models, make plans based on user requirements, and come up with their own atomic behaviors to carry out the plans. One of the main assumptions of classical logic is complete and correct knowledge about the world. However, in real environments, service robots often cannot construct a model of the entire world. For service robots, such environments are considered to be partially observable and uncertain. A robot may have an incomplete knowledge base that contains false negatives and false positives resulting from the uncertainties of observation of its own actions as well as unobserved external events resulting from the actions of other agents. As a result, a robot may fail to complete the service tasks. A goal-oriented improvisation based strategy could enable service robots to complete those tasks.

The two main approach types for planning and performing actions based on an incomplete knowledge base are logical approaches and probabilistic approaches, which include partially observable Markov decision processes. In a study regarding logical approaches, Morgenstern [1] proposed a flexible and expressive theory of action and planning for known precondition problems for actions and plans. Blyth [2] presented a planning methodology for handling uncertainty in the form of external events that are not completely predictable. He proposed two steps for planning goals and subgoals. The logic planner makes plans through backward chaining, and the subgoals are selected by analyzing the probability of success of the partial plan. Etzioni *et al.* [3] presented syntax and semantics for representing goals and actions under the assumption that incomplete information is available to the planner using the University of Washington Language (UWL). Herzig *et al.* [4] proposed a purely logical framework for planning in partially observable environments using epistemic logic S5. They proposed a method of representing the effects of physical or non-physical actions that change the external world as ontic effects, and actions that change a robot's knowledge stage as epistemic effects. These rich and flexible representations of logical approaches allow for compact encodings of planning in very large domains. Nevertheless, all of these logical approaches are limited by having incomplete information about the world, the robot's own actions, or external events; either robots cannot handle the uncertainty of observations or they can only address it in a simple way. To overcome this limitation, Kaelbling *et al.* [5] proposed a algorithm for solving partially observable Markov decision processes (POMDPs) to determine the optimal actions in partially observable stochastic domains. POMDPs provide a uniform treatment of action to gain information and to change the environment. However, POMDPs can only be applied in small domains due to the computational complexity [6], [7]. In order to obtain an optimal approach, POMDPs iterate all belief states with respect to all possible observations. Furthermore, POMDPs are limited in the types of relationships that they can represent. They can only represent single relationships, causal relationships, or single directional relationships, and cannot represent circular relationships.

As service robots acquire knowledge from their sensors, their knowledge base may be incomplete and include false negatives and false positives. This can prevent them from completing service tasks. For knowledge representation, we propose an approach that adopts memory retrieval based on the locality principle to deal with an incomplete knowledge base. In this case, locality is defined in terms of semantic networks, and not physical space. A semantic network is not just composed of concepts that represent all possible knowledge in a feasible manner; it also accounts for in-

This work was supported for the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Korea Ministry of Knowledge Economy

G. H. Lim is with the Department of Electronics & Computer Engineering, Hanyang University, Korea hmetal@hanyang.ac.kr

I. H. Suh is with the Division of Computer Science and Engineering, College of Engineering, Hanyang University, Korea ihsuh@hanyang.ac.kr

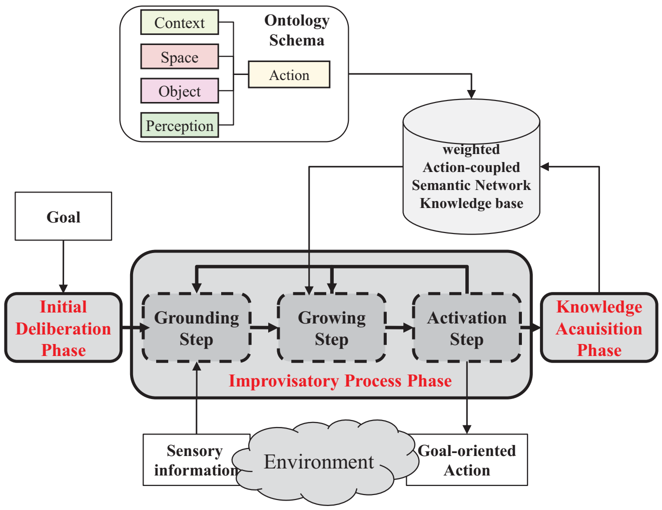

Fig. 1. System diagram of the inference framework.

dividuals as it matches this knowledge with observations. In this paper, the semantic network model is extended by adding weights. The spreading activation algorithm has been applied for knowledge reasoning in a wide range of semantic processing functions such as memory search and priming [8], [9], [10], [11], [12]. The approach proposed by Lim and Suh [12] infers alternative actions through the recursively spreading activation algorithm and weighted semantic networks. However, this approach can only handle false negatives that occur when targets are partially occluded or unobservable; it cannot handle false positives, such as an incorrect positive recognition result. Such false positives can lead to false reasoning consequences and are difficult to correct even when additional true negative results are available [13]. As false positives usually occur randomly [14], they may not be matched in existing semantic network. In such cases, the spreading activation algorithm cannot work, as it cannot handle false positives. In order to deal with false positives, this study extends the association mechanism by adding nodes that are coupled with epistemic actions. As a result, most issues caused by false positives are resolved.

II. Overall Framework

As shown in Fig. 1, the proposed framework is comprised of three phases: the initial deliberation phase, the improvisatory process phase, and the knowledge acquisition phase. First, goal and initial observations are given in the form of a weighted action-coupled semantic network (wASN) knowledge base that serves as long-term memory in the initial deliberation phase. The spreading activation algorithm is then executed over the whole wASN to find linked nodes from the initial observation node to the goal node, where linked nodes are subgoals for the given goal. Next, a board of processing (BoP), which plays the role of working memory, is initialized by copying the goal node, initial node, subgoal nodes, and their neighboring nodes from the wASN database to the BoP. In addition, action nodes associated with the goal node and its subgoal nodes are copied from wASN to BoP. Based on the robot ontology [15], [16], the wASN database stores the properties and weights of the ontology instances as a reference model for knowledge representation . Only the goal node, its associated nodes, and input nodes and their edges are copied from wASN to BoP to ensure that only goal-associated nodes are considered during the execution of the spreading activation algorithm. The next phase is to execute the spreading activation and to obtain actions based on the BoP, instead of the entire wASN. The improvisatory process phase continues if the initial action does not realize the goal. The three steps involved in the improvisatory process phase are grounding, growing, and activation. In the grounding step, sensory information is inserted in BoP as perception information and is matched against the features of ontology instances in BoP. If the inserted sensory information does not match, all directly linked neighbor nodes and their links in the BoP are copied from wASN and the activation step is repeated. The copying of neighbor nodes here has its basis in the locality principle [17], which states that related objects are located close together or are influenced only by their surroundings. If the information fails to match again, it is marked as a false positive . After the growth step, a *dummy* node is inserted and connected to a sensory input node and the last winner node of the spreading activation on BoP. In the activation step of the improvisatory process phase, the spreading activation algorithm sums the weights of nodes associated with the input and goal nodes of BoP. Thus, a winner node is selected based on the winner-take-all (WTA) activation strategy. Additionally, the action associated with the winner node is selected as an alternative action. The two winner nodes are recursively entered into BoP as sub-goals for the next procedure to enable a robot to select more goal-oriented tasks. The above steps are repeated until the goal is achieved or the number of iterations reaches a threshold. In the knowledge acquisition phase, if the goal is achieved, the weights of all edges linked to input nodes in the wASN are increased by a factor of X, where X depends on the discount reward and success rate. If the goal is not achieved, the weights of all edges in the wASN are reduced by a factor of Y, where Y depends on the discount reward and failure rate.

III. Robot Knowledge Representation On Weighted Action-Coupled Semantic Network

Several researchers represent robot knowledge using an ontology language such as description logic [18], [19] or rule language [20]. Using rule language, however, it is difficult to design rules that imply trying alternate strategies if the initial attempt yields an unsatisfactory result. In conventional robotic systems, rules are derived from robot data or a robot-centered ontology [16]. However, this task requires skilled knowledge modeling experts, and complications often arise if the knowledge in the domain-specific rules for qualification is shared or reused. Thus, providing goal-associated context, object, and space (COS)-action complexes is necessary to enable a robot to adapt itself to different environments and dynamic situations. All concepts of the semantic network for robot intelligence are associated with actions for epistemic effects and ontic effects, similar to sensory-motor coordination (SMC) [21] and an object action complex (OAC) [22], [23]. In this paper, the proposed weighted action-coupled semantic network (wASN) recommends improvisational goal-oriented actions, thus improving the ability of a service robot to achieve the required goals.

A. Action-coupled Semantic Network

Robot knowledge in the form of perceptions, objects, spaces, contexts (COS), and actions is represented by points in high-dimensional space. Specifically, action classes are associated with the other knowledge classes as ontic or epistemic action. A semantic network is a rich logical structure that can represent multi-level knowledge. Each COS class is comprised of three knowledge levels that represent the robot world model. The low level contains object parts, metric maps, spatial context etc., which are used for matching against perceptual features. The middle level represents objects, topological maps, temporal context etc. with their name and functionality. The higher level consists of situations, compound objects, and semantic maps etc., which are at an abstract level in order to describe relationships with other knowledge classes.

For example, the action KClass has three knowledge levels: A_1, A_2 and A_3. The action class includes ontic actions, which change physical environments, and/or epistemic actions, which recognize environments; the environment can be changed by the robot itself, and by other agents including humans. A_1 the behavior level and includes the robot's atomic functions, such as *goto*, *turn*, and *extractSIFT*. A_2 is the task level, where a task is described by combinations of actions. A task can be defined as a short-term sequence of actions including *PerceptualAction*, *ObjectAction*, *SpatialAction*, and *ContextAction*. A_3 is the service level and it describes long-term goals. The action class has a special property called *hasAction* that represents action-associated relations. For example, spatial contexts such as *left* and *right* are associated with the epistemic action *lookAround*. The following is a description logic representation of action-coupled properties within the perception class and all COS classes.

$$
\begin{aligned}
PerceptualAction &:= Action \\
&\wedge\ \forall hasPerceptualTarget.Feature \\
&\wedge\ \exists hasAction.extractFeature. \\
ObjectAction &:= Action \\
&\wedge\ \forall hasObjectTarget.Object \\
&\wedge\ \exists hasAction.recognizeObject. \\
SpatialAction &:= Action \\
&\wedge\ \forall hasSpatialTarget.Space \\
&\wedge\ \exists hasAction.localize \\
&\wedge\ \exists hasAction.gotoSpace. \\
ContextAction &:= Action \\
&\wedge\ \forall hasContextTarget.Context \\
&\wedge\ \exists hasAction.lookAround.
\end{aligned}
$$

B. Weighted Ontology and Dummy Node

Semantic queries are answered with one of three types of results: true, false, and unknown. Thus, it is difficult to represent the higher importance of certain properties for specific classes. One approach is to use weights to represent the relations between classes and their importance. We propose the weighted Action-coupled Semantic Network (wASN) as an extension of OMRKF to infer alternative actions associated with context, object, and space (COS). Similar to the semantic network for a robot, wASN is a graph consisting of nodes and edges that represent concepts such as perception, object, space, context, and the semantic relations between concepts. These concepts are all coupled with actions. However, wASN assigns weighted values to connections between semantic nodes, similar to neural networks. The formal representation of a tuple description for wASN is as follows:

$$
wASNtuple := \{predicate(subject, object, weight)\}. \quad (1)
$$

For the sake of comparison, RDF Triple [24] is given as

$$
RDFtuple := \{predicate(subject, object)\}. \quad (2)
$$

Based on (1) and (2), the wASN description tuple is considered to be an extended version of the RDF triple because it incorporates weight as an additional property, making it possible to infer goal-oriented improvisatory attempts without having complete knowledge and/or domain-specific rules.

$$
\exists hasAction.\top \sqsubseteq \sigma(Perception) \sqcap \sigma(Context) \sqcap \sigma(Object) \sqcap \sigma(Space). \quad (3)
$$

$$
\top \sqsubseteq \forall hasAction.\sigma(Action). \quad (4)
$$

All classes of type context, object, and space have the *hasAction* property to represent an action-associated relationship whose range is the Action class, as represented in 3 and 4.

C. Board of Processing

While wASN serves as a reference model that stores schematic knowledge, instances, relationships, and their weights, the board of processing (BoP) serves as the procedural model or working memory, characterized by the involvement of executive and attentional aspects during interim processing, disposal, and retrieval of knowledge for the current goals. All information represented in the BoP by nodes, edges, and their weights is a subset of the information in wASN. This includes goals, inputs, and

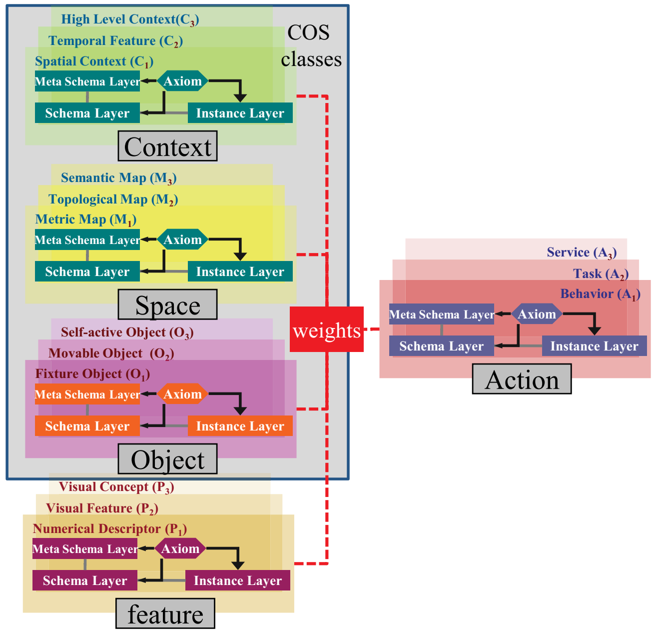

Fig. 2. A model of the weight Action-coupled Semantic Network. All robot knowledge classes are weight-coupled with actions.

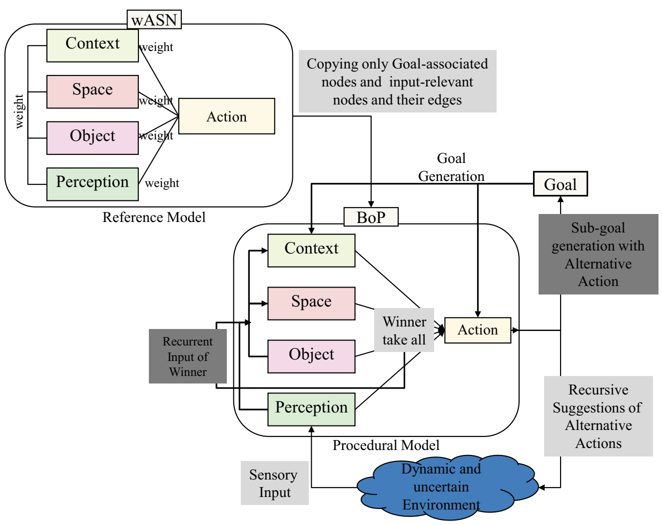

Fig. 3. The relationship between the weighted Action-coupled Semantic Network and the Board of Processing.

their neighbors which are copied from wASN to BoP. Fig. 3 depicts the relationship between wASN and BoP . The spreading activation algorithm is employed in the proposed method to complete the service task as it does not require all nodes in the wASN reference model to participate in the BoP procedure, only the perceived features and their neighbors. As sensory information is generated by perception modules, which include a visual recognizer and localizer, perception nodes are inserted into the perception class of BoP and matched with the objects in BoP. When winner nodes in both the COS class and action class are found using the spreading activation algorithm over BoP, a subgoal is generated and inserted into BoP.

False positives are handled based on weighted ontology. As false positives usually occur randomly, they may not be matched in the current BoP but may instead be matched in the wASN knowledge base. *Dummy* nodes are inserted to link mismatched sensory input and the BoP. If there is a mismatch, it may be a false positive. Thus, a *dummy* node is added and connected to the sensory input node and the previous winner node from the spreading activation algorithm in BoP. The *dummy* node is associated with an epistemic action, like recognize or localize, through the *hasAction* property. The formal representation of a *dummy* node is:

$$\begin{aligned} Dummy \quad &:= \quad Object \\ &\wedge \quad \exists hasAction.recognizeObject. \end{aligned} \tag{5}$$

$$\begin{aligned} &Dummy \quad [domain\{0:1\}* => WinnerNode]. \\ &Dummy \quad [range\{0:1\}* => SensoryInput]. \end{aligned} \tag{6}$$

The *dummy* node makes it possible to suggest epistemic actions if there is a recognition mismatch. These matching failures come from observational uncertainty from false positives in recognition results or external events, which are a result of the ontic actions of other agents.

IV. Root Knowledge Inference Using Spreading Activation

The proposed processing framework comprises of three phases: the initial deliberation phase, the improvisatory process phase, and the knowledge acquisition phase. In the initialization phase, if a goal is in the form of a target and an associated action, the board of processing (BoP) is initialized such that the goal node and its neighbor nodes are copied from the weighted action-coupled semantic network database (wASN) to BoP.

A. Initial Deliberation Phase

When a goal and initial observations are given, the initial deliberation phase is the first phase of the robot knowledge inference mechanism. The goal consists of a target node in a context, object, and space (COS) class and its associated action node in the action class. The initial observation consists of an input node and its associated action node. First, the board of processing (BoP) is emptied and two goal nodes are inserted. Next, subgoal nodes are found as linked nodes from the entire wASN knowledge base by executing the spreading activation algorithm deliberatively; their neighbor nodes, the edges representing the relationships between goal nodes and their neighbors, and the weight of each edge are copied from wASN to BoP. For an action class, the action goal node and its sub-nodes are copied. The action sub-nodes, which are primitive behaviors performed by a robot, can be output as alternative actions by the inference mechanism.

B. Improvisatory Process Phase

As some features are perceived through perception modules, the improvisatory process phase consists of three steps. In the first step, the similarity between the perceived features and individual model features in BoP and wASN is evaluated. Similarity matching is preferentially performed first with the nodes in BoP, which includes up-to-date nodes due to temporal locality; temporal locality refers to the reuse of specific information within a short time. If matching fails in BoP, matching candidates are selected from the COS of wASN. These nodes in wASN are simply the neighbor nodes of the corresponding COS nodes in BoP due to spatial locality; this refers to the high rate of information reuse located in close proximity. Spatial locality denotes strong relationships between nodes, which is reflected in a high weight value. In addition, the candidate nodes, their edges, and weights are copied from wASN to BoP in the *growth step*. Similarity is evaluated through a similarity matching function (SIM) as the proportion of matching values between input features and model features. The SIM value is calculated as follows:

$$SIM(F_i, F_m) = \frac{2 \times N_m}{N_i + N_m,} \tag{7}$$

where

$$\begin{aligned} F_m \quad \in \quad & \{F_n | \forall F.\{COS \cap BoP\} \\ & \cap \forall F.\{COS \cap \forall Rel.(wASN, BoP)\}\}. \end{aligned} \tag{8}$$

Here, N_i is the number of input feature elements F_i, and N_m is the number of matched elements between the input feature F_i and model feature F_m. The SIM value is used as the weight of the edge connecting the input feature node and the matched COS node.

In the *growth step*, , the feature node of the perception class and the matched node of the COS class are inserted into BoP. The edge between the feature node and the COS node indicates the property *hasFeature*, and its weight is assigned by SIM. If the inserted sensory information fails to match, all immediate neighbor nodes and the respective edges are copied over from wASN and the activation step is repeated. If matching still fails, a *dummy* node is inserted and linked to the sensory input node and the previous winner node from the activation step in BoP. As the *dummy* node is associated with an epistemic action, it is possible to suggest the epistemic action as an alternative action. Performing the additional epistemic action as an alternative action can eliminate most false positives and update the world model to reflect the most recent information.

As a final step of the improvisatory process phase, the spreading activation algorithm is executed as the *activation step*. The spreading activation algorithm evaluates all the activation values of the nodes and assigns the full activation value to goal and subgoal nodes and input nodes in COS classes. The activation step consists of a number of activation weaves from one node to all other nodes connected to it. The activation values are computed using the following formula:

$$A_i = \sum_j w_{ij} O_j, \tag{9}$$

where A_i is the activation value of node i, w_{ij} is a weight associated with the edge between nodes i to j, and, O_k is the output of node j connected to node i.

AA winner node is selected based on the winner-take-all (WTA) activation strategy in COS classes.

$$winner_{COS} = \arg_{A_i} \max\{A_i \in \{COS \cap BoP\}\}. \tag{10}$$

In addition, after assigning the full activation value to goal and subgoal nodes, and associating the action node with the winner node of COS, the spreading activation algorithm is executed again in the Action class. Consequently, one more winner node is selected by the WTA strategy from the action class.

$$winner_{Action} = \arg_{A_i} \max\{A_i \in \{Action \cap BoP\}\}. \tag{11}$$

The two winner nodes are the output of the inference mechanism providing an alternative action. In the next step, the winner nodes are recursively inserted into BoP as subgoals so that a robot selects more goal-oriented tasks. This improvisatory process phase is repeated until the goal is achieved or the number of iterations reaches a threshold.

C. Knowledge Acquisition Phase

After the improvisatory process phase, service tasks are completed more often than before due to the availability of alternative actions in incomplete and uncertain environments. Depending on the success of service tasks, the weighted action-coupled semantic network (wASN) is updated in the weight update step; this process is analogous to the function of long-term memory of humans. If the goal is achieved, all weight values of edges linked to input nodes in wASN are increased based on the weight update rule. Otherwise, all weight values of edges in the wASN are reduced. The weight update rule revises weight w_{ij} associated with input A_j according to the rule:

$$w_{ij} \leftarrow w_{ij} + \eta(t - w_{ij} A_j). \tag{12}$$

Here, t is the target output for the training. If the goal is achieved, the target output is set to 1; otherwise, it is set to 0. η is a positive constant denoting the learning rate. The learning rate controls how much the weights are changed in the weight update step.

V. Experiment

The proposed graph model and processing mechanism were evaluated and verified through a scenario-based experiment using a Pioneer 3 AT robot to find an object that was moved by another agent to a new position from the previous position recorded by the robot. Nevertheless, if the robot looks around for another object that was close to the target

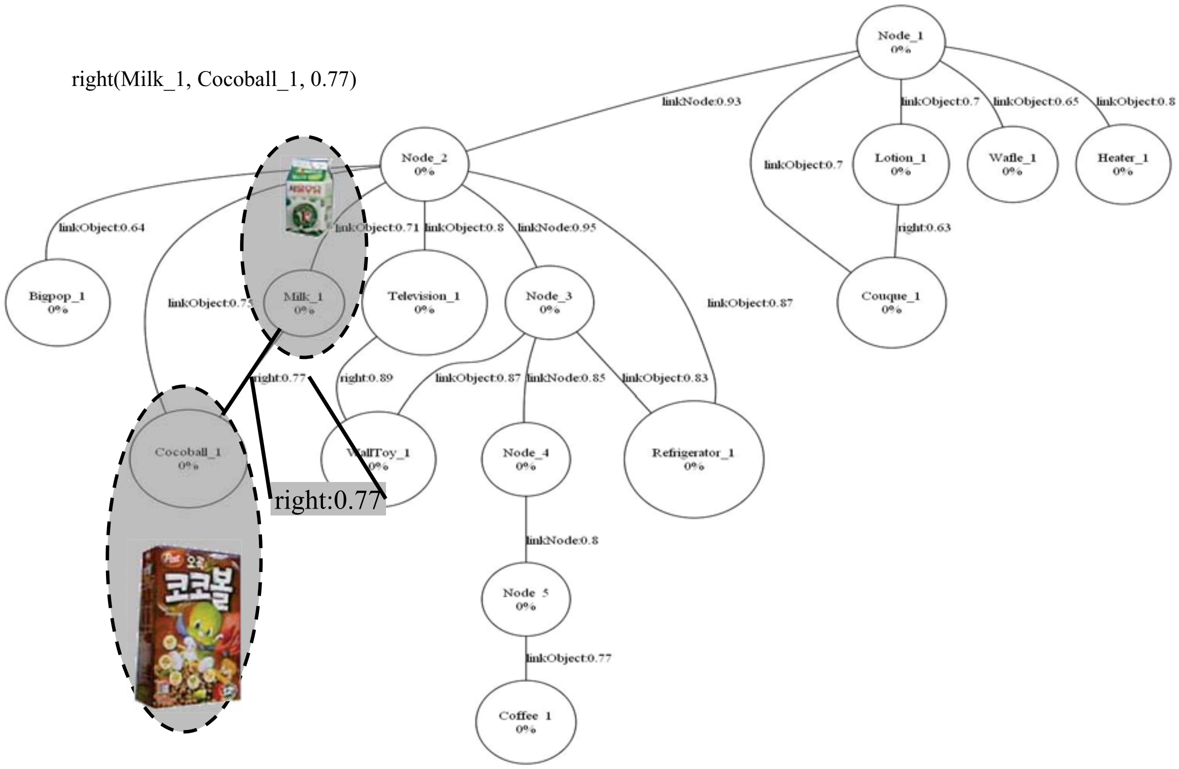

Fig. 6. An example of a weighted action-coupled semantic network (wASN).

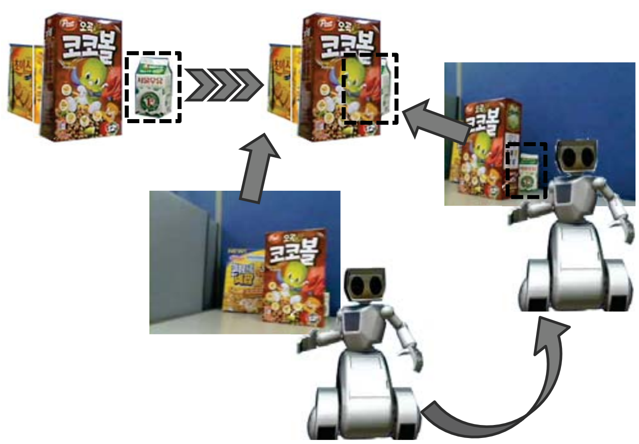

Fig. 4. Changed environments: The milk is behind the Cocoball cereal box. If the robot moves to the right side of the table, it can find the milk.

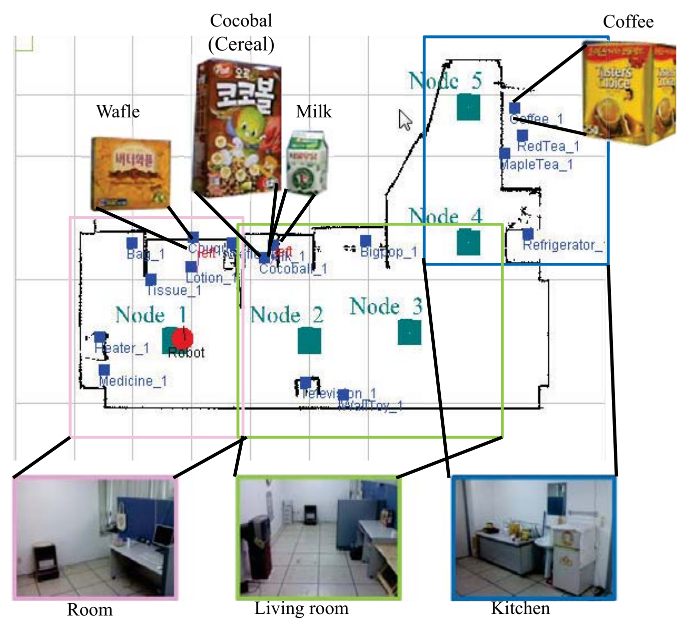

Fig. 5. One room, one living room, and one kitchen constitute the experimental environment. The milk is located to the *right* of the Cocoball cereal box *on* the dining table in the living room; a waffle is *on* the table in the room; coffee is located in the kitchen; and a robot is located at *Node_1* in the room.

object, it may be able to find the target, as shown in Fig. 4. The milk is to the *right* of the Cocoball cereal box, and both the milk and the cereal box are on the table in the living room. The goal for the robot is to find the milk, which was located to the *right* of the cereal box on the dining table in the living room. However, the milk was moved behind the cereal box and hidden when the robot was not watching the dining table, as shown in Fig. 5. If the robot looks behind the cereal box, then it will find the milk. Fig. 6 shows the COS instances of wASN for the experimental environment. *Milk_1* is located to the *right* of *Cocoball_1* and its weight is 0.77.

When a robot located at *Node_1* in the room is requested to find *Milk_1*, the first BoP is initialized. The goal node *Milk_1*

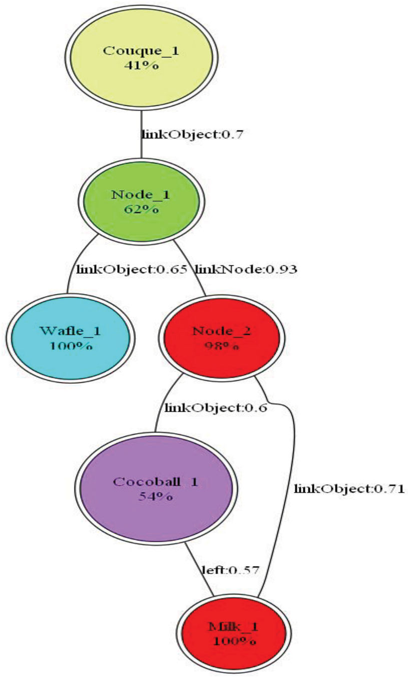

Fig. 7. An example of BoP after activation step in improvisatory process phase.

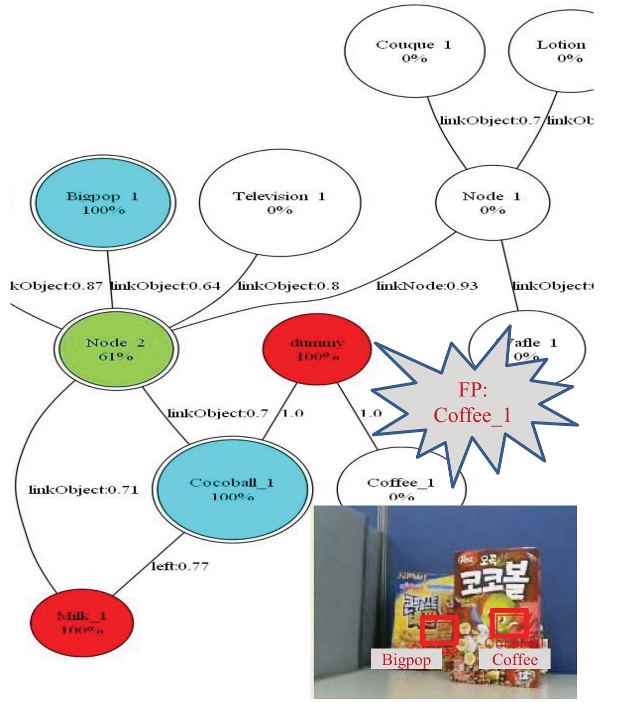

Fig. 8. A *dummy* node is generated because of false positive observation: *Coffee_1*.

is inserted into the BoP, and the nearest neighbor nodes, edges, and their weights are copied over from the wASN. As the robot is located at *Node_1*, that input node is also inserted into the BoP. In the *growth step* of the improvisatory process phase, additional nodes such as *Waffle_1* and *Node_2* as neighbors of *Node_1* and *Cocoball_1* and *Node_2* as neighbors of *Milk_1* are inserted into the BoP. An example of the *activation step* is shown in Fig. 7 where *Waffle_1*, located near *Node_1* in the room, is recognized, and inserted into the BoP. *Node_2* is selected as the winner node in COS classes and its associated ontic action goto is the winner node in the action class. Therefore, *goto (Node_2)* is the suggested subgoal from this procedure.

A. Detection of False Positive Observation

On the way to *Node_2*,an uncertain observation of *Coffee_1* gives a false positive. From Fig. *Coffee_1* is known to be located in the kitchen. The number of growth step iterations reaches the threshold, which is 3 in this case. Hence, a dummy node is added and connected to *Cocoball_1*, which is the winner node of the previous procedure, as shown in Fig. 8. The action *recognizeObject* is selected as an alternative action because it is an epistemic action of the object class. At the end of the next procedure, there are no matching results for coffee in the perception results, implying that coffee is a false object recognition result. If *Coffee_1* was moved by another agent, the observation is then true. As before, *recognizeObject* is selected as an alternative action, but *Coffee_1* is recognized again in the next step. Thus, the *dummy* node is merged with *Coffee_1* and the link is inferred as a spatial context whose domain and range properties are in

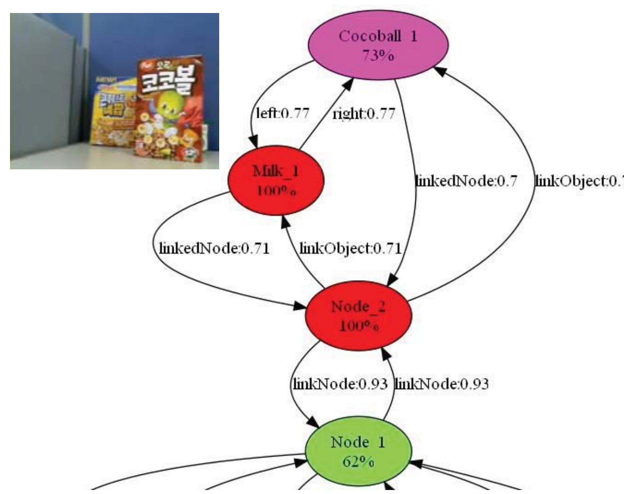

Fig. 9. The alternative action provides a chance to find *Milk_1*.

the object class, and the two objects are recognized to be in the same time and space. Finally, the weights of the inferred spatial context, *right*, are copied over from wASN. Although the robot's environment has changed, wASN is not updated immediately. It will be updated in the knowledge acquisition phase after service completion.

B. Handling Incompleteness of External Events

When the robot arrived at *Node_2*, there were no matches for the milk. There is a spatial relationship between the goal node *Milk_1* and the input node *Cocoball_1*. The spatial relationships *left* and *right* are associated with an epistemic action *lookAround*. Thus, *lookAround (Cocoball_1)* is suggested as an alternative action, as shown in Fig. 9. Consequently, *Milk_1* is found as shown in Fig. 4.

VI. Conclusion

The novel weighted action-associated Semantic Network (wASN) provides improvisational recommendations of goal-oriented actions even when the knowledge base is incomplete due to false negatives and/or false positives. Action-coupled semantic knowledge and the spreading activation algorithm increase the ability of service robots to complete service tasks. For an incomplete knowledge base that is based on domain-specific rules, uncertain observations or external events are detected by identifying unexpected observations that do not appear in the BoP (board of processing) or among the neighbors. Such uncertain observations are handled by adding dummy nodes that are associated with epistemic actions to update knowledge by making new observations. The proposed mechanism was successfully applied and tested in a service task which involved finding objects that were moved by other agents or were incorrectly recorded. Furthermore, we aim to study the scalability of this method in terms of handling several tasks simultaneously. This should allow service robots to perform service tasks for humans in real environments based on the proposed mechanism.

References

[1] L. Morgenstern, "Knowledge preconditions for actions and plans," in *Proceedings of the 10th international joint conference on Artificial intelligence*, 1987, pp. 867–874.

[2] J. Blythe, "Planning with external events," in *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, 1994, pp. 94–101.

[3] O. Etzioni, S. Hanks, D. Weld, D. Draper, N. Lesh, and M. Williamson, "An approach to planning with incomplete information," in *Proc. 3rd Int. Conf. on Principles of Knowledge Representation and Reasoning*. Citeseer, 1992, pp. 115–125.

[4] A. Herzig, J. Lang, and P. Marquis, "Action representation and partially observable planning using epistemic logic," in *Internal Joint Conference on Artificial Intelligence*, vol. 18. Citeseer, 2003, pp. 1067–1072.

[5] L. Kaelbling, M. Littman, and A. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[6] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa, "Online planning algorithms for pomdps," *Journal of Artificial Intelligence Research*, vol. 32, no. 1, pp. 663–704, 2008.

[7] C. Baker, R. Saxe, and J. Tenenbaum, "Action understanding as inverse planning," *Cognition*, vol. 113, no. 3, pp. 329–349, 2009.

[8] A. Collins and E. Loftus, "A spreading-activation theory of semantic processing," *Psychological review*, vol. 82, no. 6, pp. 407–428, 1975.

[9] J. Anderson, "A spreading activation theory of memory," *Journal of verbal learning and verbal behavior*, vol. 22, no. 3, pp. 261–295, 1983.

[10] A. Katifori, C. Vassilakis, and A. Dix, "Ontologies and the brain: Using spreading activation through ontologies to support personal interaction," *Cognitive Systems Research*, vol. 11, no. 1, pp. 25–41, 2010.

[11] P. Maes, "The agent network architecture (ANA)," *ACM SIGART Bulletin*, vol. 2, no. 4, pp. 115–120, 1991.

[12] G. H. Lim and I. H. Suh, "Weighted action-coupled semantic network (wasn) for robot intelligence," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008, pp. 2035–2040.

[13] G. H. Lim and I. H. Suh,, "Robust robot knowledge instantiation for intelligent service robots," *Intelligent Service Robotics*, vol. 3, pp. 115–123, 2010, 10.1007/s11370-010-0063-6. [Online]. Available: http://dx.doi.org/10.1007/s11370-010-0063-6

[14] D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to ip traceback," *ACM Trans. Inf. Syst. Secur.*, vol. 5, pp. 119–137, May 2002. [Online]. Available: http://doi.acm.org/10.1145/505586.505588

[15] G. H. Lim, I. H. Suh, and H. Suh, "Ontology-based unified robot knowledge for service robots in indoor environments," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 41, no. 3, pp. 492 –509, may 2011.

[16] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007*, 2007, pp. 429–436.

[17] I. Abraham, A. Badola, D. Bickson, D. Malkhi, S. Maloo, and S. Ron, "Practical locality-awareness for large scale information sharing," *Peer-to-Peer Systems IV*, pp. 173–181.

[18] O. Corcho and A. Gómez-Pérez, "A roadmap to ontology specification languages," *Knowledge Engineering and Knowledge Management Methods, Models, and Tools*, pp. 80–96.

[19] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, *The description logic handbook*. Cambridge University Press New York, NY, USA, 2007.

[20] K. Eshghi, "Abductive planning with event calculus," in *Logic Programming: Proceedings of the Fifth International Conference and Symposium*. MIT Press, 1988, p. 562.

[21] R. Pfeifer and C. Scheier, *Understanding intelligence*. The MIT Press, 2001.

[22] F. Wörgötter, A. Agostini, N. Krüger, N. Shylo, and B. Porr, "Cognitive agents - a procedural perspective relying on the predictability of object-action-complexes (oacs)," *Robot. Auton. Syst.*, vol. 57, pp. 420–432, April 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1517866.1518269

[23] 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 *IEEE RAS International Conference on Humanoid Robots*, 2006.

[24] O. Lassila and R. Swick, "Resource description framework (rdf) model and syntax," *World Wide Web Consortium, http://www. w3. org/TR/WD-rdf-syntax*.