Ontology Modeling and Storage System for Robot Context Understanding

Eric Wang[1], Yong Se Kim[1], Hak Soo Kim[2], Jin Hyun Son[2],
Sanghoon Lee[3], and Il Hong Suh[3]

[1] Creative Design and Intelligent Tutoring Systems Research Center
Sungkyunkwan University, Korea
wang@me.skku.ac.kr, yskim@skku.edu
[2] Department of Computer Science and Engineering, Hanyang University, Korea
{hagsoo,jhs}on@cse.hanyang.ac.kr
[3] Graduate School of Information and Communications, Hanyang University, Korea
shlee@incorl.hanyang.ac.kr, ihsuh@hanyang.ac.kr

Abstract. A mobile robot that interacts with its environment needs a machine-understandable representation of objects and their usages. We present an ontology of objects, with generic shape representations obtained through form-function reasoning. Sets of objects are associated with typical human activities, which supports context understanding. We describe an efficient ontology document storage system, which is based on stable and well-known relational databases. We first design a relational data schema appropriate for Web Ontology Language (OWL) documents, and then develop a transformation mechanism from OWL documents to the relational schema.

1 Object Recognition and Scene Interpretation

To support a robot's interaction with a typical human environment requires a machine-understandable representation of objects, including their shapes, functions, and usages. Object recognition is supported by reasoning from object shape information, while scene understanding is supported by reasoning about sets of objects.

Previous research has explored the relationship between form and function for object recognition. However, these have not directly addressed an ontological representation of objects. The Generic Recognition Using Form and Function (GRUFF) system [1] represents objects as a set of functional elements (mostly planar surfaces), and spatial relations between elements. It performs generic object recognition by matching functional surfaces in the sensor input data to objects' definitions, using customized geometric algorithms. GRUFF organizes object classes into an *is-a* hierarchy, but does not use a standard ontology representation.

Neumann *et al.* [2] performs context-based scene interpretation by modeling scenes as *aggregates*, where an aggregate is a set of entities and their spatial and temporal relations. They represent aggregates of scenes in description logic (DL), and match input models to scene definitions using the RACER DL reasoner [3]. While they do not directly use an ontology representation, their DL representation is essentially equivalent to using the standard OWL ontology language. However, their scene

R. Khosla et al. (Eds.): KES 2005, LNAI 3683, pp. 922–929, 2005.

interpretation capability is beyond the current state-of-the-art in description logics, because a complete representation of the relations between entities exceeds the allowed expressiveness of RACER's DL.

2 Object Ontology

We adopt the ontology formalism in developing a generic ontology of objects. We use the standard OWL web ontology language, and the de facto standard Protégé ontology editor with OWL plugin [4]. Using this ontology, we have instantiated a knowledge base of ~300 objects for a typical indoor environment.

Representation of objects. Manufactured objects are typically assembled from multiple components, where each component contributes some specific functionality. Reflecting this, we adopt a hierarchical feature-based representation, shown in Fig. 1. An object is decomposed into a set of features and their spatial relationships, where a *feature* is a functionally significant subset of an object or another feature. Features are characterized by the functions they provide.

Each feature can be further decomposed into more features.

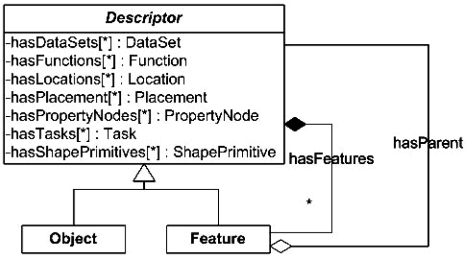

Fig. 1. Part-whole modeling of Object and Feature classes

Spatial Relations. We define several spatial relations that frequently occur in everyday objects. For each spatial relation, we provide a definition that can be implemented as a (geometric) algorithm. For example, the ***above****(A, B)* relation is defined as: A is above B iff A's highest point is higher than B's highest point (with respect to the gravity direction), and A's lowest point is not lower than B's highest point.

Form-Function Reasoning. We characterize features using generic functions taken from function-based taxonomies for design [5][6]. While a feature is a 3D component, its functional elements, or *organs* [7], may correspond to subsets of its 3D shape. By applying form-function reasoning, we deduce geometric shape requirements for each functional element.

For example, a table's primary function is to limit the downward motion of many objects of any shape. The key feature for a table is a counter, which is typically a thin, rigid 3D slab. A counter's key organ is its top surface. To contact many objects implies many contact points, from which we deduce a planar surface. A table should also minimize the energy required to translate objects to different positions, which implies a horizontal orientation. Hence, we deduce a shape requirement of a *horizontal planar surface* for a counter's top surface.

Geometric Shape Elements. We define a qualitative representation of geometric shape elements, shown in Fig. 2. A shape element has a geometric datum (usually a surface), which represents a generalized portion of a solid's boundary. Other constraints on the allowable orientation, curvature, and tolerance of a shape element are specified using a phrase structure.

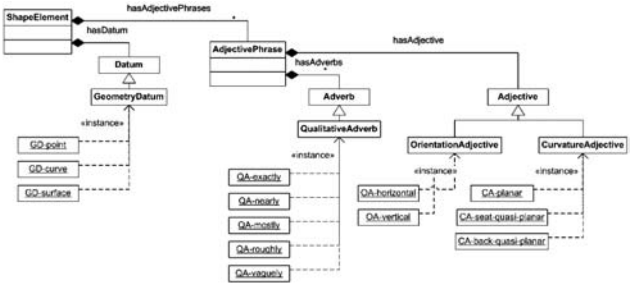

Fig. 2. Representation of geometric shape elements

Representation of Solids. A boundary representation (B-rep) is a 3D model that rigorously describes a solid by enumerating the topological elements of its boundary, including its faces, edges, and vertices. Other solid representations can be converted to B-rep, so a B-rep is a good candidate to be a generic solid representation.

On the other hand, an ontology of objects should also support generic representations of object families. This requires a capability to tolerate wide variations in specific geometry, while capturing the critical geometric relations only.

We adopt a *partial B-rep* scheme, in which a subset of a solid's boundary is fully specified, representing the critical geometric and topological relations only. Remaining portions of the boundary are abstracted away. Each solid has a bounding box data field, reflecting the principle that all real solid objects are bounded. Each feature's shape information is then represented as a partial B-rep with 1 or more geometric shape elements.

3 Use of Ontology for Context Understanding

The object ontology can be used to support object classification. Given a scene with one or more objects:

- Decompose the scene into a set of geometric shape elements, and compute all spatial relations between these shape elements.
- By comparing each shape element in the scene to each feature's required shape elements, and other data such as bounding boxes, classify each shape element into a set of candidate features.
- For each object in the object ontology, check if all of its required features exist, and whether all spatial relations between its features are satisfied. This groups a set of features and spatial relations into a new instance of that object class.
- Repeat using only the unassigned shape elements in the scene data, until all input elements have been assigned to some object.

For scene understanding, we take the perspective that a human activity is characterized by a set of objects that are typically used during that activity. We instantiate associations from sets of objects to activities. Then given a scene of multiple objects, the corresponding activities can be deduced.

4 Ontology Document Management

Nowadays, many researchers have a high interest in the context understanding services, and in reality their concepts have been applied to various application areas, especially such as home networking, telematics, and intelligent robotics. Because the concept of context-awareness is basically considered to be supported by the semantic web technologies, most applications which want to provide context-understanding services may use or adopt semantic web-related international standards: OWL as a web ontology language and Web Services for interaction between software modules [8]. For the efficient support of context understanding services, first of all, the way to store and manage ontology documents should be discussed [9]. In this section, we propose a new data schema based on the relational database and develop a transformation mechanism from OWL ontology documents to the relational schema.

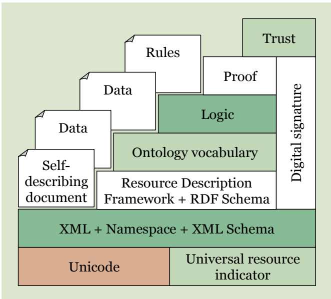

Fig. 3. Semantic Web Stack

The XML-based semantic web specification by W3C is composed of five stack layers: resource description framework, ontology vocabulary, logic, proof, and trust layer as in Fig. 3. Here, only resource description framework and ontology vocabulary layers are officially standardized as RDF (RDFS) and OWL, respectively. With RDF and RDFS, we can not only describe web resources with simple statements, but also define classes and properties that may be used to describe other classes and properties as well as web resources. On the other hand, OWL, a revision of the previous DAML+OIL, provides more facilities for expressing meaning and semantics by extending RDF and RDFS. Within these semantic languages, a web resource is represented by a simple statement of a triple data structure (subject, predicate, object). Up to date, several semantic query languages are proposed, for example, such as RQL and RDQL for querying RDF and RDFS documents, DQL for DAML+OIL documents, and OWL-QL for OWL documents. Now, RDQL and OWL-QL are considered as de-facto standards.

As some outstanding toolkits related with the semantic web ontology storage system, Jena [10] and Sesame [11] can be considered to compare with our proposed system. Jena, developed by HP, is a Java-based semantic web framework in which users can easily build semantic web-enabled applications. Using Jena, we can store XML-based semantic web documents including RDF, RDFS, DAML+OIL and OWL, and query the stored documents with RDQL query language. On the other hand, Sesame by NLNet Foundation is basically a storage system for RDF(S) documents, which supports RQL and RDQL to query the stored documents. Being extended by BOR [12], Sesame+BOR can support DAML+OIL documents as well as RDF(S) documents.

Our main contribution in the ontology storage system is to propose an appropriate data schema for ontology documents and develop an ontology document management system in which semantic queries can be processed within the reasonable time even using small amount of data stored.

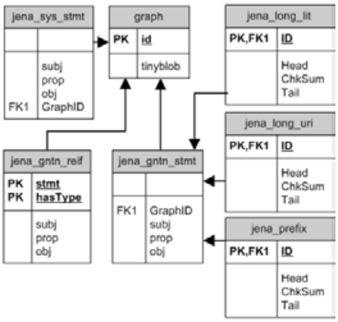

Fig. 4. Jena's Data Schema

5 Proposed Relational Data Schema

Our system's generic goal is to propose a relational schema in which OWL semantic documents can be effectively stored and then efficiently retrieved when querying the stored documents with RDQL. Because DAML+OIL and OWL are almost identical semantic document format, Jena and Sesame+BOR systems which all support DAML+OIL document format and RDQL semantic query language can be compared with our proposed storage system.

In principle, Jena is designed for each document-based storage system. In other words, whenever we insert a semantic document to Jena, two tables, jena_g*n*t*n*_reif and jena_g*n*t*n*_stmt (Here, *n* is any number), dependent on the document are newly generated while other basic common tables are shared as in Fig. 5. In case of the query processing in Jena, we can query only for each document contents while Jena may load into the memory model. On the other hand, Sesame+BOR provides the common relational schema which would be shared for all inserted documents as in Fig. 5. Hence, the query can be targeted to the all data stored in Sesame+BOR system. But, note that Sesame+BOR's table will generally contain too much data because it may generate much more additional information by inferring from the original inserted document. In this regard, we think that our proposed storage system should be like Sesame+BOR but contain as less data as possible without loss of information compared to the original documents. Because of that, we assure that our proposed system be appropriate for embedded systems with the limited resources such as intelligent robotics.

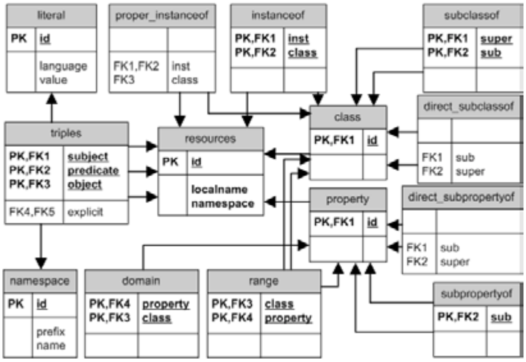

Fig. 5. Sesame+BOR's Data Schema

Because OWL is extended upon RDF and RDFS, its constructs composing of a document are inherited from the RDF(S)'s six core constructs, i.e., rdfs:Class, rdfs:Resource, rdfs:subClassof, rdf:Property, rdf:domain, rdf:range, and rdf:subPropertyof. We design and implement, therefore, our storage system with the relational data schema based on these six core constructs as in Fig.6.

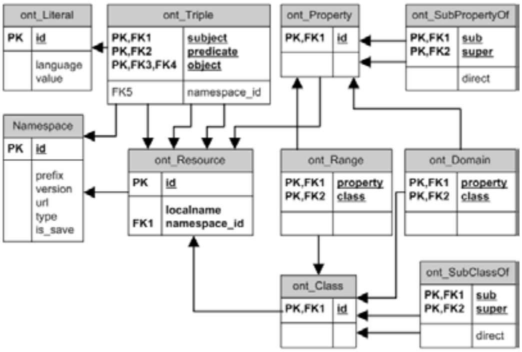

Fig. 6. Our Proposed Data Schema

6 Experiments

To show our system's validness, we compare our system with Sesame+BOR into two different aspects: One is to find the number of records inserted, and the other is to evaluate the average query processing time for the five different semantic queries written by RDQL which are so general in the semantic information processing. As shown in Fig. 7, the number of records generated by Sesame+BOR is as many as twice to our proposed system. If we experiment it with much more big and complex semantic documents, we can find out that our system is more useful for the embedded applications.

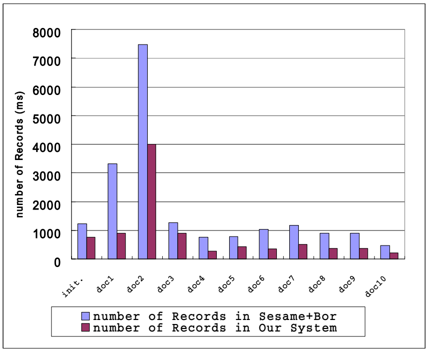

Fig. 7. Comparison in the aspect of the number of records stored

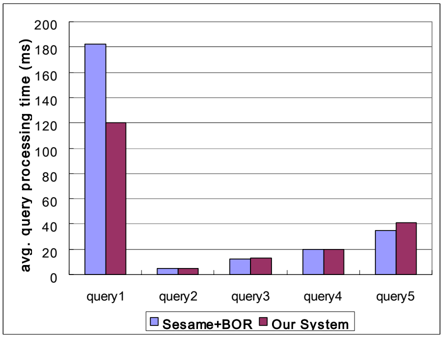

Fig. 8. Comparison in the aspect of the Average Query Processing Time

To compare the two systems in the aspect of the efficiency in RDQL semantic query processing, we have measured the average query processing time for the five different queries as in Fig 8. The five semantic RDQL queries are:

– Query 1: Retrieve all subjects satisfying the predicate and object requirements
– Query 2: Query for the immediate descendant relationship
– Query 3: Query for the descendant relationship
– Query 4: Query on the Range and Domain properties
– Query 5: Retrieve all instances for a certain class

We notice that our system can support the query efficiency similar to Sesame+BOR. As a result, our proposed storage system can provide reasonable query efficiency while keeping small amount of data.

7 Summary and Further Work

We have developed an ontology document storage system based on a relational schema that is appropriate for managing OWL documents in the embedded intelligent robot environment. Some experimental results are additionally presented to justify our proposed storage system.

As further work, we plan to study the way to efficiently process semantic queries, especially using new inference and indexing mechanisms.

Acknowledgement

This work is supported by the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Korea Ministry of Commerce, Industry and Energy.

References

1. L. Stark and K. Bowyer, “Function-based generic recognition for multiple object categories,” *Computer Vision, Graphics and Image Processing*, vol. 59, no. 1, pp. 1–21, Jan. 1994.
2. B. Neumann and R. Möller, “On scene interpretation with description logics,” *FBI-B-257/04 (Technical Report), Fachbereich Informatik*, Universität Hamburg, 2004.
3. V. Haarslev and R. Möller, “RACER system description,” *Proc. Int’l Joint Conf. on Automated Reasoning* (IJCAR 2001), LNAI vol. 2083, Springer, 2001, pp. 701–705.
4. Protégé, http://protege.stanford.edu.
5. C. F. Kirschman and G. M. Fadel, “Classifying functions for mechanical design,” *Journal of Mechanical Design*, vol. 120, pp. 475–482, 1998
6. R. B. Stone and K. L. Wood, “Development of a functional basis for design,” *Proc. ASME Conf. on Design Theory and Methodology*, Las Vegas, 1999.
7. J. Haudrum, *Creating the Basis for Process Selection in the Design Stage*, Ph.D. Thesis, Institute of Manufacturing Engineering, Technical University of Denmark, 1994.
8. T. Berners-Lee, J. Hendler, and O. Lassila, “The semantic web,” *Scientific American*, vol. 284, no. 5, pp. 34–43, 2001.
9. Z. X. Pan and J. Heflin, “DLDB: Extending relational databases to support semantic web queries,” *Workshop on Practical and Scalable Web Systems*, pp. 109–113, 2003.
10. J. Carrol and B. McBride, *The Jena Semantic Web Toolkit Public API*, HP Labs, Bristol, 2001, http://jena.sourceforge.net.
11. J. Broekstra, A. Kampman, and F. van Harmelen, “Sesame: A generic architecture for storing and query RDF and RDF schema,” Sesame, http://www.openrdf.org.
12. BOR, http://www.ontotext.com/bor/index.html.