論文 / 著書情報
Article / Book Information

論題	Mapping Landmarks in Structured Indoor Environments by using RDF data models and SPARQL engine based navigation
Title	Mapping Landmarks in Structured Indoor Environments by using RDF data models and SPARQL engine based navigation
著者	セロンロペス、アルトゥーロ エドゥアルド, 福島E. 文彦
Author	Arturo E. Ceron Lopez, Cesar A. Hernandez Reyes, Carlos W. Ponce Quiroga, Martha S. Lopez de la Fuente, Edwardo F. Fukushima
掲載誌/書名	, , ,
Journal/Book name	Proceedings of the 2015 JSME Conference on Robotics and Mechatronics, , ,
発行日 / Issue date	2015, 5
URL	http://www.jsme.or.jp/publish/transact/index.html
権利情報 / Copyright	本著作物の著作権は日本機械学会に帰属します。
Note	このファイルは著者（最終）版です。 This file is author (final) version.

Mapping Landmarks in Structured Indoor Environments by using RDF data models and SPARQL engine based navigation

○ Arturo E. CERON LOPEZ, Tokyo Institute of Technology, aceron@robotics.mes.titech.ac.jp
Cesar A. HERNANDEZ REYES, Universidad de Monterrey, cesar.hernandezr@udem.edu
Carlos W. PONCE QUIROGA, Universidad de Monterrey, carlos.ponce@udem.edu
Martha S. LOPEZ DE LA FUENTE, Universidad de Monterrey, martha.lopez@udem.edu
Edwardo F. FUKUSHIMA, Tokyo University of Technology, fukushimafmhk@stf.teu.ac.jp

This work proposes a method for mapping landmarks within structured indoor environments to enable robots to understand its surroundings and move through them. A new approach based on the RDF, which simplifies the implementation of the knowledge database system, is introduced and a navigation algorithm is derived from a SPARQL engine. The total navigation system was implemented, tested and validated using OpenRTM-aist, voice operated commands, and the V-REP simulator.

Key Words: navigation, landmark mapping, structured environment model, RDF

1 Introduction

Service robots have become more appealing as they are applied in various research fields for exploration tasks, especially in search and rescue operations. Independently from the environment, a robot needs a representation of the real world in order to understand its surroundings and move through it.

Nowadays, there are various frameworks that enable this kind of feature. However, the available frameworks are too extensive and lack from simplicity, making difficult on-the-fly implementation for practical tasks.

In the state-of-the-art, there are frameworks that enable a robot to find a target room inside a house by using the KnowRob system (based on the Resource Description Framework or RDF) [1], which stores and processes data about the objects found in the environment to resolve locations. Other frameworks let a robot to find out physical location relationships between objects that are meaningful for both humans and robots by using semantic map models represented by the Web Ontology Language (OWL) [2]. Moreover, a telepresence application requiring knowledge storage and management uses a three pillar model approach that includes prior knowledge, environment model and sensor information [3], where among other features, the existence of certain objects in the environment can be resolved/inferred. However the mentioned methods propose relatively complex solutions that make them cumbersome to implement unhesitatingly in comparison with bare RDF data models and a SPARQL (SPARQL Protocol and RDF Query Language) engine in conjunction.

1.1 Objective

In the search of an organized, yet simple way of mapping landmarks for navigation within a known environment, we propose a method for defining such information into files using a standard data model, where the files are loaded into a database that can be used to provide a robot the ability of understanding navigation commands given by a human user, this via an artificial intelligence algorithm based on database queries. The focus is on letting human users to learn about the principles of using knowledge databases in service robots and apply them on-the-fly in educational and practical tasks.

For this article, we used a structured indoor environment of rooms within a building floor. However this approach can be extended to outdoor environments.

The implementation includes mapping the environment into an RDF model, processing the data with a SPARQL engine, and using voice commands for driving the robot.

2 Mapping the environment with RDF

In order to keep the mapping method simple and practical, we decided to use the RDF for serializing the data (RDF/XML format). According to the World Web Consortium, "RDF is a language for representing information about resources in the World Wide Web" [4]. It represents metadata about resources, where resources are declared as Uniform Resource Identifiers (URI). URIs are not limited to represent web resources, and their use can be extended to represent other kind of entities as seen in previous approaches. For instance, we can use URIs to represent physical landmarks.

Thus, RDF can be used to build a complex, but easily portable database of relationships between landmarks. Relationships are defined as triples in the form of Subject-Predicate-Object (e.g. Landmark_A – has – Landmark_B).

For example, we can describe a floor inside a building. A floor contains rooms, and a room can contain chairs and desks. Additionally, every landmark has a Cartesian coordinate.

Once the RDF database is built, a SPARQL engine can directly use the information to make queries, useful to reason navigation commands given by a human user.

As a case study, the engineering laboratories (GENERA) at Universidad de Monterrey were modelled for our test implementation.

2.1 Building the environment model

Each floor in the building was modelled as a "Bag" of rooms by using the RDF Schema Bag class. Then, the "rdf:Description" and "rdf:about" tags were used to set the properties for each floor and each room (Fig. 1 and 2).

2.1.1 RDF Editor

For mapping the landmarks into a model, we developed a simple RDF editor using the Apache Jena API [5] for Java. The following steps are for defining a mapping:

a) Enter path of working directory.

Example of working directory file path:
C:\Jena_RDF

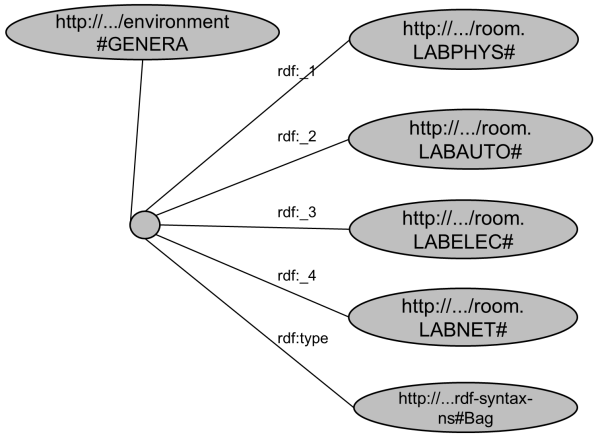

Fig.1 A floor modelled as a "bag" of rooms.

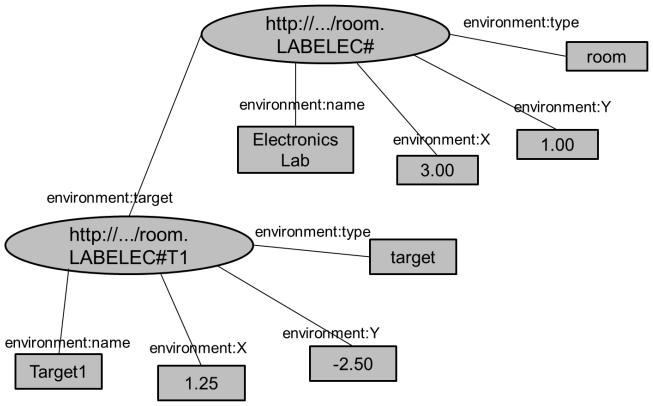

Fig.2 A room and its properties.

b) Define namespace URI and prefix.

Example of namespace URI:
http://demo.server.com/environment

c) Enter subject node (floor or room), and add it to the namespace.

Example of model subject (Added to namespace URI):
http://demo.server.com/environment#GENERA

d) Declare the properties that describe the subject node, these can only have a plain literal value.

Example of declaring properties for
"http://demo.server.com/environment#GENERA":
Property0 = name (environment:name)
Property1 = target (environment:target)

e) Assign values to defined properties.

Example of value assignment:
environment:name = "GENERA"

f) In the case of container (i.e. a floor), add bag property, otherwise declare a property node (see step h).

Example of adding bag property:
Property2 = room_bag (rdf:Bag)

g) Add resources to the container elements.

Example resource URIs of a Bag:
http://demo.server.com/environment/room.LABPHYS#
http://demo.server.com/environment/room.LABAUTO#
http://demo.server.com/environment/room.LABELEC#
http://demo.server.com/environment/room.LABNET#

h) If there is no need for a container, declare the properties that accept only full URI values.

Example of declaring properties for
http://demo.server.com/environment/room.LABELEC#:
Property0 = target (environment:target)

i) Assign URI to the defined properties.

Example of assigning resource URIs:
environment:target =
http://demo.server.com/environment/room.LABELEC#T1

j) Save model and iterate from c) to j) until no more models are needed.

File Example: Lab_Elec.rdf

k) Merge the models and save as RDF XML file.

Save database file as: Sim_Database.rdf

3 Implementation

Our system was mainly developed in OpenRTM-aist [6] due to the availability of useful tools for our research. We also made use of the V-REP [7] simulator to visualize the navigation of the robot with the help of its RemoteAPI and a web-based panel for monitoring the system from a remote location.

3.1 Speech processing

After the environment has been mapped into an RDF model, it can be used for commanding the robot to navigate to a certain place by using natural speech. For this, we used the following phrase pattern:

"Go to the" + TARGET + "and take it to" + DESTINATION.

Where "Go to the" and "and take it to" are merely string literals, TARGET is a string variable determining the first navigation goal, and DESTINATION is a string variable determining the second navigation goal.

This audio signal was processed by using the OpenHRI [8] package for OpenRTM-aist. The subsystem consists of three RT-Components (RTC), producing two character strings containing the TARGET and the DESTINATION as output. The RT-Middleware subsystem for speech processing is shown in Fig. 3.

3.2 Navigation algorithm

The navigation algorithm was implemented using a SPARQL engine [9]. This algorithm takes the two results of the speech processing subsystem as inputs.

3.2.1 Using SPARQL queries

This navigation algorithm was implemented as an RTC (Query_Eng). It uses the SELECT query, which works with the following logic:

a) Let the data be represented as an RDF Triple:

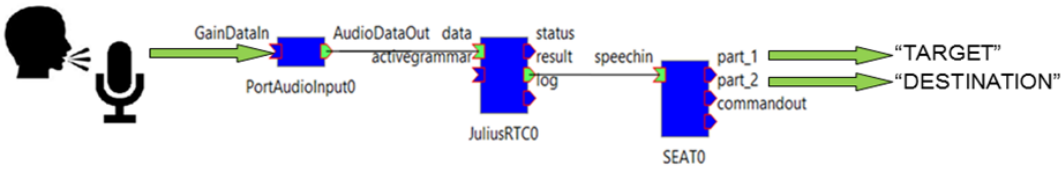

Fig.3 Speech recognition subsystem.

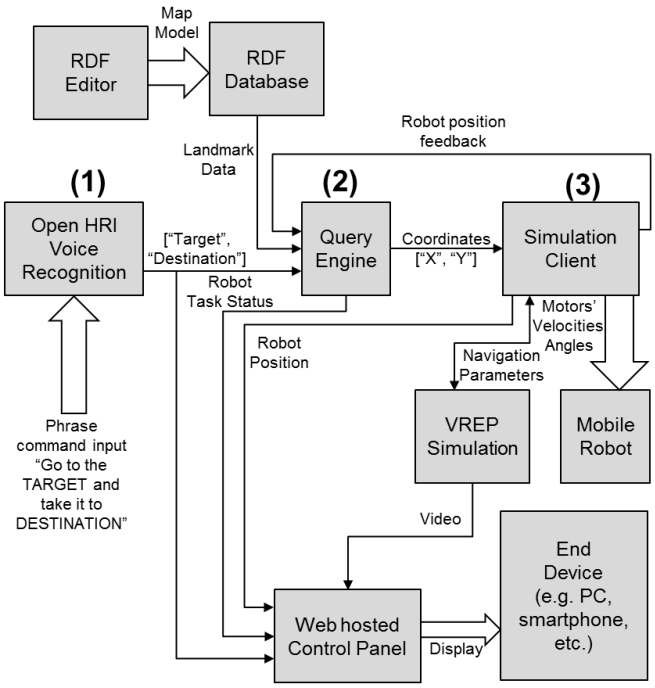

Fig.4 Diagram of the test system.

<http://demo.server.com/environment/room.LABELEC#>
<http://demo.server.com/environment#name>
"Electronics Lab".

b) The query is coded as follows:

SELECT ?resource
WHERE {?resource
<http://demo.server.com/environment#name>
< "Electronics Lab">}

c) This query retrieves the URI of the resource described by the RDF Triple.

Query Result:
URI:
http://demo.server.com/environment/room.LABELEC#

d) By utilizing SPARQL SELECT queries, the resource (URI) is retrieved, which contains TARGET (e.g. Electronics Lab.) as its name.

e) Then, the "x" and "y" properties of the resource are retrieved to get the Cartesian coordinates.

f) By using the URI of the TARGET, the algorithm looks upwards in the RDF database structure and retrieves the URI of the uppermost level (i.e. the floor) as well as its coordinates.

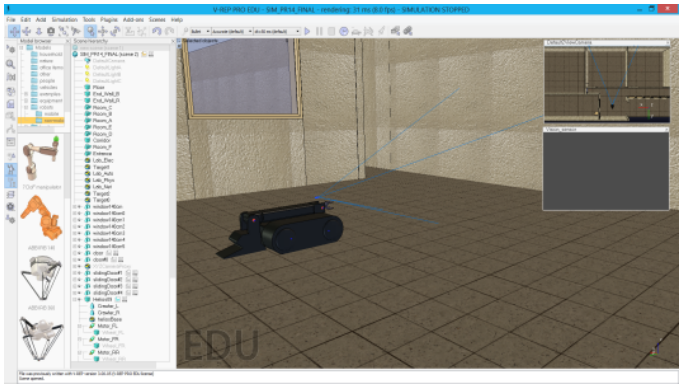

Fig.5 Screenshot of the VREP simulation.

3.3 Testing

In Fig. 4, the main software blocks and data flow are described. Fig. 5 displays a screenshot of the robot in the simulator and Fig. 6 shows the RT-Middleware representation of the system. In order to test the system, spoken command using the pattern defined in section 3.1 were given by the human user through a microphone, then the system performed the following steps:

a) The audio signal is processed by the OpenHRI subsystem (AudioIn RTC, Julius RTC, SEAT RTC), producing two strings as output (Fig. 4, step 1).

b) The result enters the Query_Eng RTC, performs the pertinent queries, and proceeds to send the information about the TARGET (or DESTINATION) coordinates. (Fig. 4, step 2).

c) The V-REP simulator client drives the robot and sends its coordinates as feedback to the Query_Eng RTC. A validation is done every cycle for the robot to reach the right floor, the right room, and finally the TARGET (or DESTINATION) coordinates. (Fig. 4, step 3).

d) When the robot arrives at the TARGET coordinates, the Query_Eng RTC repeats the cycle from step "b", until the robot arrives to DESTINATION. If the robot arrived at the DESTINATION coordinates, the program ends.

As for testing, a human user input the phrase "Go to the Pencil and take it to ElectronicsLab", where "Pencil" is TARGET and "ElectronicsLab" is DESTINATION. The robot followed a sequence of goals which consisted in firstly reaching the coordinates of the door corresponding the room that contained the "Pencil" (which was "AutomationLab"), and afterwards reaching the coordinates of the "Pencil". When the robot reached the "Pencil", it went back to the coordinates of the room's door to exit, then headed to the coordinates of the door corresponding to "ElectronicsLab". Finally, it went to the center coordinates of that room.

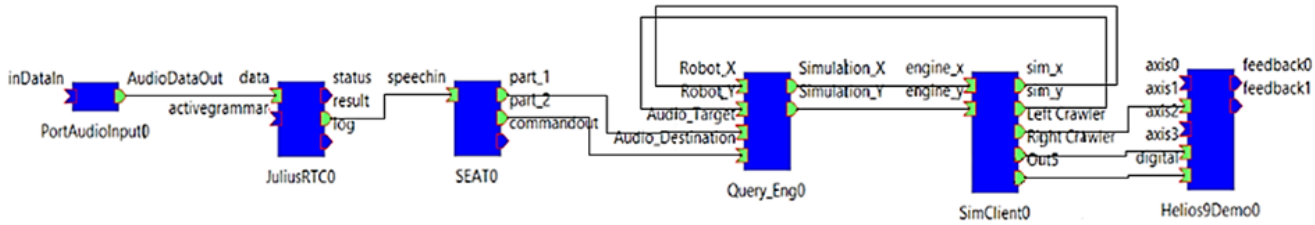

Fig.6 RT-Middleware representation of the system.

4 Conclusion

In this article we proposed a simple method for mapping landmarks by using the Resource Description Framework (RDF). The information is serialized as RDF/XML files, where the files are loaded into a portable database that provides a robot the ability of understanding human user navigation commands, via an artificial intelligence algorithm based on SPARQL queries. Its functionality was validated by letting a human user to command the robot by using speech (a phrase pattern), then the robot reasoned the queries and navigated to the desired coordinates successfully. The test system was implemented by using OpenRTM-aist. Future improvements consist of enhancing the reasoning and navigation algorithms to make the approach scalable to various situations, including navigation in practical situations, such as exploration in disaster and other difficult sites, both indoors and outdoors.

Acknowledgement

The first author acknowledges support from Instituto de Innovación y Transferencia de Tecnología (N.L., MX), Consejo Nacional de Ciencia y Tecnología (MX) and Roberto Rocca Education Program.

References

[1] Toro, W.; Cozman, F.; Revoredo, K. and Reali Costa A.; “Probabilistic Relational Reasoning in Semantic Robot Navigation”, Proceedings of the 10th International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2014), pp. 37-48, Oct. 2014.

[2] Wang, T. and Chen, Q.; “Object Semantic Map Representation for Indoor Mobile Robots”, Proceedings of 2011 International Conference on System Science and Engineering, pp. 309-313, June 2011.

[3] Gheta, I.; Baum, M.; Belkin, A.; Beyerer, J. and Hanebeck, U. D.; “Three pillar information management system for modeling the environment of autonomous systems”, Proceedings of the 2010 IEEE International Conference on Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS 2010), pp. 12-17, Sept. 2010.

[4] Manola, F., and Miller, E.; “W3C Recommendation (2004, February 10). RDF Primer”, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

[5] The Apache Software Foundation; “The core RDF API”, https://jena.apache.org/documentation/rdf/index.html

[6] National Institute of Advanced Industrial Science and Technology; “OpenRTM-aist”, http://www.openrtm.org/

[7] Coppelia Robotics; “V-REP: Create. Compose. Simulate. Any Robot”, http://www.coppeliarobotics.com/

[8] OpenHRI; “Opensource software components for Human Robot Interaction”, http://openhri.net/?lang=en

[9] W3C SPARQL Working group; “SPARQL 1.1 Overview”, http://www.w3.org/TR/2013/REC-sparql11-overview-20130321/