Using Spatial Ontology in the Semantic Integration of Multimodal Object Manipulation in Virtual Reality

Sylvia Irawati[1 2], Daniela Calderón[1 2], Heedong Ko[1 2]

[1] Department of Human Computer Interaction and Robotics, University of Science and Technology

[2] Imaging Media Research Center, Korea Institute of Science and Technology

Abstract

This paper describes a framework for multimodal object manipulation in virtual environments. The gist of the proposed framework is the semantic integration of multimodal input using spatial ontology and user context to integrate the interpretation results from the inputs into a single one. The spatial ontology, describing the spatial relationships between objects, is used together with the current user context to solve ambiguities coming from the user's commands. These commands are used to reposition the objects in the virtual environments. We discuss how the spatial ontology is defined and used to assist the user to perform object placements in the virtual environment as it will be in the real world.

Keywords: multimodal interaction, object ontology, 3D object manipulation

1. Introduction

Object manipulation is an important task in virtual environments. Many 3D interaction devices and techniques have been developed to improve object manipulation in virtual reality [1]. Recent approaches to provide users a more natural way of interaction with virtual environments have shown that multiple modes of interaction between the user and the system may be beneficial and intuitive. There have been many works in improving 3D object manipulation using multimodal interaction. However the understanding of the user commands in multimodal systems still causes ambiguities when fulfilling the user intentions with multiple inputs.

To address that issue, in this paper we present a multimodal interaction framework that uses spatial ontology to integrate semantically the multimodal input. The spatial ontology stores the information about virtual objects and spatial relationships between them. It is used together with the current user context to solve the ambiguity problems among the user commands and intentions of where to place the virtual objects. We create the spatial ontology to verify the validity of the user intentions when manipulating virtual objects. Hence it can be used to assist the user when placing and manipulating objects in virtual environments and to perform actions that will be valid in the real world.

We will present first previous works on object manipulation and multimodal interaction in the virtual environments to make a feasible comparison with other approaches. Continuing the multimodal interaction framework is explained. The results of using our framework in virtual environments are then presented. Finally, we draw our conclusions and present future works.

2. Related Works

A number of researches have improved multimodal object manipulation techniques so that they appear natural, following the laws of physics and common-sense conventions. Each of them has used different kinds of procedures in order to enhance the best performance of their applications.

Smith and Stuerzlinger [2] enhanced the system by attaching semantic information to objects in the form of labels "binding areas" and "offer areas". Xu [3] combined automatically-generated placement constraints, pseudo-physics, and a semantic database to guide the object placement. In our approach the objects properties are defined in the spatial ontology and combined with the multimodal interactions and the user's context, the placements constraints will be resolved via the semantic integration.

More recent works have used ontology definition for complementing their semantic models. Gutierrez et al. [4] propose this methodology in order to map the output of the interaction device to functionality on a particular virtual entity. In this work they use object ontology to express the relationships between interaction devices and virtual entities in Virtual Environments. This approach differs from ours in that we don't relate our ontology with the interaction devices of the multimodal object manipulation. Instead we define a spatial ontology to help in the resolution of the semantic integration in our multimodal framework. Basically Gutierrez et al. map the output of an interaction technique to the functionality on a particular virtual entity. We manage interaction techniques output, user intention and context and the spatial ontology to resolve semantics in order to process the user's intention.

Gutierrez et al. [5] again in other of their works describe a semantic representation of the functions, characteristics and relationships between virtual objects, now with the aim of having adaptive entities, from the geometric and interface point of view, to reutilize them in a variety of contexts without re-implementing the application. They turn the objects in the virtual environment into autonomous and reusable entities. Our approach will present virtual objects as spatial objects. Each object will be defined in the spatial ontology. In our framework you will see the domain dependent part, that in the present state of art is the extension of the base spatial ontology and speech grammar for speech recognition, and the independent domain part. Both parts can be use in different applications without re-implementing the application but adapting it.

Another recent approach is about entities description for the virtual reality applications. Heumer et al. [6] approach is to unify the heterogeneous representation formats within the components in a virtual reality project. For this they mainly categorized all the objects in the environment as 2 main classes, which are not relevant at this point, so that they can homogenized the information and so synchronized it between the components of the application. In our framework we categorized the objects as spatial objects and so any object in a 3d environment can be classified as part of a class of spatial objects as it is going to be further explained.

The last work we want to mention is a very similar approach to ours in terms of ontology definition and processing of the information in it. Latoschik et at. [7] present in his work how to use VR databases for graphics and physics simulations into an AI knowledge base using semantic net representations. They construct an ontology. One part of this ontology describes objects spatial attributes and includes the definition of spatial predicates such as connectable, supports, etc. Our spatial ontology is very similar as this part of their object ontology definition. And another similar point is that they relate the conceptual representations to the lexical data for language parsing and interpretation. Our ontology is closely related with the speech recognition in terms of lexical information.

Although we have presented different type of works which aim is to improve 3D interaction using multimodal interaction, the management of ambiguities between user intentions, context and modalities has not been directly related to the use of ontologies. In this paper, we present a framework for multimodal object manipulation for the virtual environments resolving user intentions across modalities and contexts using spatial ontology.

3. Multimodal Interaction Framework

The multimodal interaction framework for object manipulation is intended to be used for developing multimodal interaction in virtual reality applications. Figure 1 illustrates the components of our multimodal interaction framework. A *user* is the one who gives the input to the system and gets the information presented by the system. *Input* is a multiple input modes, such as speech, handwriting, keyboard, mouse, button, analog, tracker device, etc. Output is multiple output modes, such as vision, audio, haptic, etc. Input can be classified into *recognized input* and *decoded input*. *Recognized input* includes speech, handwriting, and vision. Special speech, handwriting, vision recognition system are required to convert the user input into character strings. *Decoded input* includes analog, button and tracker input. The event generated by pressing a button or repositioning an analog or tracker devices are decoded into character strings or certain values. *Interaction manager* is the component which coordinates data from various inputs to be shown in the outputs. It uses basic selection and manipulation techniques, such as 3D Cursor, Ray Casting, Flash Light, etc., which are provided by *Interaction Technique Functions* component. *Domain specific functions* component is a component which functions depends on the specific application domain. *Object ontology* database is similar to a glossary, but with greater detail and structure that enables computers to process its content. It consists of a set of concepts, axioms (rules), relationships that describe a domain of interest.

More details about multimodal interaction components are shown in Figure 2.

3.1 Input and Output Components

As shown in Figure 2, there are various types of components within input and output components, such as joystick, head tracker, spidar, microphone, speaker, etc. Those devices are connected to the device servers and communicate with the interaction manager through their managers.

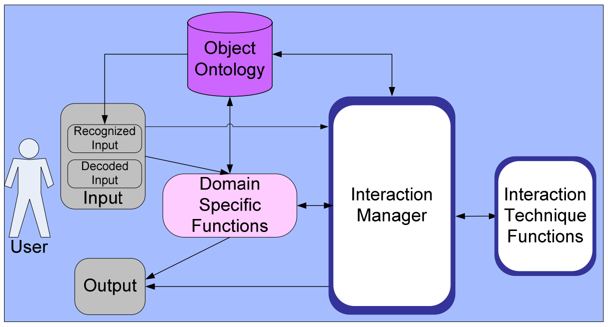

Figure 1. Multimodal Interaction Framework

3.1.1 Speech Server and Speech Manager

One particular case of the input components is the speech server and speech manager. The speech server is going to require an extra utility file. This file is the speech grammar file. This file consists in all the possible lexicon words that the speech server is going to recognize. Each group of words described in the speech grammar is known as a rule. The grammar specification may include semantic interpretation so that it can be used by speech recognition to perform subsequent processing of raw text to produce a semantic interpretation of the input. Some part of the speech grammar is shown in Figure 3. *List* means that the elements of the rule may be one of them which are mentioned in the list. *Property name* and *value* are used for producing a semantic interpretation. Some part of the grammar is related with the object ontology which is going to be explained more detail in the Section 3.2.2.

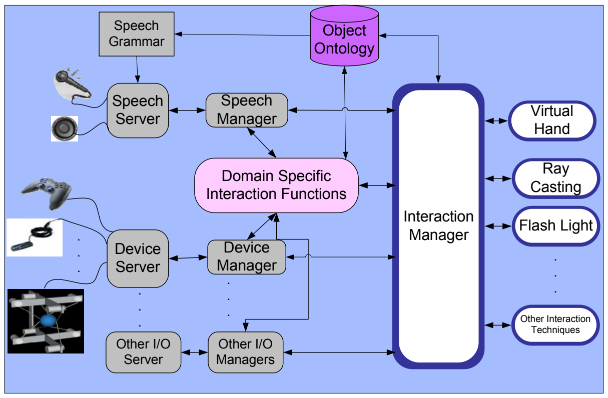

Figure 2. Detail Components of Multimodal Interaction Framework

After getting the speech interpretation result, the speech server sends that result to speech manager for further processing and then the speech manager will pass those values to interaction manager.

3.1.2 Device Server and Device Manager

The device server, in which the interaction devices are connected, is responsible for capturing the user input and sending the feedback to the output device. As shown in Figure 2, interaction devices, such as joystick, tracker, spidar, etc. are connected to the device server. Each device can be connected independently in different computers, hence, it is possible to have more than one device server. For configuring each server, a server configuration file is required. This file specifies the device properties, such as device name, update rate, etc.

The device server communicates with device manager to send the device values or get the feed back values and send those values to the device. Receiving the input from device servers, the device manager maps those values to the certain meaningful values, such as, user head position and orientation (pose), user hand pose, etc. based on the script configuration file. Then, those values are sent to the interaction manager.

3.2 Interaction Manager

The role of interaction manager is integrating the interpretation result of multimodal inputs to be a single complete interpretation and then sending it to the interaction manager. The multiple inputs may be complement, contradiction, substitution or redundant. *Complement*, two or more input modalities complement each other when they are combined to issue a single command. For example, the user pointed to some direction, and utters “Select the lamp”. The user hand pose and utterance are combined to find the object lamp which is located in the user pointed direction. *Substitution*, two or more input modalities substitute each other when those modalities have the some function, so one modality can substitute the others. For example, the user may use button or speech input to change the interaction mode. *Contradiction*, two or more input modalities are contradictive when they produce a contradictive interpretation. For example, the current mode is manipulation mode. The user moves the interaction device to the left while he also gives speech input “Translate the table to the right”. *Redundant*, two or mode modalities are redundant when they issue the same command. For example, moves the interaction device to the right while he also gives speech input “Translate the table to the right”.

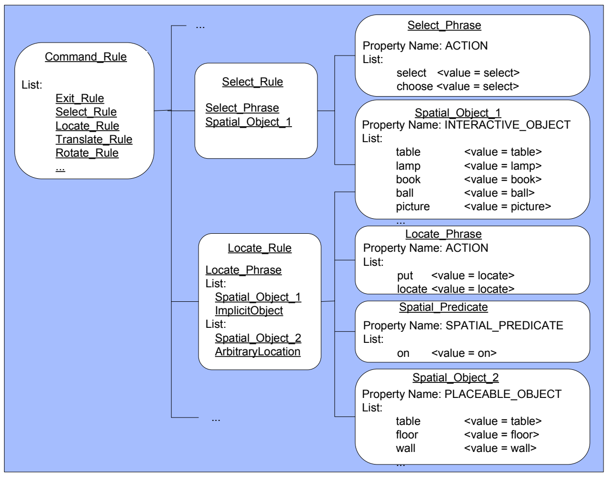

Figure 3. Speech Grammar

In our framework, the user context and the object ontology are used together to integrate the multimodal inputs. Both are described in detail in the following sections.

3.2.1 User Context

The interaction manager maintains the user context. It stores the user interaction history in the interaction log. It is used for finding the object and location according to the user intention which is mentioned implicitly in the speech input. The user interaction, especially speech input may be related with the previous user interaction.

The interaction manager communicates with interaction technique functions which are shown as virtual hand, ray casting, and flash light in Figure 2, to find the list of selected objects. Receiving the user head or/and hand pose and depending on the current interaction techniques which is used, the interaction manager sends those values to the interaction technique modules for updating the current head/hand avatar, finding the selected objects, or manipulating the selected objects, depending on the current interaction mode. To find the area that the user is pointing, it will depend on the interaction technique which is used. For example, the user can tell the system to use ray casting interaction technique which finds the objects of the virtual environment that intersects with the ray emitted by the user hand, in this case the user hand position and orientation is manipulated by the interaction device. In case of virtual hand, the area to find the objects will be specified to where the virtual hand is positioned in the virtual environment and which objects are intersecting with it. More specific, the object touched in that moment of the interaction process. Other interaction techniques are available, and depending on which interaction the user told the system to use, the pointing area of the user hand will be determined.

3.2.2 Spatial Ontology

The object ontology is used for sharing the knowledge between the interaction manager, the speech grammar, and VR application. The object ontology can be classified into domain-independent and domain-dependent ontology. The domain-independent ontology is reusable. Once it is defined, it can be used for other application domain. The domain-dependent ontology is specific to application domain. It may need to be redefined depending on the application domain.

In this framework, the spatial ontology is defined. It has the information about the virtual objects and the spatial relationships between them. Each virtual object is defined as a spatial object with attributes as position and orientation. A static object is generalized to spatial object; it is a spatial object which can not be manipulated by the user. An interactive object is a spatial object with which the user may interact. The user can change the position and the orientation of the interactive object. A placeable object is an object which can be a base of other objects. Horizontally located object is an object which is usually located on the horizontal surface of the object. Vertically located object is an object which is usually located on the vertical surface of the object. As shown in Figure 4, object wall and floor are static objects. The others are interactive objects. Wall, floor and table are placeable objects; the user may put something on those objects. Wall and picture are vertically located objects and the others are horizontally located object.

Each object may have one or more spatial relationships with other objects. That kind of relationships can be seen as a constraint object placement which is defined in the object ontology. It includes common sense knowledge; it is the relation of the objects in the real world environment. For example, we know that the chair should be placed on the floor and not on the table; the lamp is placed on the table, etc. Those constraints can also be defined in the object ontology.

The object ontology can be used to solve the ambiguity in the user speech command. Given the user hand pointed direction and the speech input, "put the lamp there". The interaction manager gets the list of selected objects, for example {table, wall, book}. By retrieving the object ontology, the interaction manager understands "there" as "table" because the relation between table and lamp is defined and the other relationships are not defined in the object ontology. Thus, using the object ontology the system can have a better understanding of the user intended interaction.

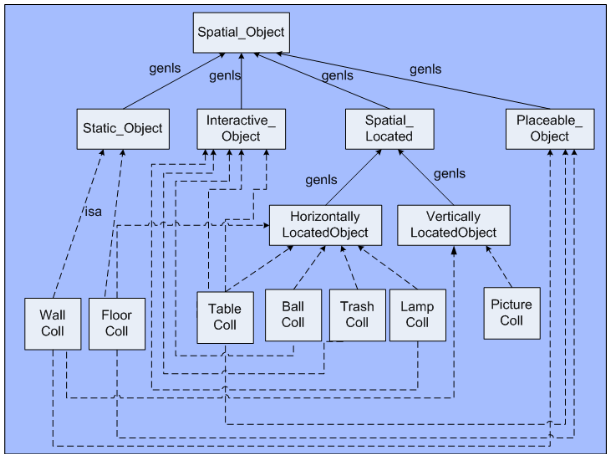

Figure 4. Object Ontology for Spatial Object

Some part of the grammar, terminal node which is related with object type, such as table, floor, wall, lamp, etc. shown in Figure 3 in the rule *Spatial_Object_*1 and *Spatial_Object_2* are corresponding with the class defined in the object ontology shown in Figure 4. Rule *Spatial_Object_1* has property name INTERACTIVE_OBJECT. As shown in Figure 4, the object ontology has an Interactive_Object class. By querying to the object ontology to get the instances of Interactive_Object, the phrases and values of the rule *Spatial_Object_1* can be generated. The *Spatial_Object_2* can also be generated using the same way. Thus, using the object ontology, the VR application and speech recognition engine have the same knowledge of the objects. Once the object ontology is modified, the speech grammar can be easily adjusted according to the object ontology.

3.3 Domain Specific Functions

This component depends on the application domain. As shown in Figure 2, it communicates with the object ontology, speech manager, device manager and interaction manager.

Domain specific function component communicates with object ontology for querying the object description and asserting the fact that is specific to the application domain. It communicates with the speech manager to get the speech interpretation which is not part of domain-independent interpretation or to send the voice feed back which depends on the application domain. It communicates with the device manager to get the device values to be mapped into meaningful values depends on the application domain which is not accommodated by interaction manager. It communicates with interaction manager to get list of selected objects.

4. Implementation and Result

We have implemented a 2 prototype applications based on the presented framework which simulates a virtual room with several objects inside. The user can interact with the system by giving a speech input command and control the hand position and orientation by controlling devices such as joystick, wand, or spidar. Each object has properties such as, objectID, position, orientation, size, objectStatus, objectAlignment, etc.

Our implementation is based on NAVERLib [11], microkernel architecture in the distributed network environment. It provides libraries for a variety of interactions, interfaces, and virtual contents than can be composed in the VR system. In order to communicate with the interaction device, we use VRPN library [12] which consists of a set of classes within library and a set of servers that implement a device-independent, network transparent interface between application programs, and the set of physical devices (trackers, buttons, etc.). The device server, in which the interaction devices are connected, works as VRPN server, whereas the device manager works as a VRPN client. The device manager communicates with the device server to get the device values or send the reaction feedback to the output device.

The object ontology is defined in the OpenCyc [13]. OpenCyc is a general knowledge base and commonsense reasoning engine. Cyc API has two main layers, content and transport layer. The content layer categorizes the available functions and provides the function signatures and documentation used by application, whereas the transport layer establishes the connection to a Cyc server and performs the message handling. The content layer has an inference module which is used for inference engine to query and modify the knowledge base. We added our own ontology to the current database to make possible direct querying to the common sense knowledge of all the possible relationships among objects as defined in Section 3.2.2. Moreover, the constraint object placement in the object ontology can be seen as a constrained-based problem, which is defined by a set of variables and a set of constraints [14]. The variables can be seen as the spatial terms and the predicates as spatial constraints. We use the inference engine of OpenCyc to query about the situation presented in the application and so determine the possibility of performing the user's intended action. This truth or false value will be inferred from the spatial knowledge asserted in the database. This knowledge consists to all possible actions in a real world regarding to spatial positioning. It will determine the constraints of the virtual application. As mentioned before, one constraint can be that the books can be on the table but not on the floor, so by the inference engine of OpenCyc those constraints will be determined and taking on count when performing some action in the virtual environment.

We used Microsoft Speech API 5.1 for speech engine. The grammar is defined in Speech API text grammar format. It is used to define the phrases recognized by speech recognition engine and their semantic interpretation. The current implemented grammar can recognize four types of command, "select", "locate", "translate", and "rotate". We define four different actions that are needed to be done by interaction manager, based on the command type. For example, "select" command, the interaction manager will set the objectStatus property become "selected", "locate" command, the interaction manager will move the selected object from the current position to the new position, "translate" or "rotate" command, the interaction manager will translate or rotate the selected object to certain direction.

4.1 Interaction Loop

Figures 5 illustrate our implementation based on the multimodal interaction framework described here and the interaction loop among the components of the framework.

We used speech and peripheral devices to interact with the virtual world. For that, we have device server, device manager, speech server and speech manager as the input and output component. The speech server uses speech grammar to recognize and interpret the speech input. We have implemented the interaction techniques, such as virtual hand, ray casting and flashlight. The representation of each technique is shown in Figure 6. We have implemented a domain specific dependent module which can understand the user command related with rearranging objects inside a room. This module can understand the user command such as "Put the lamp on the table", "Put the picture there", etc. This module also considers about the relation among objects, for example, the lamp is located on the table. If the user translates and rotate the table, the lamp will move together follows the object parent, in this case is table. Those relationships are store in the object ontology which is defined in the OpenCyc database, as explained in Section 3.2.2.

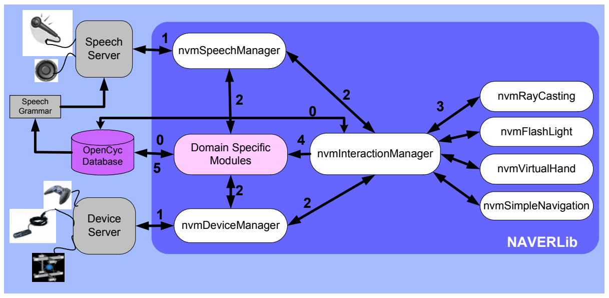

Figure 5. Interaction Loop Diagram

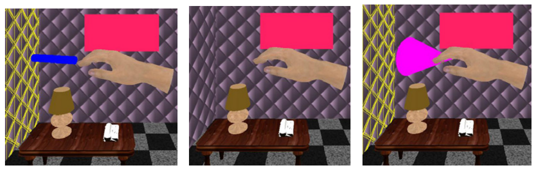

Figure 6. Ray Casting, Virtual Hand and Flash Light Techniques

The interaction loop which occurs in that framework is described below. Step 0 is the initialization step and the others are repeatedly called in the interaction loop.

In the initialization step, the interaction manager initializes the connection to OpenCyc database, gets the object description (e.g. static / interactive object, vertically / horizontally located object, placeable object) from database, and asserts the fact (the current object of the scene graph to the database. The domain specific module initializes the fact which is related with the application domain.

At the interaction loop, first the speech server, using the speech grammar definition file, recognizes the raw speech input. At the same time the device server incharge gets the input values from the joystick. At this point the device server, including the speech server, will send all the collected raw data together with a timestamp of each input from all the modalities to the device and speech manager for further processing. The timestamp will be required in the next processes to relate the modalities between them depending on the values gathered and the time relation between them.

Getting the values from device server, device manager passes those values to the interaction manager. Those values are mapped to the meaningful values such as, head pose, hand pose, etc. depending on the configuration written in the script file. The device manager may also send a device values to the domain specific modules. At the same time, the speech manager sends the speech interpretation result to the interaction module. Since speech is domain dependent, so that, some part of the speech interpretation is domain-dependent and can not be understood by the interaction manager. Thus, speech manager sends the interpretation result to the interaction manager and domain specific modules.

Getting the head/hand pose, interaction manager sends those values to the interaction techniques module (e.g. virtual hand, ray casting, flash light module) depending on the current interaction technique which is used. Those modules map those values into the proper interaction tasks (e.g. updating the head/hand avatar, checking the intersection with the objects, changing the object position/orientation) depending on the current interaction mode. If the interaction mode is selection, the interaction module sends the list of possible selected objects to the interaction manager for further processing.

Getting the result from the interaction technique module and speech interpretation result, the interaction manager combines the input to find the user intended interaction. If the user command is domain specific command, it may not be able to be understood by the interaction manager. The interaction manager will send the result from interaction technique module to the domain specific interaction module to be integrated with other input mode, in this case, speech input. The input timestamps is used in integrating the multimodal inputs. After finishing selection or manipulation task, interaction manager stores the interaction information such as, action and selected object in the interaction log. This log can be used by interaction manager for understanding the next user input which may lack of information.

After the user released the selected object, the domain specific module updates the fact in the database related with domain specific knowledge.

4.2 Results

We have tested our framework using spatial ontology for semantically integrating the multimodal object manipulation in 2 different virtual reality applications. These applications have been tested in a CAVE Environment. CAVE system is suitable to give users a wide angle of field of view and a fully immersive feeling. Spidar [15] which is composed of 4 strings attached to the corners of the CAVE and end effector, provides a human scale device and does not interfere the user's visibility volume. We investigate the use of speech, spidar, and joystick for evaluating the multimodal interaction in our framework. The user wears a wire-headset to give a speech command to select/manipulate the object or to change the interaction mode. The dominant hand of the user grabs the end effector of the spidar to control the hand position. The other hand holds a joystick to control the hand orientation and to change the interaction mode/technique using joystick button. The system also provides a stereo graphics which is seen through the stereoscopic glasses.

4.2.1 Virtual Room Application

The first application simulates a room with furniture and other interior decoration, such as table, lamp, phone, picture, ball, trash, etc., as shown in Figure 7. The user can rearrange the objects inside the room. The application has been demonstrated to several users. Most of the users do not have any difficulties in using the existing interaction technique (virtual hand, ray casting and flash light). They have different preferences in choosing the interaction techniques. Some users prefer the ray casting instead of flash light because too many objects are selected when using the flash light. It may make the user confused in selecting the objects. Some of them do not like the ray casting technique since the user has to point exactly to the object. It is difficult to locate the ray into the small and far object. By providing the user with more interaction techniques, the user can have more options to do the interaction task.

Relating with multimodal inputs, the users do not have any difficulties in using the interaction devices (spidar and joystick), however, some of them have a problem with the speech input. The speech recognition can be improved by doing more training in the speech server and extending the speech grammar with more vocabularies. However, speech input requires the user to remember the accepted commands. Adding speech as an alternative interaction channel is suitable, especially for complementing the current spidar device which does not have any buttons. Using speech, the user can change from one interaction mode to the others; meanwhile he controls the hand position using spidar. Most of the users feel that multimodal inputs are more efficient compared to single modal input. By combining speech and pointing devices, the user can specify his intended object or action. For example, when the user wants to select object lamp, the user can say “select the lamp” and pointing to certain direction. He does not need to locate the selection tool to the object lamp and press the button to make the lamp selected.

4.2.2 Magic House Application

The second application is a magic house, with other kinds of objects, such as balls, sofa, wall clock, etc. Refer to Figure 7. The interaction scenario is defined below. First, the user is outside the house. When the user is outside the house, the objects are floating in the air. Once the user comes into the house, the objects move to land on the proper location. The user may change the object position and orientation using speech and other input devices. Since this is magic house, the user may also change from one object to be other object. Finally, when the user leaves the room, the objects are again floating in the air. If there is another user comes in to the room, the objects move to land on the proper location.

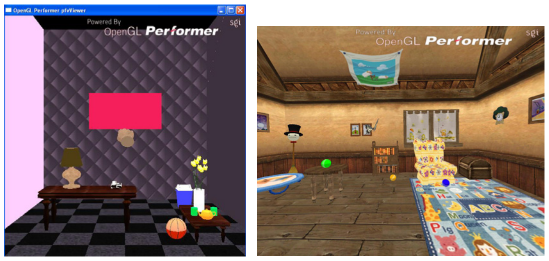

Figure 7. Virtual Room and Magic House

For accommodating the interaction needed in the Magic House application, a new domain specific interaction module is defined. The same interaction manager can be used to accommodate the general task, such as mode changing, selection, manipulation and navigation task. The new interaction module has to support object changing (which is not supported by the interaction manager) and landing, floating and positioning the objects. The other interaction task, such as selection, translation and rotation are done by the interaction manager. Besides adding the new interaction module, the speech grammar and the object ontology have to be adjusted to support the objects in the new application. All objects in the scene graph have to be defined in the object ontology. Once we modified the object ontology, some part of the speech grammar can be generated from the object ontology.

As you can see, in both systems we have spatial objects with whom we can play. For that, these systems can understand spatial commands such as “select the spatial object”, “put this spatial object there”, “rotate”, “move forward”, etc. The system checks the current user context to find which object and position are meant by the user. It checks whether it is related with the previous command, or it is related with the user hand pointed direction or gaze direction. Given the command, the system will find the spatial object referred using the current user context, and find the location that is referred using the current user context and the object ontology, as defined in the Section 3.2.

5. Conclusion and Future Work

We have described a framework for multimodal object manipulation in virtual environment that uses object ontology, which has information about where to put the things in spatial terms. We have shown how spatial ontology is defined and used together with the user context to solve spatial constraints and handle ambiguous interactions across modalities in 3D object manipulation. This framework provides some basic selection and manipulation technique to achieve the interaction task. Any interaction techniques, selection and manipulation techniques can be used and combined with speech input to make the interaction more easily. The user may switch from one technique to others depending on his need.

By separating the domain-dependent and domain-independent part, it reduces the work of VR designer in developing VR application. The domain-independent part can be reused for any application domains meanwhile the domain-dependent part which is specific to application domain may need to be redefined. The spatial ontology for spatial objects is very general, giving the developer and easy way to create new applications with just classifying the objects in the new applications within the range of the ontology already defined. However, the complexity of building a domain-specific interaction module mainly depends on the application, what kind of interaction the user wants to do in the virtual world. If the interaction task is very specific, it will require more work on defining a new domain-specific interaction module.

Using the object ontology defined in this framework, the VR application, interaction manager and the speech grammar can share the same knowledge. They may have the same knowledge about the objects in the virtual world. In the current state of the project the speech grammar is not automatically generated from the ontology, but there is already a parallel effort to automatized this process, which will give the developer even a easier way to develop the VR application. Moreover, the object ontology is reusable, once we defined the object ontology, it can be used for other application.

In the present project the grammar is domain dependent but a parallel project is been develop to create dynamically the speech grammars from the application ontology. So as the spatial object ontology is increased, the domain application will increase the expressive power.

By now we have implemented three basic interaction techniques, virtual hand, ray casting, and flash light techniques. More basic interaction techniques can be implemented to provide more choices for the user in interacting with the system.

This framework is intended for supporting 3D selection and manipulation in virtual environment. It can be extended to support navigation task in the virtual environment by providing some basic travel and way finding techniques.

The object ontology and the grammar specification can be extended to be more complete and to other application domain. Then, domain-dependent interaction module for those domains can be created and tested using this framework. By testing with more application domains, this framework is expected to be a general framework for 3D interaction in the virtual environment.

From the related works mentioned above, we tried to get the best of each work and construct a complete framework for semantic integration of multimodal object manipulation in virtual reality applications.

References

[1] Doug A. Bowman, Ernst Kruijff, Joseph J. Laviola, JR, Ivan Poupyrev, "3D User Interfaces Theory and Practice", Addison-Wesley, Person Education, Inc., Boston, 2004.

[2] G. Smith, W. Stuerzlinger, "Integration of Constraints into a VR Environment", VRIC, Virtual Reality International Conference, 2001.

[3] K. Xu, "Automatic Object Layout using 2D Constraints and Semantics", Master’s thesis, University of Toronto, 2001.

[4] Mario Gutierrez, Daniel Thalmann, Frederic Vexo. "Semantic Virtual Environments with Adaptive Multimodal Interfaces," *mmm*, pp. 277-283, 11th International Multimedia Modelling Conference (MMM'05), 2005.

[5] Mario Gutierrez, Frederic Vexo, Daniel Thalman. “Semantic-based representation of virtual environments”. International Journal of Computer Applications in Technology 2005 – Vol. 23, No. 2/3/4 pp. 229-238.

[6] G. Heumer, M. Schilling, and M. E. Latoschik, "Automatic data exchange and synchronization for knowledge-based intelligent virtual environments", In Proceedings of the IEEE VR2005, pp. 43-50, Bonn, Germany, 2005.

[7] M. E. Latoschik and M. Schilling, "Incorporating VR Databases into AI Knowledge Representations: A Framework for Intelligent Graphics Applications", In Proceedings of the Sixth International Conference on Computer Graphics and Imaging. IASTED, ACTA Press, 2003.

[8] Park, C.H, Ko, H., and Kim, T, "NAVER: Networked and Augmented Virtual Environment aRchitecture; design and implementation of VR framework for Gyeongju VR Theater", Computers & Graphics, 27, pp. 223-230, 2003.

[9] Russell M. Taylor II, Thomas C. Hudson, Adam Seeger, Hans Weber, Jeffrey Juliano, Aron T. Helser, "VRPN: A Device-Independent, Network-Transparent VR Peripheral System", ACM Symposium on Virtual Reality Software and Technology, pp. 56-61, 2001.

[10] OpenCyc, http://www.opencyc.org/

[11] Stuart Russell, Peter Norvig, "Constraint Satisfaction Problem", Artificial Intelligence A Modern Approach Second Edition, Prentice Hall, 2003.

[12] Sato, M., "Development of string-based force display: SPIDAR", VSMM, Virtual System and Multimedia, 2002.