9

Realizing Semantic Virtual Environments with Ontology and Pluggable Procedures

Yu-Lin Chu and Tsai-Yen Li
Department of Computer Science, National Chengchi University, Taipei
Taiwan, ROC

1. Introduction

Multi-User Virtual Environment (MUVE) has attracted much attention recently due to the increasing number of users and potential applications. Fig. 1 shows the common components that a MUVE system may provide. Generally speaking, a MUVE refers to a virtual world that allows multiple users to log in concurrently and interact with each other by texts or graphics provided by the system. On-line games can be considered as a special kind of virtual environment with specific characters, episode, and ways of interactions. Other MUVE systems such as SecondLife provide a general framework for users to design their own 3D contents and interact with other users through their avatars in a more general way. Although the users are allowed to build their own world, the animations that can be displayed are limited to those that have been prepared by the system. In addition, due to the lack of semantic information, it is not feasible to design virtual avatars that are controlled by the computer to interact with other avatars.

Under the concept of web 2.0, we think future virtual environments will also depend on how easily the users can share their own designs of procedures for customized animations and high-level behaviours. However, it is a great challenge to design an extensible virtual environment system that allows the users to write their own customized procedures that can dynamically acquire the information of the virtual environment and other users. In our previous work, we have succeeded in extending a MUVE system developed by ourselves, called IMNET (Li et al., 2005), to allow user-defined animation procedures to be specified, downloaded, and executed on the fly (Chu et al., 2008). However, in order to enable these user-defined procedures to create richer animations for interactions, we must be able to describe the semantics of the objects in the world in a standard way accessible to all potential animation/behaviour designers.

In this paper, we aim to make use of ontology to describe the semantics of the objects in the virtual environment such that users can design their own animation procedures based on the information. For examples, if we can acquire object information such as 3D geometry, height, and 2D approximation, we can design a motion planning procedure that can generate a collision-free path for the avatar to walk to a given destination. In addition, we have also designed the ontology for information exchange between avatars and added a new information query mechanism to facilitate the communication between avatars. These

new functions will be demonstrated through several examples where the user-designed programs acquire application-specific semantics in the standard ontology format after the programs have been deployed dynamically to other clients' machines.

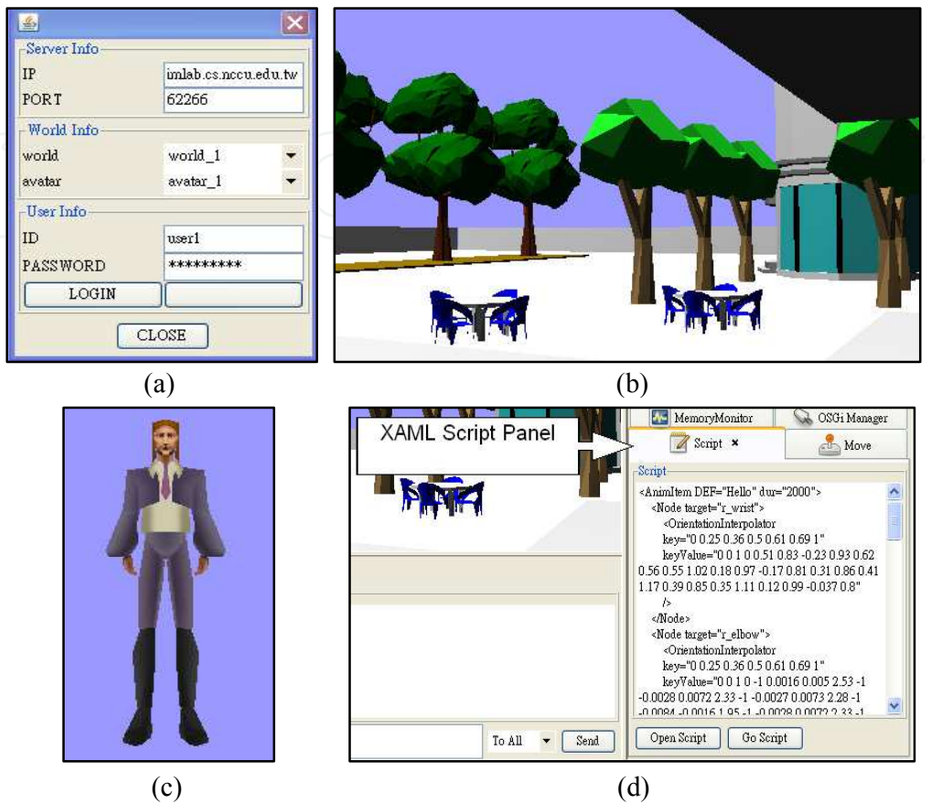


Fig. 1. Common components in a multi-user virtual environment: (a) login; (c) choose an avatar; (b) interact with virtual world; (d) a scripting interface (optional)

The remaining of the paper is organized as follows. In the next section, we will review the research related to our work. In the third section, we will describe the example design of ontology for the objects in the virtual worlds and for the avatars. In Section 4, we will describe the improved communication protocol allowing on-demand query of information among avatars. Then, in the following two sections, we will give examples of how to design animation components that can take advantage of the semantic information to generate richer user behaviours. Finally, we will conclude the paper with future directions.

2. Related work

In this work, we aim to provide a smart virtual environment that can enable richer contents and interactions. Before the concept of semantic virtual environment emerges, there has been much research about how to integrate AI technologies into a virtual environment system. R. Aylett et al. (2001) found that this type of virtual environments have several common features. For example, these systems added components for solving problems such

as configuration, scheduling, and interaction. These systems constructed knowledge-level representation of the virtual scene and supported high-level processing with a natural language interface. These systems also used causal behaviours to replace the simulation of physical features.

The concept of semantic virtual environment was proposed to facilitate advanced applications that allow better communications between agents and the environment (Otto, 2005; Kleinermann et al., 2007). Unlike traditional virtual environments that were designed mainly for visual effects, semantic virtual environments are designed to contain richer structural semantic information adequate for a computer to process (Otto, 2005). For example, a plate on top of four sticks may be easily interpreted as a table by a human but it will be more difficult for a machine to infer its function from low-level geometry reasoning. An analogy of this unstructured information for visual interpretation is the vast home pages on the web. This is also why semantic web was proposed to facilitate automatic processing and communication among web servers. Similarly, in order to facilitate the reuse of 3D design and animation procedures, it is crucial to annotate the objects in a virtual world with semantic information in a standard format such that the same contents can be reused in different worlds or in different MUVE systems.

The idea of SVE has been realized from various aspects by much work in the literature. For example, the SEVEN system was proposed by Otto (2005) as an example of SVE with the concept of software components. It focuses more on the reusability of system components on different MUVE systems instead of components designed by the users. Gutierrez et al. (2005) also have proposed an ontology for virtual human by incorporating semantics into human shapes. They also regarded that the design of ontology is a continuous process where richer semantic information about human attributes should be added in a collaborative fashion. Instead of dealing with human shapes, Abaci et al. (2005) focused on adding action semantics in smart objects to facilitate the interaction between the avatars and with the objects in the virtual environment. Garcia-Rojas et al. (2006) also proposed to add semantic information, such as emotion and expressiveness, to animations in order to facilitate the selection of appropriate animations for a specific scenario.

3. Ontology design in IMNET

For developing and sharing animation procedures in the IMNET system, we will describe our ontology design for virtual objects and avatars in IMNET in this section. The topology described here is to serve as a demonstration example for potential applications; therefore, the design is by no means complete.

3.1 Ontology design of virtual environment

The objective of ontology design for virtual environment in this work is two-fold. First, we would like to keep the information that exists in the original IMNET such as object geometry and transformation. Second, we hope to use an example to show that more semantic information about the virtual objects can facilitate the computation of advanced reasoning procedures such as a path planner that may be designed by the users.

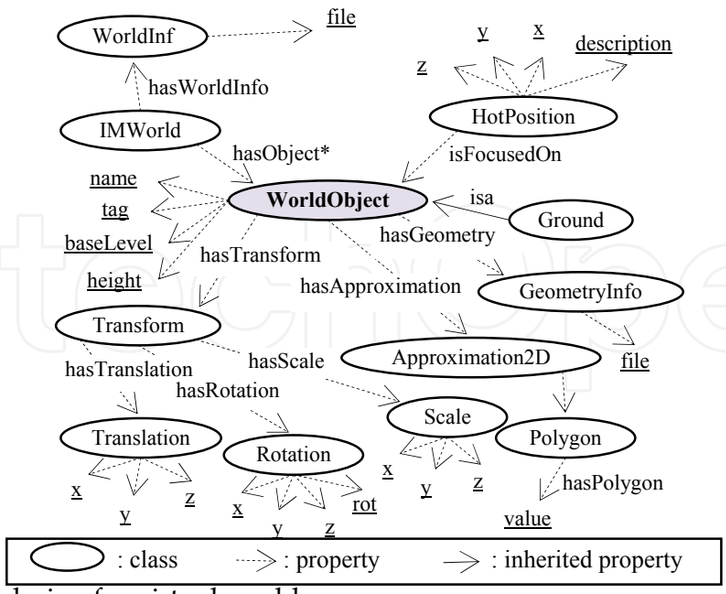


Fig. 2. Ontology design for virtual world

Our ontology design of the virtual environment is shown in Fig. 2. The root of the world document is the IMWorld node, which contains world information (WorldInfo) and all the virtual objects (WorldObject) in the world. In order to retain the semantic information of the virtual objects existing in the original IMNET, we have designed the GeometryInfo and Transform nodes. Each object also has some additional attributes such as name, tag, baseLevel, and height. The tag attribute is designed for the user to denote application-specific properties for virtual objects. For example, in the example of path planning, one can tag certain objects as sidewalk and crosswalk such that these regions can be treated appropriately by the path planner according to their meanings in the world. Each object may also have the attribute of Approximation2D, which is a polygon that can be used to define 2D approximation of obstacles in the environment for the path planner. In addition, if 2D approximation is over simplified for the application, one can also use the baseLevel and height attributes to define 3D approximation regions where the obstacles are located. If these attributes are not available for some objects, they still can be computed from the given 3D geometry and transformation of the objects. Some objects may also serve as the ground of the world through the node of Ground to define the boundary of the world. In addition, some objects could also be treated as HotPosition when they are the foci of interest in the application.

3.2 Ontology design for avatars

In MUVE's, an avatar could be controlled by a real user or by a computer program (called virtual user) if the system provides such a function. Virtual users can be used to by the designer of the virtual world to perform simple tasks such as a watching a gate or offering guided tours. In this section, we describe the basic ontology classes and attributes (as shown in Fig. 3) that we have designed for the applications of avatar interactions. Although an avatar is also an object in a virtual world, they have more active and complicated roles to play. For example, a user may choose to use his/her own animation for a specific behaviour

by using the hasBehaviour property to connect to the Behaviour class. This Behaviour class defines the procedure (with name, package, and codebase) in order to generate the desired animation. In addition, an avatar may contain some basic attributes such as name, geometryInfo, and status. We also use the hasFriend and hasPosition properties to get the friendship and current position information of the avatars.

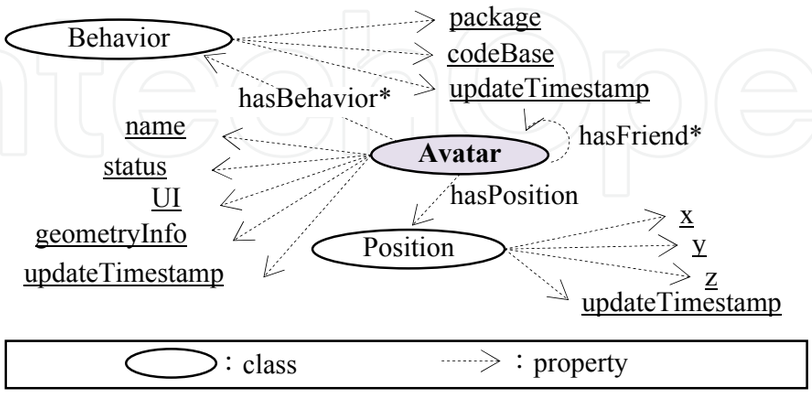


Fig. 3. Ontology design for avatars

3.3 Using ontology to load the virtual world

In the two subsections above, we have defined an ontology for the objects and avatars in the virtual world. However, in the original IMNET system, the geometry of the virtual world is loaded from a single VRML file. The geometry is parsed and converted into the underlying format for 3D display by a module called VRMLModelFactory as shown in Fig. 4. In order to augment the system with semantic information, we have split the geometry into several VRML files with one file for each object. This file is specified in the geometryInfo attribute of every worldObject. We have adopted the Web Ontology Language (OWL) established by W3C as the file format for the ontology of the virtual world. As shown in Fig. 5, the system

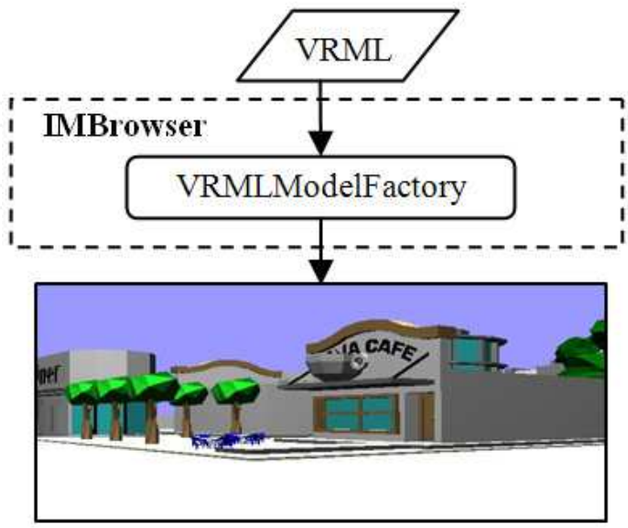


Fig. 4. Processing a single VRML file to generate the virtual world

first loads and parses the OWL file into an object format through the automatic generated Java class and the Protégé API. The geometry file for each object is then retrieved from the ontology and loaded into the system by the VRMLModelFactory module.

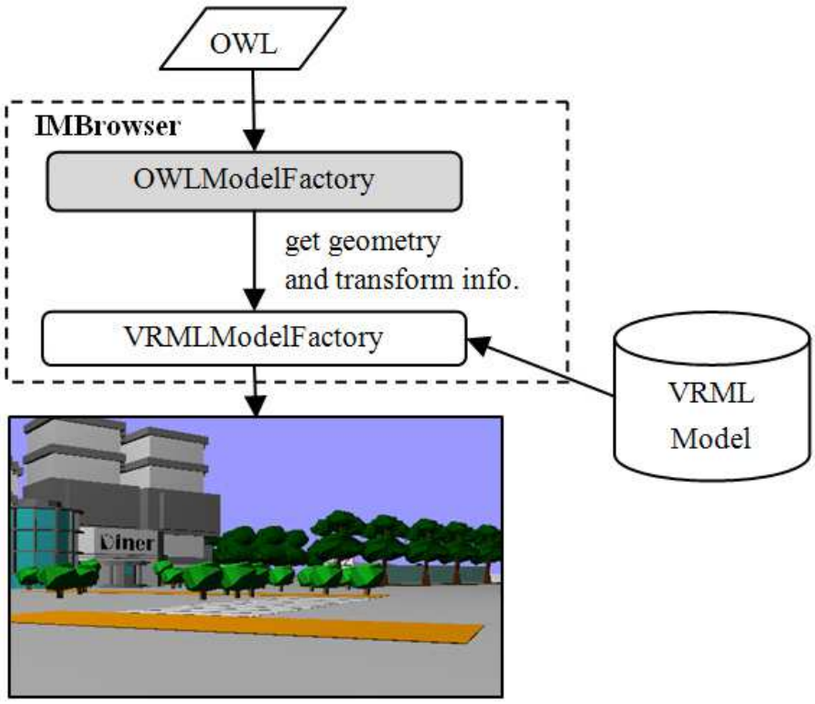


Fig. 5. Processing an OWL file and loading multiple VRML files to generate the virtual world

4. Communication protocol for information query

In a semantic virtual environment, we think semantic information should not only be used by internal modules, but should also be accessible to other clients through user-defined pluggable modules. In the previous section, we have described the ontology of the world and how it is loaded into the IMNET system. However, the clients are not required to specify all attributes defined in the ontology of the avatar. In addition, not all information described in the ontology will be broadcast to all clients. Therefore, we need to have a flexible way for the avatars to communicate semantic information with each other. In this subsection, we will describe how we modify the current communication protocol of IMNET to take information query into account.

```
<IMNET from="user1" to="user2">
  <Chat> Hello, How are you? </Chat>
</IMNET>
```

Fig. 6. An example of IMNET message

The application protocol in the original IMNET is similar to other MUVE's that only encapsulates predefined message types in the XML format. The underlying animation

scripting language, XAML, is an example of message type (Li, et al., 2004). Another example is the message for textual information used in the chat module. For instance, in Fig. 6, we show an example where User1 wants to send a <Chat> message to user2. However, in the original design there is no way for the clients to query the information of other avatars that may be defined by the avatar designers instead of the system. This function is crucial for the avatars to exchange information for richer interactions in a semantic virtual environment.

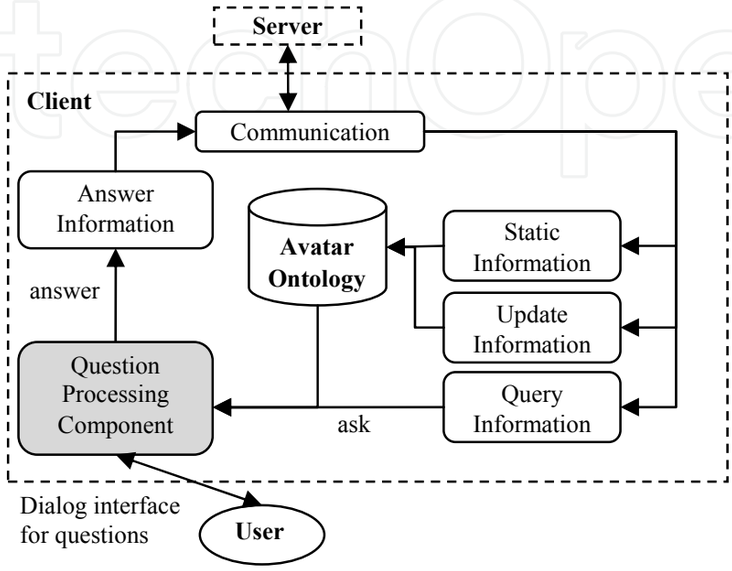


Fig. 7. Client architecture for the processing of three types of information

In the work, we have enhanced the communication protocol of IMNET to incorporate a broader range of message types. We distinguish three types of information exchange between avatars and have designed their processing flow in IMNET as shown in Fig. 7. The first one is static information, such as the id and name properties that are delivered only once at the beginning when a user logs in. The second type is update information, such as the position of an avatar, which is voluntarily pushed to all clients in a more frequent way. An example of update information is shown in Fig. 8. The third type is query information, such as optional attributes or questions, which are sent to the inquirer only upon requests. We have integrated these three types of messages with the tag <Info> and distinguish their types in an internal tag. For example, in Fig. 9, we show a scenario where user1 asks user2 if user2 wants to be a friend of user1. The message is sent with a <queryInfo> internal tag and with a timestamp property. This timestamp property can be used as the id of the query. The query processing component in Fig. 7 may prompt user2 (if user2 is a real user) with a dialog box or evoke an auto-responding component (if user2 is a virtual user) for a response. Then he/she can use this id to indicate this query (askId) in his/her reply message to user1 as shown in Fig. 10.

```
<Info from= “user2” to=“all” timestamp=“1215476355”>
  <updateInfo position=“20 34”/>
</Info>
```

Fig. 8. An example of message for update information

```
<Info from="user1" to="user2" timestamp="1215476323">
  <queryInfo ask="make friend"/>
</Info>
```

Fig. 9. An example of query message

```
<Info from="user2" to="user1" timestamp="1215476330">
  <queryInfo askId="1215476323"  answer="yes"/>
</Info>
```

Fig. 10. An example of responding message

5. Demonstrative examples

In this section, we will give two examples of using semantic information in the virtual world to enhance the functions and behaviours of the avatars.

5.1 Example 1: Motion planning for avatars

A common way for a user to navigate in a virtual environment is by controlling his/her avatar by input devices such as keyboard or mouse. However, it is a low-level task that may not be easy for a novice user to control his/her avatar to reach a destination. There has been much research that proposed to use motion planning techniques to generate collision-free navigation paths for the avatar to follow (Salomon et al., 2003). However, in order to define a motion planning problem, we need to obtain the geometric information of the objects in the environment such that we know where the boundary of the world is and which of the objects needs to be treated as an obstacle.

In this subsection, we will use a motion planning component as an example to illustrate how a user-defined animation procedure can be installed dynamically and retrieve necessary world information for the needs of the application. A user first prepares the procedure as a software bundle according to the specification of OSGi . Then he/she can use a XAML script, such as the one shown in Fig. 11, to indicate where he/she wants to move to. In this script, he/she needs to specify the name of the package, the initial (optional) and goal locations, and the URL for downloading the bundle if it is not already installed. In this case, the bundle will be installed dynamically and evoked through the OSGi mechanism (Chu et al., 2008).

```
<MoPlan package='imlab.osgi.bundle.interfaces'
   codebase='http://imlab.cs.nccu.edu.tw/plan.jar'>
   <param name="-s" value="1.1  2.3"/>
   <param name="-g" value="5.2  3.8"/>
</MoPlan>
```

Fig. 11. An example of specifying a motion planning problem in a user-defined component, MoPlan

In order to generate a collision-free path, a motion planner needs to acquire obstacle information from the world. In our system, the motion planning component obtains this semantic information through the ontology of the virtual world defined in Section 3. According to the obtained obstacle information, the planner first converts the obstacle information into a 2D bitmap and then computes a potential field that is commonly used in a motion planner. And then the planner performs a best-first search to find a feasible path to the goal according to the potential field. Finally, the planner component translates the path to a XAML script and assigns it to the avatar to generate the walking animation as depicted in Fig. 12.

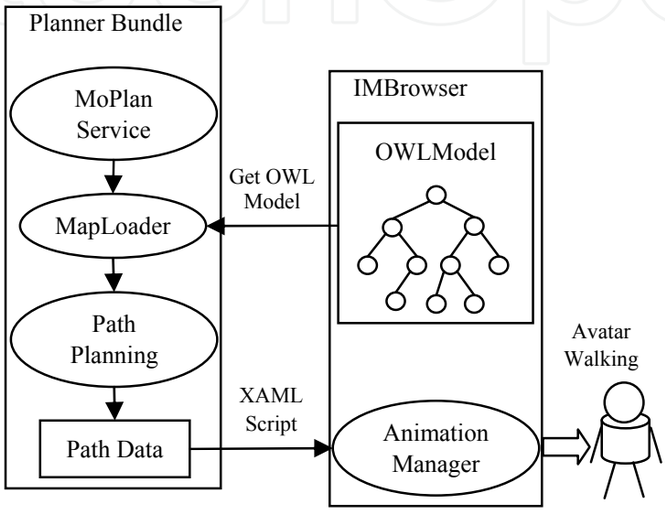
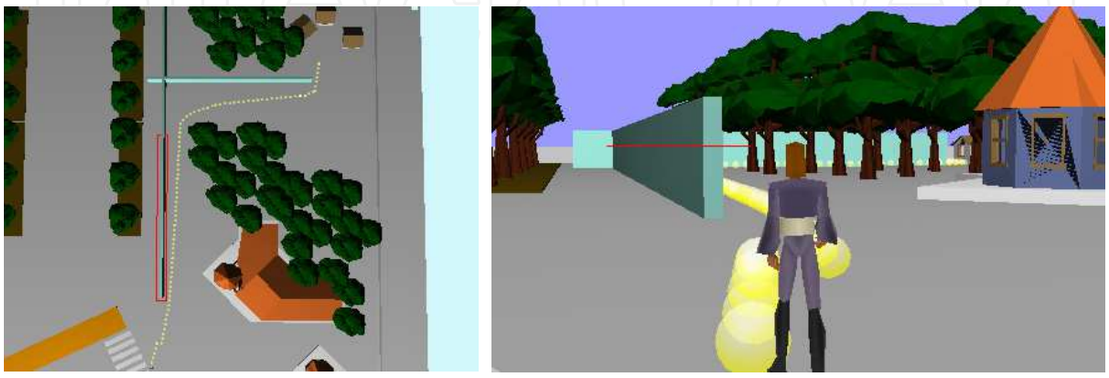


Fig. 12. The process of how to generate animations through the motion planning component

Obstacle information can not only be inferred from low-level geometry but also be given as approximation by the scene designer. In Section 3, we have designed an optional attribute called Ap-proximation2D in the ontology of a virtual object. In Fig. 13, we show an example of the collision-free path generated by the planner by the use of the 2D approximation of the objects in the world. If the planner can find this 2D approximation for an object, it will use it to build the 2D bitmap needed in the planner. If not, it still can build the convex hull of the 3D geometry and project it into the ground to form a 2D approximation. In other words, semantic information could be designed to facilitate automatic reasoning but it is not mandatory. The designers of virtual objects are not obligated to define all attributes in an ontology that could be large in collaborative creation. In addition, the user-defined animation procedures do not easily break down in such a loosely coupled distributed environment either since they can take this into account in the design stage.

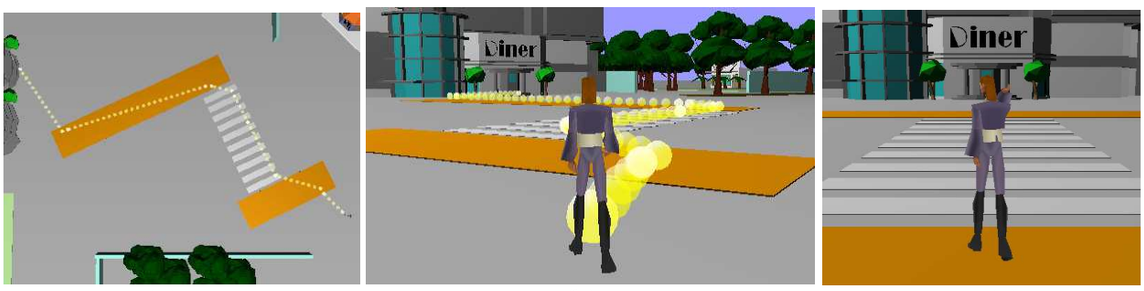
However, some semantic information cannot be inferred directly from geometry. For example, in the virtual environment, there could be some crosswalk or sidewalk regions that need to be used whenever possible. One can tell this kind of objects from their appearance but it would be difficult for the machine to infer their functions through geometry. In this case, the planner has to acquire this information through the semantics defined in the ontology of the virtual world. In the example shown in Fig. 14, the planner

knows where the sidewalk and crosswalk through object tagging in the ontology and makes the regions occupied by these objects a higher priority when planning the path for the avatar. The potential values in these regions are lowered to increase the priority during the search for a feasible path. Consequently, a path passing through these regions was generated in the example shown in Fig. 14. In addition, according to this semantic information, appropriate animations, such as looking around before moving onto the crosswalk region, could be inserted into the motion sequence of walking to the goal, as shown in Fig. 14(c).



(a) (b)

Fig. 13. An example path generated by the path planner: (a) avoiding the obstacles described by the 2D approximation in semantics; (b) a snapshot of the scene from the avatar's view.



(a) (b) (c)

Fig. 14. Another example path generated by the path planner: (a) the path generated by taking crosswalk and sidewalk into account; (b) a snapshot of the scene from the avatar's view; (c) special animation can be inserted before crossing the crosswalk.

5.2 Example 2: Interaction between avatars

An objective of this work is to allow different animation components owned by different clients to interact with each other. The users can communicate through customized tags to acquire the avatar ontology of each other and use this information to perform specific interactions. The user behind an avatar can actually be a real user or a virtual user controlled by a computer program. In this subsection, we will use two scenarios to illustrate these two types of interactions. The first scenario is to demonstrate the interaction between two real users, and the second scenario is for the interaction between a real user and a virtual user.

To facilitate the interaction between the avatars, we have designed a component called SocialService. There are three steps for initiating an interaction between avatars as shown in Fig. 15. A user who would like to initiate the interaction first sends a customized XAML script shown in Fig. 16 to the other avatar (step 1) for it to install this social interaction component (step 2). Once the component has been installed, interaction queries related to social activities can be delivered through the communication protocol described in Section 4 and processed by the SocialService component (step 3).

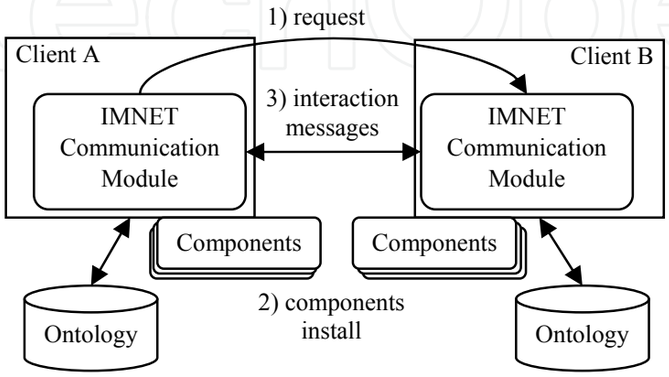


Fig. 15. The steps for initiating an interaction between avatars

```
<SocialService package='imlab.osgi.bundle.interfaces'
    codebase='http://imlab.cs.nccu.edu.tw/social.jar'/>
```

Fig. 16. Starting the mechanism of the interaction between avatars through XML script

In the first scenario, both users are real users. First, user1 would like to invite user2 to be his friend (Fig. 17(a)). Therefore, a query message: "'User1' added you to his friend list. Do you want to invite 'user1' to be your friend as well?" appeared in user2's interface (Fig. 17(b)). If user2 choose 'yes', user1 would be added into her friend list and a confirmation message would be sent back to user1 (Fig. 17(c)). Through the interaction between the two real users, the friend information was updated into the ontology of both avatars.

(a) (b) (c)

Fig. 17. An example of interaction between the real users

In the second scenario, user1 arranged a virtual user called door-keeper to watch the door and provide information to potential guests (Fig. 18(1~2)). When user2 entered a designated region, the doorkeeper would turn to face user2 and ask: "May I help you?" At the first encounter, user2 just entered this area by accident and therefore chose the answer: "Just look around." The doorkeeper replied: "Have a good day!" (Fig. 18(3~5)) The state of the doorkeeper in this interaction was then set to FINISH. After user2 left the area, the state was restored to IDLE (Fig. 18(6)). Assume that after some period of time, user2 approached the doorkeeper again for the second time. This time user2 chose: "I'm looking for my friend." The doorkeeper replied: "Who's your friend?" Then user2 answered: "Jerry." At this moment, the doorkeeper queried the avatar ontology of user1 (named Jerry) to see if user2 is in his friend list. If so, the doorkeeper would inform user2 the current position of user1. Otherwise, the doorkeeper would answer: "Sorry, Jerry does not seem to know you." If there is no such a user called Jerry, the doorkeeper would answer: "Jerry is not in this world." (Fig. 18(7~9))

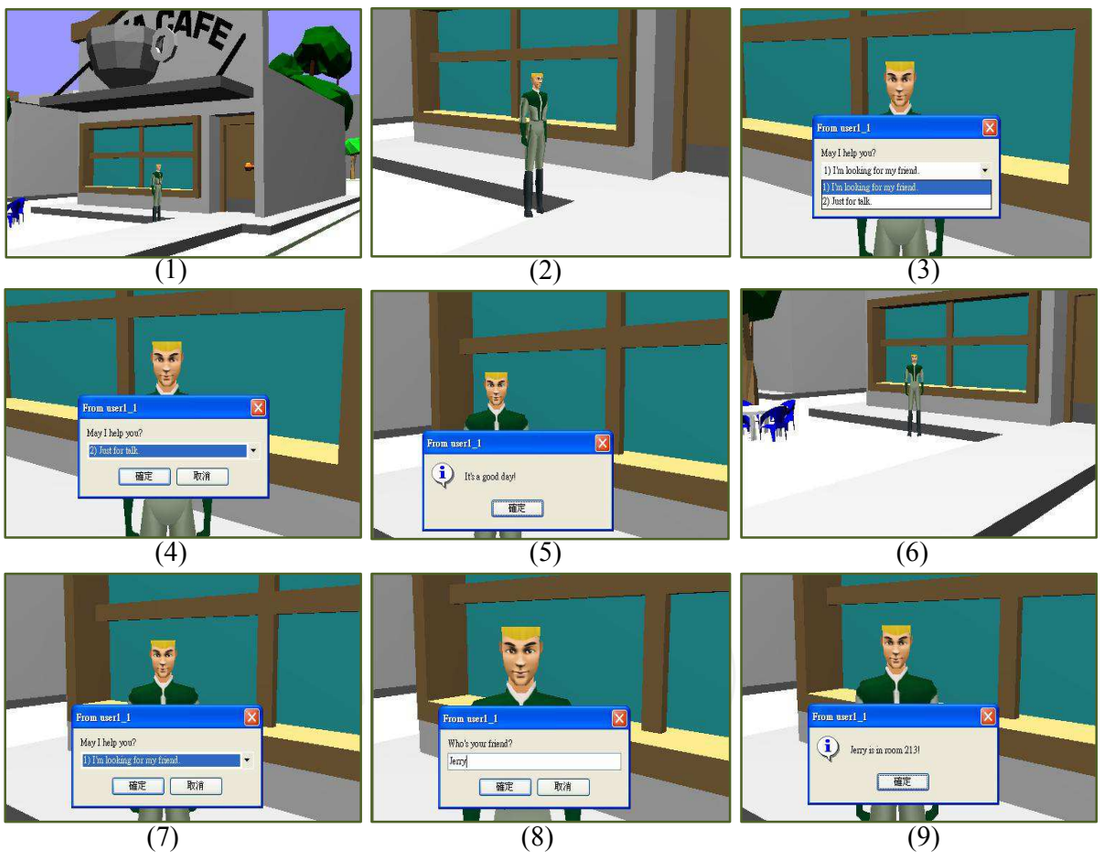


Fig. 18. An example of interaction between a real users and a virtual user (doorkeeper)

6. Conclusions and future work

It is crucial to be able to share user-designed software components to enable richer contents and behaviours in the future development of MUVE systems. In previous work, we have designed a mechanism under OSGi to facilitate dynamic installation of user-designed

software components in IMNET. In this work, we have extended the MUVE system to allow the semantics of the objects and avatars in the virtual environment to be described in the form of ontology. This provides a standard way for the software components to acquire semantic information of the world for further reasoning. We have used two types of examples: path planning and social interaction, to show how users can design their own code to facilitate richer or autonomous behaviours for their avatars (possibly virtual). We hope that these examples will shed some lights on the further development of object ontology and more sophisticated applications.

7. Acknowledgement

This research was funded in part by the National Science Council of Taiwan, R.O.C., under contract No. NSC96-2221-E-004-008. The paper is extended from a conference paper published in the International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI2008)

8. References

Abaci, T.; C'iger, J. & Thalmann, D. (2005). Action semantics in Smart Objects. *Proc. of Workshop towards Semantic Virtual Environments*

Aylett, R. & Cavazza, M. (2001). Intelligent Virtual Environments - A State-of-the-art Report. *Proc. of Eurographics*

Chu, Y.L.; Li, T.Y. & Chen, C.C. (2008). User Pluggable Animation Components in Multi-user Virtual Environment. *Proc. of the Intl. Conf. on Intelligent Virtual Environments and Virtual Agents*, China

Garcia-Rojas, A.; Vexo, F.; Thalmann, D.; Raou-Zaiou, A.; Karpouzis, K. & Kollias, S. (2006). Emotional Body Expression Parameters In Virtual Human Ontology. *Proc. of the 1st Intl. Workshop on Shapes and Semantics*, pp. 63-70, Matsushima, Japan

Guti'errez, M.; Garcia-Rojas, A.; Thalmann, D.; Vexo, F.; Moc-Cozet, L.; Magnenat-Thalmann, N.; Mortara, M. & Spag-Nuolo, M. (2005). An Ontology of Virtual Humans: incorporating semantics into human shapes. *Proc. of the Workshop towards Semantic Virtual Environments*

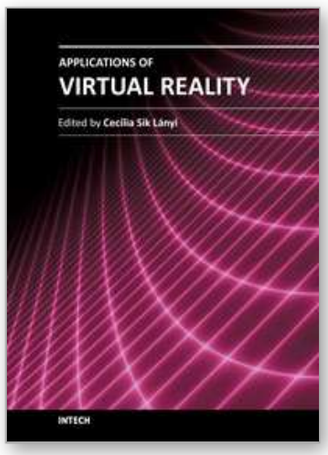
Kleinermann, F.; Troyer, O.D.; Creelle, C. & Pellens, B. (2007). Adding Semantic Annotations, Navigation paths and Tour Guides to Existing Virtual Environments. *Proc. of the 13th Intl. Conf. on Virtual Systems and Multimedia (VSMM'07)*, Brisbane, Australia.

Li, T.Y.; Liao, M.Y. & Liao, J.F. (2004). An Extensible Scripting Language for Interactive Animation in a Speech-Enabled Virtual Environment. *Proc. of the IEEE Intl. Conf. on Multimedia and Expo (ICME2004)*, Taipei, Taiwan.

Li, T.Y.; Liao, M.Y & Tao, P.C. (2005). IMNET: An Experimental Testbed for Extensible Multi-user Virtual Environment Systems. *Proc. of the Intl. Conf. on Computational Science and its Applications, LNCS 3480*, O. Gervasi et al. (Eds.), Springer-Verlag Berlin Heidelberg, pp. 957-966.

Otto, K.A. (2005). The Semantics of Multi-user Virtual Environments. *Proc. of the Workshop towards Semantic Virtual Environments*

Salomon, B.; Garber, M.; Lin, M. C. & MANOCHA, D. (2003). Interactive Navigation in Complex Environment Using Path Planning. *Proc. of the 2003 Symposium on Interactive 3D graphics*.



Applications of Virtual Reality

Edited by Dr. Cecília Sík Lányi

ISBN 978-953-51-0583-1
Hard cover, 210 pages
Publisher InTech
Published online 02, May, 2012
Published in print edition May, 2012

Information Technology is growing rapidly. With the birth of high-resolution graphics, high-speed computing and user interaction devices Virtual Reality has emerged as a major new technology in the mid 90es, last century. Virtual Reality technology is currently used in a broad range of applications. The best known are games, movies, simulations, therapy. From a manufacturing standpoint, there are some attractive applications including training, education, collaborative work and learning. This book provides an up-to-date discussion of the current research in Virtual Reality and its applications. It describes the current Virtual Reality state-of-the-art and points out many areas where there is still work to be done. We have chosen certain areas to cover in this book, which we believe will have potential significant impact on Virtual Reality and its applications. This book provides a definitive resource for wide variety of people including academicians, designers, developers, educators, engineers, practitioners, researchers, and graduate students.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen