A Framework for VR Application Based on Spatial, Temporal and Semantic Relationship

Changhoon Park[1], TaeSeok Jin[2], Michitaka Hiroseo[3], and Heedong Ko[4]

[1] Dept. Of Game Engineering, Hoseo University,
29-1 Sechul-ri, Baebang-myun, Asan, Chungnam 336-795, Korea
chpark@office.hoseo.ac.kr
[2] Dept. of Mechatronics Engineering, DongSeo University,
San 69-1 Churye-dong, Sasang-ku, Busan 617-716, Korea
[3] The University of Tokyo,
4-6-1 Komaba, Meguro-ku, Tokyo 153-8904, Japan
[4] Korea Institute of Science and Technology,
39-1 Hawolgok-dong, Sungbuk-gu, Seoul 136-791, Korea

Abstract. This paper proposes a framework for VR application based on 3 different kinds of hierarchical structures for spatial, temporal, and semantic relationship. To achieve this, we incorporate scene graph, time graph, and ontology into the framework. This approach will enable to integrate synthetic space, real space, and knowledge seamlessly as a new way of developing immersive tangible space.

Keywords: Framework, Immersive Tangible Space, Spatial relation, Temporal relation, Semantic relation, Scene Graph, Time Graph, Ontology, Seamless Integration.

1 Introduction

Recently the Systems Technology Division in Korea Institute of Science and Technology (KIST) is continuously researching and developing the core technology for the intelligent HCI. It has launched a project named "Tangible Space Initiative (TSI)." The art of TSI is based on the concept of tangible space where several physical objects of real space are integrated into virtual objects of synthetic space. Thus, the space creates a new kind of living environment for human which exceeds all the spatial and physical limits.

In TSI, more effective VR interface metaphor is sought: a user may experience an interaction environment that is realized by both physical and virtual means. TSI effort is led by three parallel and cooperative components: Tangible Interface (TI), Responsive Cyber Space (RCS), and Tangible Agent (TA). TI covers how the user sees and manipulates the 3D cyber space naturally. RCS creates and controls the virtual environment with its objects and behaviors. TA may sense and act upon the physical interface environment on behalf of any components of TSI. (Ko, 2002).

R. Shumaker (Ed.): Virtual Reality, HCII 2007, LNCS 4563, pp. 329–337, 2007.

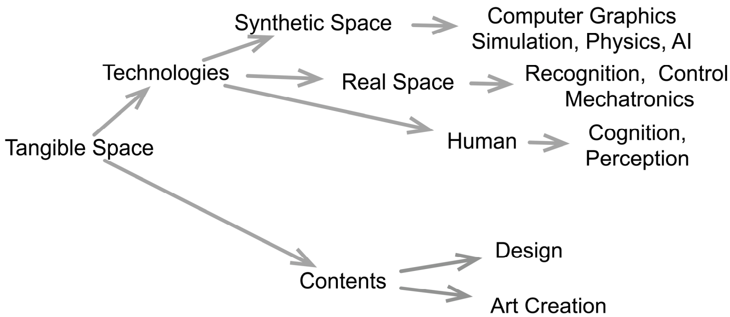

Fig. 1. The initiative involves experts from diverse related fields such as virtual reality (VR), intelligent control, human robotics, image processing, multimedia database, and artificial intelligence

This paper will propose a framework in order to facilitate the research of TSI. For the seamless integration between synthetic and real spaces, we believe that it is important to integrate diverse technologies into the framework. So, the framework will be designed to support adding new functions independently. Then, various experts can focus on their business without understanding others. And, different TSI applications need different functions. So, it will be needed to organize our framework based on XML description. Then, we can optimize and build various applications for fast test.

In this section, we have introduced the motivation of this research relating to TSI. Section 2 will review a framework named NAVER(Networked and Augment Virtual Environment aRchitecture) based on distributed micro-kernel. In section 3, spatial, temporal, and semantic hierarchical structures are explained. Then, we explain about the implementation and present some results that have been achieved. Finally, we concluded in section 5 with possibilities for future work.

2 Related Work

We have been developing a framework named NAVER, which was developed for the Gyeongju VR Theater of the World Culture EXPO 2000 in Korea from September 1 to November 26, 2000. This VR Theater is characterized by a huge shared VR space with tightly coupled user inputs from 651 audience members in real time. Large computer-generated passive stereo images on a huge cylindrical screen provide the sensation of visual immersion. The theater also provides 3D audio, vibration, and olfactory display as well as the keypads for each of the audience members to interactively control the virtual environment. At that time, the goal of NAVER was to coordinate both virtual world and various devices [1].

In recent, NAVER introduced the concept of module to facilitate adding new functions such as handling new input device, visual special effect and so on. And, it is important to support the integration of diverse techniques for synthetic or real space. Therefore, module will enable to extend function of NAVER without detailed understanding or modifying the kernel [2].

Module is defined as a set of callback functions and XML-based specification. These functions will be invoked by the simulation loop of the kernel. As a building block, some of modules can be chosen and plugged into the kernel without modifying the kernel. And, XML description allows specifying a configuration of modules without programming. So, we can configure and optimize NAVER for a specific application or content with evaluating and finding functions causing bad performance. NAVER consists of several modules listed in Table 1. And, we have been developing modules for augmented reality, physics, character animation and so on.

Table 1. NAVER modules

Modules	Description
nvmVRPNClient	This module manages a communication with a VRPN server in which various peripheral devices are connected.
nvmDisplay	This module is used to display the virtual world into multiple PCs.
nvmLoader	This module is used to load the models to construct the scene graph
nvmScenario	This module controls the development of stories in virtual environment
nvmNavigation	This module is used to change the camera position and orientation.

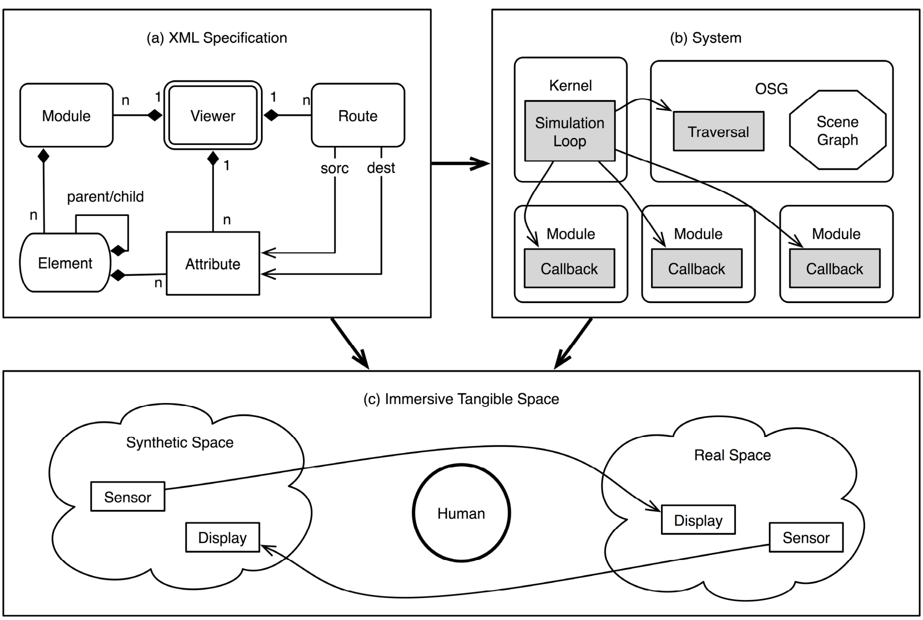

Fig. 2. NAVER enables to define and manage both synthetic and real spaces by means of XML specification consisting of one viewer, and a set of modules and routes. Route is provided to make modules work together by sending and receiving events among them.

3 Three Hierarchical Representations

In this section, we explain how to use spatial, temporal, and semantic relationship for the seamless integration between synthetic space, real space, and knowledge as a new way of developing immersive tangible space. These relationships will be organized into a tree-like hierarchical structure. The structure allows repeating information using parent and child relationship. This provides an efficient way of managing and accessing complex data.

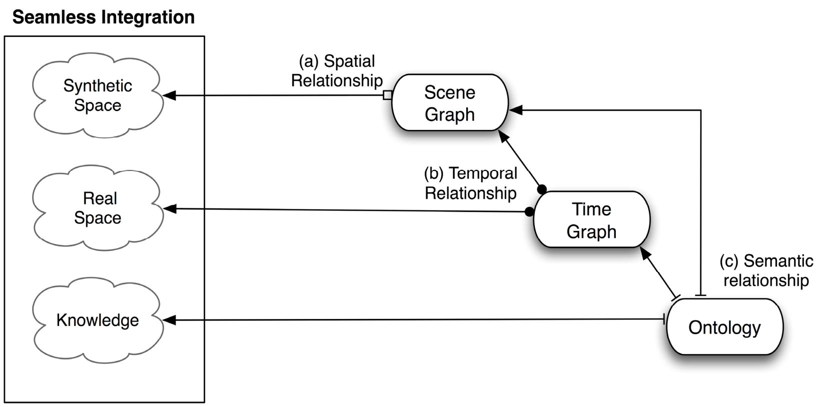

Fig. 3. (a) Scene graph is used to construct a synthetic space by describing its spatial relationship. (b) Time graph enables to synthetic and real space over time by describing the temporal relationship of scene graph and modules for real space. (c) Ontology is used for a domain-specific knowledge which will be applied to manage time graph and scene graph.

3.1 Spatial Relationship

In our framework, scene graph is introduced as a way of representing and managing a synthetic space. The synthetic space is then broken down into a hierarchy of nodes representing either spatial groupings of objects, settings of the position of objects, animations of objects, or definitions of logical relationships between objects. The leaves of the graph represent the drawable geometry, their material properties, and multimedia such as video and audio. This enables to organize synthetic so that it can be rendered quickly and efficiently.

Scene graph enables to organize how some object is located in synthetic space in relation to some reference object so that it can be rendered quickly and efficiently. This means that we can modify and reference spatial relationship of everything in synthetic space. Three are two different spatial relationships when we apply scene graph: Bounding volume hierarchy and transform hierarchy as below.

Bounding Volume Hierarchy. In order to be able to exploit the spatial locality of a scene graph, we need to introduce the concept of bounding volumes for each node. At the bottom of the hierarchy the size of the volume is just large enough to encompass a single object tightly. As you walk up the hierarchy each node has its own volume which tightly encompasses all the volumes beneath it. Bounding Volume Hiearchies are useful for speeding up efficient culling and speeding up collision detection between objects. Bounding volumes change as the transformations and geometry of scene graph nodes change over time.

Transform Hierarchy. All nodes should contain their linear transformation relative to their parent. The transformation hierarchy includes all of the root nodes and root node descendants that are considered to have one or more particular locations in the synthetic space. In this way, the node defines its own local coordinate system relative to its parent. We can transform points in the local coordinate system of the node to the world coordinate system by concatenating all the transformation matrices of nodes in the path from the root node of the scene graph to the current node.

And, we will access scene graph to make not only dynamic and complex synthetic space but also real-time rendering. In general, scene graph is closely related to the performance of culling and state sorting. Culling is the process of removing the objects from a scene that won't be seen on screen. The hierarchical structure of the scene graph makes this culling process very efficient. And, state sorting is the process that all of the objects being rendered are sorted by states such as textures, materials, lighting values, transparency, and so on. Then, all similar objects are drawn together. In our framework, scene graph and its traversing will be updated during run-time to make better performance.

3.2 Temporal Relationship

We introduce a timing model to make more complex and dynamic tangible space which can change its presentation of synthetic space and configuration of real space dependent on not only event but also time. Our framework will use a timing model to control the change of scene graph and modules.

It should be noted that scene graph can include time dependent objects such as keyframe animation, video and audio as well as static. And, module can be added to handle new device as described before. Therefore, timing model enables to describe temporal composition of everything of synthetic and real space. This means we can integrate synthetic and real space by means of a timing model.

We propose a timing model by extending SMIL[1] (Synchronized Multimedia Integration Language), XML-based language to integrate a set of independent multimedia objects into a synchronized presentation. In our framework, this time graph will be used to control not only 3D objects of synthetic space but also interaction devices of real space over time. And, it also enables to schedule event routing between different spaces. Thus, we can build more complex and dynamic VR applications with integrating synthetic and real things at the same context.

---

[1] The World Wide Web Consortium (W3C) for encoding multimedia presentations for delivery over the Web developed SMIL. SMIL integrates a set of independent multimedia objects into a synchronized presentation. This means SMIL is not a content media type because it doesn’t define any particular type of media. Instead, the author can describe a composition of existing multiple media with referring other files in other format.

In SMIL, there are 3 timing container elements for a structured composition. Each element specifies different temporal relationships between their children. The seq element specifies its children handled in sequence, one after the other. The par element specifies that its children be handled in parallel. The excl element activates only one of its children at a time. Thus, it is allowed to construct a hierarchical or structured timeline by nesting 3 timing containers. It means that any child of a timing container element can be an element for media object or other timing containers [3,4].

And, SMIL has a set of attributes to control timing of media object elements and composite elements. Elements have a begin, and a simple duration. The begin can be specified in various ways – for example, an element can begin at a given time, or based upon when another element begins, or when some event happens. It should be noted that this model is dynamic because we can't expect when event happens such as mouse click.

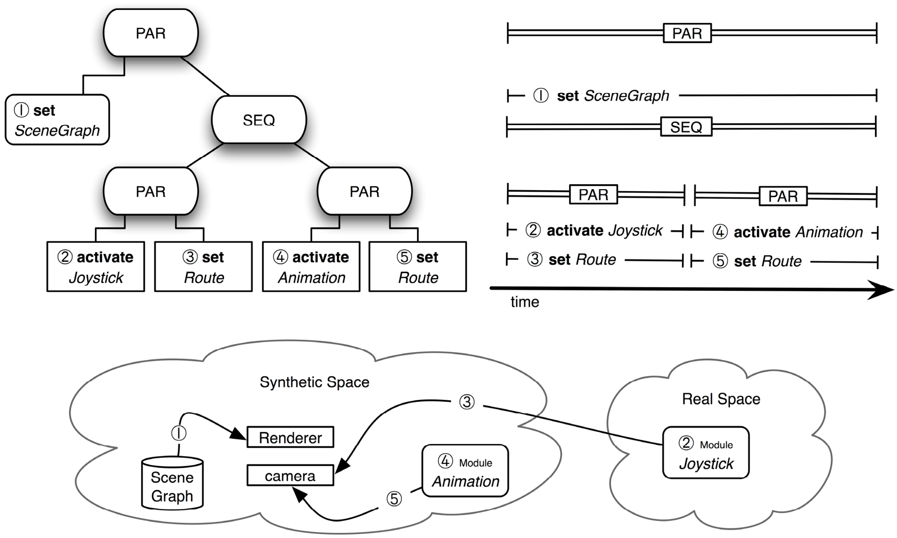

Fig. 4. The root of time graph is a *par* node so its children will be processed in parallel. Then, we can invoke an action ① to assign a scene graph for a visual rendering of synthetic space. At the same time, there is a *seq* node including two *par* nodes. And, the first *par* node has 2 commands ② and ③ as its children. ② will activate a module for joystick in real space. At the same time, ③ makes a path from the joystick to the camera of synthetic space. Then, it will be allowed to navigate synthetic space by controlling joystick. After this, ④ and ⑤ make a camera path animation with activating a module to generate animation and routing its result to the camera. Actually, timing attributes like begin and duration are not explained here.

3.3 Semantic Relationship

This framework use ontology to make knowledge driven tangible space based on semantic relationship. Ontologies provide the theoretic and axiomatic basis to underlying knowledge bases. Based on this formalization, we will express semantic relationships between synthetic and real space. Thus, the supporting framework enables to perform reasoning by inference on the content and the semantics of both synthetic and real spaces.

And, semantic information can be used to perform knowledge driven rendering, interaction, integration and so on. Ontology allows describing domain concepts by means of their properties as well as their relationship. This means that we will use ontology to define domain specific knowledge besides scene graph for spatial relation and time graph for temporal relation. After all, these information will facilitate the development of VR applications with supporting semantics based description and control.

Semantic-based Culling. Culling removes objects that are outside the viewing frustum from the rendering process. In a networked virtual environment, state updates will not be exchanged between players if they can't see each other any reason in a shared virtual space. We'd like to use domain-specific knowledge to improve culling for rendering and network.

Semantic-based LOD. Accounting for level of detail involves decreasing object complexity as it moves away from the viewer or according other metrics such as object eye-space speed or position. We'd like to change not only the complexity of geometry but also various behaviors depending on semantics. This approach will help analysis and manipulate scientific data in virtual space. Ontology will be used to define complexity mapping and semantic distance.

Semantic-based Highlighting. When a modeler makes a scene graph, he gives some name to a node of scene graph or texture to represent what it is. Then, we will use this name to define a domain specific ontology. By using this ontology, we can have a kind of context. This context helps system intelligent with understanding a scene graph. For example, there is node named door. Then we can add transform to its parent in order to open the door when an avatar is close to the door. And, if a name of texture includes "stone", system can set its mass or features to the objects which has the texture especially for physics simulation.

Semantic-based Interaction. In our approach we use ontologies to process the user's input and context semantic integration in order to manipulate virtual devices in a virtual environment. We use a domain independent device conceptualization as part of the ontology describing general virtual entities of virtual devices in combination with a domain dependent world application conceptualization that describes the application environment. This ontology in collaboration with the context information will compute the semantic integration [5.6].

4 Implementation

NAVER can be download from http://naver.imrc.kist.re.kr. This site provides information about new feature, getting started and installation. In recent, the part of visual rendering has changed from OpenGL Performer to OpenSceneGraph. And, many modules for such as physical simulation, character animation, bump mapping, particle system, and augmented reality have been developing based on Cal3D[2], osgART[3] and so on.

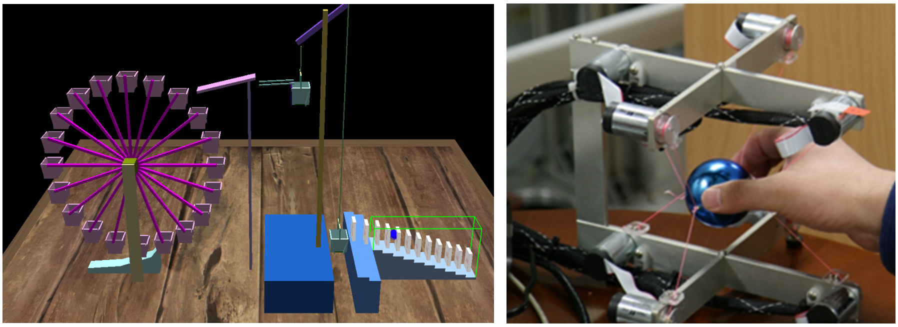

Fig. 5. A complex rigid body System was the recent demo based on NAVER. Providing an interactive rigid body simulation that allows the user to interactively build a different configuration of rigid bodies and simulate them to see how the domino effect is generated with a help of multi-modal interaction including SPIDER(right), and speech command.

5 Conclusion

We have presented a framework for the development of immersive tangible space by integrating synthetic and real spaces seamlessly. To achieve this, scene graph, time graph and ontology are integrated into the framework for spatial, temporal and semantics relationship respectively. The scene graph is used for the composition of synthetic space. By using scene graph, we can coordinate and interconnect synthetic space and various devices over time managed by scene graph and modules. Ontology allows organizing a domain specific knowledge in order to build intelligent tangible space.

Acknowledgments. This research was supported by the Academic Research fund of Hoseo University in 2006. (20060091)

References

1. Park, C., Ko, H., Kim, T.: NAVER: Design and Implementation of VR Framework for Gyeongju VR Theater. Computer and Graphics 27(2), 223–236 (2003)
2. Park, C., Hirose, H., Ko, H.: A Platform for Interactive VR Storytelling. In: Pan, Z., Aylett, R., Diener, H., Jin, X., Göbel, S., Li, L. (eds.) Technologies for E-Learning and Digital Entertainment. LNCS, vol. 3942, pp. 193–202. Springer, Heidelberg (2006)

---

[2] Cal3D is a skeletal based 3D character animation library written in C++ in a platform-/graphic API-independent way. It supports combining animations and actions through a "mixer" interface.

[3] OSGART is a cross-platform library that simplifies the development of Augmented Reality applications by merging computer vision based tracking with OSG rendering.

3. Ayers, J., et al.: Synchronized Multimedia Integration Language (SMIL) 2.0. World Wide Web Consortium Recommendation (2001) http://www.w3.org/TR/2001/ RECsmil20-20010807/
4. Schmitz, P.: Multimedia Meets computer Graphics in SMIL2.0: A Time Model for the Web. WWW 2003, pp. 45–53 (2002)
5. Kalogerakis, E., Christodoulakis, S., Moumoutzis, N.: Coupling ontologies with graphics content for knowledge driven visualization. In: Virtual Reality, 2006, pp. 25–29 (2006)
6. Irawati, S., Calderón, D., Ko, H.: Spatial Ontology for Semantic Integration in 3D Multimodal Interaction Framework. In: Proceedings of ACM International Conference on Virtual Reality Continuum and its Applications, June 2006, pp. 129–135 (2006)