Standardized Virtual Reality, are we there yet?

Mario A. Gutiérrez A.
School of Engineering and Architecture
ITESM Campus Toluca, Mexico
m.gutierrez@itesm.mx

Frédéric Vexo, Daniel Thalmann
Virtual Reality Laboratory
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
{frederic.vexo,daniel.thalmann}@epfl.ch

Abstract

Despite its history of several decades and impressive achievements, developing a VR application is still a complex task. Setting-up a Virtual Environment requires choosing a suitable combination of elements from a large amount of technologies, software frameworks, animation/modeling formats and many other components. In this paper we present an overview of the efforts focused on defining a commonly accepted set of specifications -standards- for the multiple "ingredients" of a VR application. Our analysis of current and previous initiatives provides elements to answer the question about whether or not we can talk about "standard VR".

1. Introduction

This paper presents an overview of the most widely accepted technologies and specifications for building VR applications and its components. Our goal was to find out what are the elements that can be used to develop a standard Virtual Reality application and furthermore, can we really talk about something such as "standard" VR?

When referring to Virtual Reality applications, we consider those computer based simulations where one or more users can interact with a fully or partially immersive environment with 3D imagery. Hence, in this paper we deal with a definition of Virtual Reality which is built upon three main concepts: immersion, interaction and real-time (see figure1). This general definition includes web-based applications, video games, and all sorts of 3D interactive simulations for medical applications, design, training, etc. Our main goal is to identify the most commonly accepted specifications and technologies that are used to build such systems.

In the following section we elaborate on this definition of VR and use it to explain the inherent complexity of this kind of applications. We will continue by explaining the importance of standardization as a key factor for success and continuous development of VR applications. An overview of official and unofficial -de facto- standards for the various components of a VR constitutes the central part of the paper. Finally we conclude by making some reflections about current trends in VR development and its standardization.

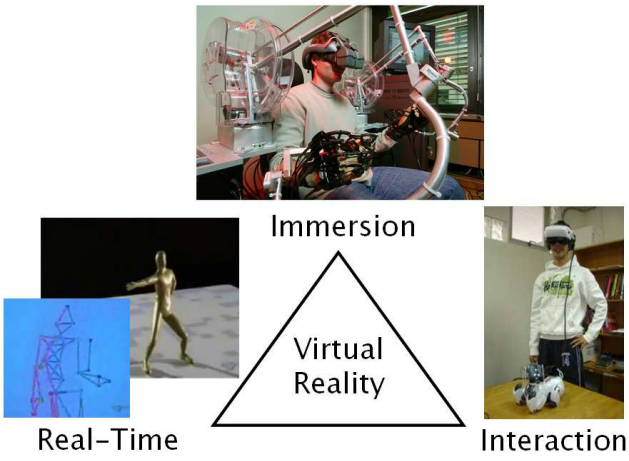

Figure 1. The VR triangle: main components of a Virtual Reality application.

2. The complexity of a VR application

When we talk about a Virtual Reality application, we are referring ourselves to an interactive Virtual Environment. Virtual Environments (VEs) commonly involve the use of 3D graphics, 3D sound and real-time interaction for creating a simulation. A Virtual Environment can be defined as: an environment which is partially or totally based on computer generated sensory input. Sensory information used to create a VE addresses three main types of senses: sight, hearing and touch (haptics).

The creation of VEs is a high complexity task requiring diverse areas of expertise, which may range from networks to psychology. Developing VE systems is a very expensive task in terms of time, financial and human resources. VEs can be applied in a broad range of areas, such as scientific visualization, socializing, training, psychological therapy, gaming. Such diversity of applications produces a set of requirements that make it very difficult, if not impossible, to build a single system to fit all needs. Traditionally, VR application are monolithic systems, highly optimized to a particular application, without possibility of reusability with a different purpose. In this sense, we could consider that there is nothing such as a standard VR application.

According to Oliveira et al. [19], the problem of lack of reusability -and standardization- is due to the current trend in the VE community: developing a new VE system for each different application. The "reinventing the wheel" and "not invented here" syndromes limit the innovation and delay the use of VEs in wider areas for the general public.

We identify the low level of reusability and adaptability of Virtual Environments as important problems that could be tackled by means of standardization: strategies, algorithms, specification languages and development frameworks that could be adopted by the community of researchers and developers.

We will now present an overview of development frameworks and specifications that have reached some success in the development of VR applications. This way we expect to identify available alternatives to conform a standard set of tools for VR applications development.

3. Development frameworks for VR applications

We start this overview of technologies and specifications for VR development with a revision of toolkits, integrated systems and frameworks aimed at easing the process of design, development and use of Virtual Reality applications.

The first systems we consider are those which try to group all the required tools in a single package: monolithic systems. Monolithic systems such as DIVE [11], MASSIVE [4], NPSNET [16], SPLINE [1], dVS/dVISE [5] amongst others have proliferated since last decade due to the lack of system flexibility to implement a particular VR application [19].

The introduction of more modular architectures led to the emergence of toolkits such as WorldToolkit [24], Avocado [28], VR Juggler [2], VHD++ [22], Virtools [30], etc. These software suites have different degrees of flexibility. Frameworks like VHD++ differentiate from the others due to its specialized skills on a particular domain, e.g. Virtual Humans simulation technologies. All of them are based on a hierarchical representation of the virtual environment: a scene graph. It is interesting then to look deeper into systems or specifications implementing the concept of *scene graph*.

3.1. Scene Graph - based systems

A scene graph is an abstract logical access structure used to represent objects composing the environment (scene data) and the relationships between them. Scene graphs are often confused with data structures used to do visibility culling or collision queries such as octrees/BSP/ABT/KdT, etc. Scene graphs are used to connect game-rules, physics, animation and AI systems to the graphics engine.

Popular implementations of scene graph programming interfaces (APIs) include: Cosmo3D (SGI), Vega Scene Graph [18], Java3D [27], OpenSceneGraph [20] and OpenGL Performer [26]. All of them were designed to create real-time visual simulations and other performance-oriented 3D graphics applications. OpenGL Performer is a commercial toolkit which evolved from Open Inventor [25].

Open Inventor, is considered as the archetypical example of scene graph library. It presents an object oriented programming model based on a 3D scene database (scene graph) that provides a higher layer of programming for OpenGL.

Java 3D gained popularity as the main scene graph-based API for developing 3D applications with Java. It is frequently used for developing web-based applications enhanced with real-time 3D graphics [13],[3],[33]. It is also a very representative example of a scene graph-based 3D toolkit.

As far as we know, all of the available scene graph implementations are based on a hierarchical spatial representation of the objects in the scene, e.g. a terrain contains a house, inside the house there is a person who is holding a hammer in her right hand.

Usually, the semantic information that is encoded in the scene graph corresponds mainly to visualization aspects: geometry to draw, and associated effects such as a sound sample. Only basic relationships between objects can be specified, e.g. smart objects [14] (simple tools and objects such as a hammer or a drawer) which contain information

describing how they can be grasped by a virtual human and defining a pre-determined behavior to perform.

Efforts aimed at enhancing the adaptability and reusability of VE applications and entities within them, have focused on designing software component frameworks for managing the resources and building blocks of a VE system. Such is the case of the Java Adaptive Dynamic Environment (JADE) [19] which permits dynamic runtime management of all components and resources of a VE system. While this kind of initiatives are successful in terms of providing reusability an interoperability at source code level, they do not address the fundamental problem of reusing the virtual entities that participate in the VE application. The use of scene graphs as hierarchical spatial representations is not questioned. As a result, source code implementing animation/visualization/interaction algorithms can be reused to some extent. But the knowledge associated to a virtual entity remains difficult to reuse. The following section deals with the problem of expressing and reusing knowledge associated to virtual entities as the fundamental building blocks of a VR application. The discussion allows for introducing several language specifications for Virtual Environments, which would constitute a core part of standard VR.

4. Language specifications for standard VR

Multi-user virtual environments implemented across the web allow many users to interact with the virtual world and each other in a seamless manner. These constitute one of the most popular VR applications nowadays. The current -de facto- standard for embedding virtual worlds in the Internet is VRML 2.0. The VRML (Virtual Reality Modeling Language) specification is widely accepted as a standard language for designing virtual environments, including mainly the geometry and some interaction functionalities (navigation modes, basic animation, etc.).

Common VRML implementations lack of scalability and functionalities for efficient interaction in the context of shared VEs. Solutions are provided as custom applications usually based on C++ and OpenGL to overcome disadvantages of default VRML implementation [15].

Another missing functionality in VRML is the capability to accessing databases and further parameterization. Extensions and variations of VRML have been proposed to overcome such limitations. This is the case of X-VRML. X-VRML is a high-level XML-based language that overcomes the main limitations of the current virtual reality systems by providing convenient access to databases, object-orientation, parameterization, and imperative programming techniques. Applications of X-VRML include on-line data visualization, geographical information systems, scientific visualization, virtual games, and e-commerce applications such as virtual shops [31].

Although VRML has made viewing 3D content on the web possible, remotely accessing large and complex 3D worlds requires a large amount of bandwidth. In the absence of such bandwidth users will suffer substantial latency in receiving the entire scene before they are able to view and interact with it. Streaming the 3D content and displaying the parts currently available while allowing users to interact with and navigate through the world reduces the time users have to wait in order to become involved in the virtual scene and hence improves their experience. The MPEG-4 standard has been used for streaming 3D worlds and the corresponding animation over the web while allowing users navigation and manipulation of the content, as it becomes available [12].

VRML is still quite popular amongst VR developers and researchers. However, the Web 3D Consortium, an international community composed by industrial and academic partners has been working on the successor of VRML since a few years from now: X3D. X3D[32] is a royalty-free open standards file format and run-time architecture used to represent and communicate 3D scenes and objects using XML. It is an ISO ratified standard that provides a system for the storage, retrieval and playback of real time graphics content embedded in applications, all within an open architecture to support a wide array of domains and user scenarios.

X3D has a rich set of modular features that can be tailored for use in engineering and scientific visualization, CAD and architecture, medical visualization, training and simulation, multimedia, entertainment, education, and more.

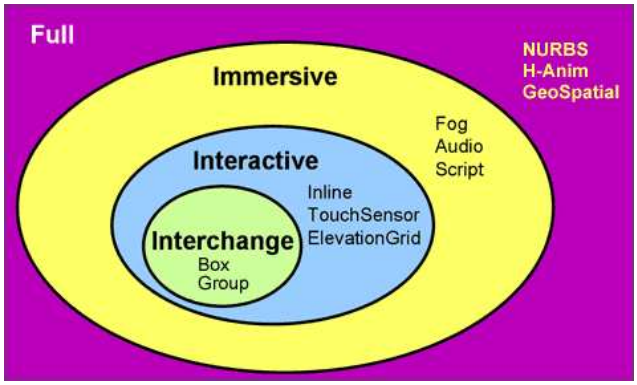

Figure 2. X3D Baseline Profiles.

The modular architecture of X3D allows for layered "profiles" that can provide 1) increased functionality for immersive environments and enhanced interactivity or 2) focused data interchange formats for vertical market applications within a small downloadable footprint composed of modular blocks of functionality ("Components"), that can

be easily understood and implemented by application and content developers.

A component-based architecture supports creation of different "profiles" which can be individually supported. Components can be individually extended or modified through adding new "levels", or new components can be added to introduce new features, such as streaming. The X3D Baseline Profiles are (see figure 2):

- **Interchange** is the basic profile for communicating between applications. It support geometry, texturing, basic lighting, and animation. There is no run time model for rendering, making it very easy to use and integrate into any application.
- **Interactive** enables basic interaction with a 3D environment by adding various sensor nodes for user navigation and interaction (e.g., PlanseSensor, TouchSensor, etc.), enhanced timing, and additional lighting (Spotlight, PointLight).
- **Immersive** enables full 3D graphics and interaction, including audio support, collision, fog, and scripting.
- **Full** includes all defined nodes including NURBS, H-Anim and GeoSpatial components.

Despite the novel functionalities and extensibility offered by X3D, this specification is still based on the concept that Virtual Environments are composed by a large number of low-level geometric objects that lack any semantic description. Such situation prevents advanced uses of the data contained inside the environments, such as selection and extraction of semantic objects or advanced queries that refer to high-level properties of the environment.

Specification languages have been designed with the goal of annotating 3D environments using the descriptive capabilities of MPEG-7 standard [17],[23]. However, MPEG-7 allows only for a rather low-level description of the content: mostly geometrical information dependent on the scene.

Incorporating additional knowledge and information -semantics- into the building blocks of a VR application is one of the most recent approaches towards the reutilization and standardization of Virtual Environments. In the next section we provide further details on the topic of incorporating semantics to Virtual Environments.

5. Semantics: an approach to VR standardization

Some of the most recent research on Virtual Environments development is targeted at the definition of scene-independent ontologies that can be useful in different situations (e.g., 3D world validation, semantic search through a set of worlds, etc.).

According to Gruber [6], an ontology is a formal specification of a shared conceptualization. Virtual Environments are complex entities composed by virtual entities such as virtual characters and objects with well defined features, and functionalities. Concepts and techniques related to the creation and exploitation of virtual entities are shared by the research community. Efforts are targeted at unifying such concepts and representing them in a formal way. A formal representation refers to the fact that Virtual Environment representations and their associated semantics shall be both human and machine readable -this is achieved by means of an XML-based representation.

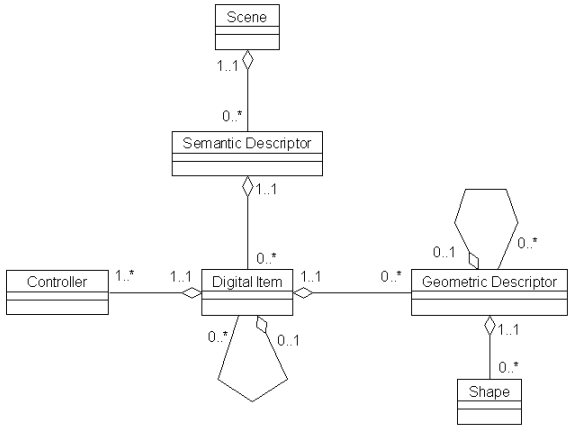

Figure 3. Semantic representation of an interactive virtual environment.

Pittarello et al.[21] propose an approach for associating semantic information to 3D worlds based on the integration of two web standards: the X3D language and the semantic web. The approach is based on the definition of scene-independent ontologies and semantic zones that complement the role of semantic objects, giving a complete description of the environment. Semantic information can be extracted from an X3D document and used to generate the associated ontology providing a high-level and multilevel textual description of a 3D environment.

Ontological principles are well recognized as effective design rules for information systems [7], [29]. This has led to the notion of "Ontology-driven information systems" which covers both the structural and temporal dimensions [7]. The structural dimension concerns a database containing the information (semantic descriptors). The temporal dimension is related to the interface (visual programming) that gives access to the information at run-time. We consider that such an ontology-driven information system can support the creation and exploitation of Virtual Environments and promote their standardization.

Associating semantic information to the components of a virtual environment has proved to be useful in terms of component reuse, content adaptation, etc. In [10], Gutiérrez et al. defined an object representation based on the semantics and functionality of interactive digital items - virtual objects- within a Virtual Environment (VE), see figure 3.

Every object participating in a VE application is a dynamic entity with multiple visual representations and functionalities. This allows for dynamically scaling and adapting the object's geometry and functions to different scenarios. The semantic model provides a way to specify alternative geometric representations for each entity and a range of functionalities. The entities are called digital items and can be not only virtual objects to be rendered as part of the scene, but also independent user interfaces and controls for animation or interaction. In [8], the semantic model presented in [10] was complemented with an ontology of objects that allowed for expressing the relationships between interaction devices and virtual entities in a VE (see figure 4).

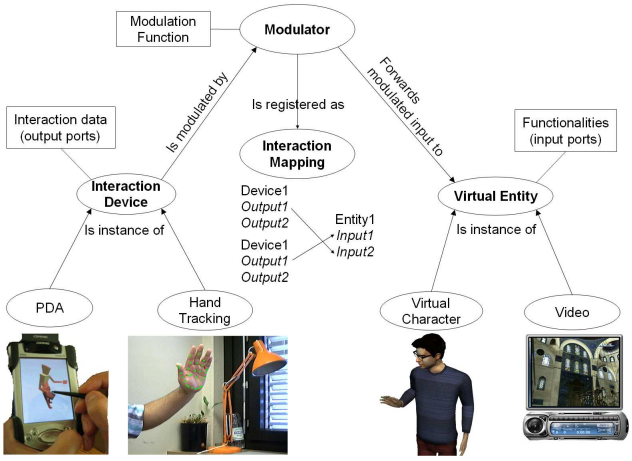

Figure 4. Ontology for interactive VEs: elements involved in a multimodal interface.

The work presented in [9] builds upon the acquired experience and focuses on a single type of virtual entity: Virtual Humans. A diagram of the ontology is presented in Figure 5. The following are the concepts defined to express in a formal way the information and knowledge associated to Virtual Humans:

Geometry: The geometry is the physical visual representation of the Virtual Human and is composed of two parts, primary: body shape, and secondary: accessories, garments, etc. This common class contains the geometric properties: number of vertex, edges, scale, material and texture.

Animation: Virtual Human's animation should distinguish between facial and body animation because the way of animating each part is different. Body animation can be KeyFramed or Motion Captured. There are standardized formats to specify animation: MPEG-4, VRML, etc. In order for an animation to take place, the character should have a skeletal structure. This structure is defined in the Structural Descriptor. This structure is based on the H-Anim specification, the de facto standard adopted by most of the animation formats.

Morphology: Morphological Descriptor contains information like: Age, weight, height, gender.

Behavior: Individual Descriptor and Behavior controller are for describing the behavior. Behavior controllers are algorithms that drive the behavior of the character considering the emotional state and its individuality.

Figure 5 presents how the concepts are connected to each other in the ontology. The concepts that are in a circle are subclasses of the concept Resource, in the bottom of the diagram. As a consequence, those concepts have inherited the properties of the Resource: author, version, file, etc. This means that the Resources are the product that the user wants to get when searching for a component. Some of the resources can be found in separated files or more than one in the same file. In the case of behavior controller the user can obtain an algorithm.

The examples we have presented show the growing interest of the research community about finding a standard more expressive than VRML and its derivatives (X-VRML, X3D, etc) that allows for specifying not only the visuals and basic interaction, but also the knowledge associated with a Virtual Environment and its components. In this sense, the notion of semantic Virtual Environments: a way to understand VR applications as a set of independent entities that can be searched, reused and represented in a variety of ways, constitutes a good effort towards a standard specification language to define and handle the components of a VR application.

6. Conclusions

We have presented an overview of the development frameworks, specifications of modeling languages and current research trends concerning the design, development and management of Virtual Environments and their components. A fast review of the specialized literature clearly shows that most of the VR applications nowadays are based on one of the technologies and specifications mentioned in this article. In particular, the preponderant standard for modeling and specifying basic interaction is VRML and or its successor: X3D. Most of current development frameworks such as Virtools, or modeling tools such as 3DS MAX, Maya or Blender3D are fluent in one or both formats and many others which are proprietary or less popular.

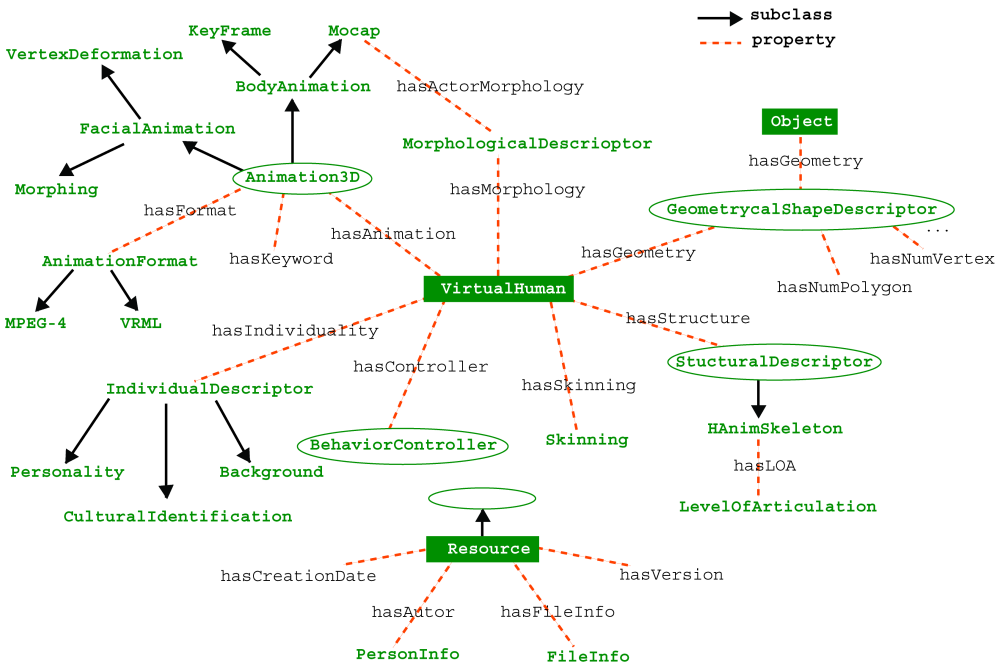

Figure 5. Main components of an Ontology for Virtual Humans.

From the point of view of modeling and animation formats, the clear trend is towards the adoption of X3D as both a defacto and official ISO standard. X3D's modular nature allows it to cover and interoperate with other specifications such as MPEG-4, the extensibility provided by its XML nature makes of X3D a robust platform for building up a solid and continuously evolving standard for VR applications.

Besides languages for specifying modeling, animation and interaction such as X3D. We have also shown that current research is being focused on the development of ontology-based systems that allow for incorporating and exploiting the semantics associated to a VE and its components. The combination of languages such as X3D and semantic web technologies is making it possible to create true repositories of virtual entities which are fully searchable and allow for unprecedent capabilities for reusing and re-adapting components such as 3D models and their animation algorithms, interaction devices, etc.

Coming back to the VR triangle, current research directions and industry developments are converging towards a standard specification language capable to express:

- A flexible representation -content adaptation- for better *immersion* in a variety of contexts: PC, video consoles, mobile devices.
- Possibility for handling a variety of devices for communicating with the Virtual Environment: *interaction* (see X3D interactive profiles [32] and ontology for interaction devices [8])
- All the above having in mind *real-time* performance in any context, be it a web-based application on a mobile device or a high-definition scene in the latest game console (content adaptation).

References

[1] D. Anderson, J. Barrus, J. Howard, C. Rich, and R. Waters. Building multiuser interactive multimedia environments at merl. *IEEE Multimedia*, 2(4):77–82, Winter 1995.

[2] A. Bierbaum, C. Just, P. Hartling, K. Meinert, A. Baker, and C. Cruz-Neira. Vr juggler: A virtual platform for virtual reality application development. In *Proceedings of the Virtual Reality 2001 Conference (VR'01)*, pages 89–96. IEEE Computer Society, 2001.

[3] R. Dörner and P. Grimm. Three-dimensional beans–creating web content using 3d components in a 3d authoring environment. In *VRML '00: Proceedings of the fifth symposium on Virtual reality modeling language (Web3D-VRML)*, pages 69–74. ACM Press, 2000.

[4] C. Greenhalgh, J. Purbrick, and D. Snowdon. Inside massive-3: flexible support for data consistency and world structuring. In *CVE '00: Proceedings of the third international conference on Collaborative virtual environments*, pages 119–127. ACM Press, 2000.

[5] C. Grimsdale. dvs-distributed virtual environment system. In *Proceedings of Computer Graphics'91, London, UK, Bleinheim Online*, pages 163–170, 1991.

[6] T. Gruber. The role of a common ontology in achieving sharable, reusable knowledge bases. In *Proceedings of the*

Second International Conference on Principles of Knowledge Representation and Reasoning, pages 601–602, 1991.

[7] N. Guarino. Formal ontology and information systems. In *Proceedings of FOIS 98, (Trento, Italy, June, 1998). IOS Press*, pages 3–15, 1998.

[8] M. Gutierrez, D. Thalmann, and F. Vexo. Semantic virtual environments with adaptive multimodal interfaces. In *Proceedings of the 11th International Conference on Multimedia Modelling (MMM2005)*, pages 277–283, 2005.

[9] M. Gutierrez, D. Thalmann, F. Vexo, L. Moccozet, N. Magnenat-Thalmann, M. Mortara, and M. Spagnuolo. An ontology of virtual humans: incorporating semantics into human shapes. In *Proceedings of Workshop towards Semantic Virtual Environments (SVE05), March 2005*, pages 57–67, 2005.

[10] M. Gutierrez, F. Vexo, and D. Thalmann. Semantics-based representation of virtual environments. *International Journal of Computer Applications in Technology (IJCAT) Special issue on "Models and methods for representing and processing shape semantics"*, 23(2/3/4):229–238, 2005.

[11] O. Hagsand. Interactive multiuser ves in the dive system. *IEEE Multimedia*, 3(1):30–39, Spring 1996.

[12] M. Hosseini and N. D. Georganas. Mpeg-4 bifs streaming of large virtual environments and their animation on the web. In *Web3D '02: Proceeding of the seventh international conference on 3D Web technology*, pages 19–25, New York, NY, USA, 2002. ACM Press.

[13] S. Huang, R. Baimouratov, and W. L. Nowinski. Building virtual anatomic models using java3d. In *VRCAI '04: Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, pages 402–405. ACM Press, 2004.

[14] M. Kallmann and D. Thalmann. Direct 3d interaction with smart objects. In *VRST '99: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 124–130. ACM Press, 1999.

[15] I. Kotziampasis, N. Sidwell, and A. Chalmers. Seamlessly integrated distributed shared virtual environments. In *SCCG '04: Proceedings of the 20th spring conference on Computer graphics*, pages 138–147, New York, NY, USA, 2004. ACM Press.

[16] M. R. Macedonia, D. P. Brutzman, M. J. Zyda, D. R. Pratt, P. T. Barham, J. Falby, and J. Locke. Npsnet: a multi-player 3d virtual environment over the internet. In *SI3D '95: Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 93–94. ACM Press, 1995.

[17] J. Martinez. "MPEG-7 Overview", ISO/IEC JTC1/SC29/WG11N5525, Pattaya, March 2003.

[18] MultiGen-Paradigm, Inc. Vega Prime, http://www.multigen.com/products/runtime/vega_prime/index.shtml, 2005.

[19] M. Oliveira, J. Crowcroft, and M. Slater. An innovative design approach to build virtual environment systems. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003*, pages 143–151. ACM Press, 2003.

[20] OSG Community. OpenSceneGraph http://www.openscenegraph.org, 2005.

[21] F. Pittarello and A. D. Faveri. Semantic description of 3d environments: a proposal based on web standards. In *Web3D '06: Proceedings of the eleventh international conference on 3D web technology*, pages 85–95, New York, NY, USA, 2006. ACM Press.

[22] M. Ponder, B. Herbelin, T. Molet, S. Schertenlieb, B. Ulicny, G. Papagiannakis, N. Magnenat-Thalmann, and D. Thalmann. Immersive vr decision training: telling interactive stories featuring advanced virtual human simulation technologies. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003*, pages 97–106. ACM Press, 2003.

[23] P. Salembier. Overview of the mpeg-7 standard and of future challenges for visual information analysis. *EURASIP Journal on Applied Signal Processing*, 4:1–11, 2002.

[24] Sense8 Corporation. WorldToolkit Reference Manual – Release 7. Mill Valley, CA, 1998.

[25] Silicon Graphics, Inc. Open Inventor, http://oss.sgi.com/projects/inventor/, 2005.

[26] Silicon Graphics, Inc. OpenGL Performer, http://www.sgi.com/products/software/performer/, 2005.

[27] Sun Microsystems, Inc. The Java 3D API Specification, http://java.sun.com/products/java-media/3D/forDevelopers/J3D_1_3_API/j3dguide/Concepts.html, 2002.

[28] H. Tramberend. Avocado: A distributed virtual reality framework. In *VR '99: Proceedings of the IEEE Virtual Reality*, pages 14–21. IEEE Computer Society, 1999.

[29] M. Uschold and M. Gruninger. Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11(2):93155, February 1996.

[30] Virtools SA. Interactive 3d content development tools: http://www.virtools.com.

[31] K. Walczak and W. Cellary. Building database applications of virtual reality with x-vrml. In *Web3D '02: Proceeding of the seventh international conference on 3D Web technology*, pages 111–120, New York, NY, USA, 2002. ACM Press.

[32] Web 3D Consortium. X3D Specifications, http://www.web3d.org/x3d/specifications/, 2006.

[33] C. Yimin, Z. Tao, W. Di, and H. Yongyi. A robot simulation, monitoring and control system based on network and java3d. In *Proceedings of the 4th World Congress on Intelligent Control and Automation*, pages 139–143, 2002.