Animation and Visualization - Current Status and Trends

Michael Gervautz, Dieter Schmalstieg

Institute of Computer Graphics, Technical University of Vienna,
Karlsplatz 13/186/2, A-1040 Vienna, Austria

email: gervautz | dieter@cg.tuwien.ac.at

Abstract. In the last years, new input and output devices have noticeable impact on Computer Animation and Visualization. Head mounted displays, data gloves, and other devices allow to built immersive user interfaces. The number of virtual reality applications has dramatically increased. This new kind of systems generates not only new possibilities but also new problems to be solved. Among them are real-time rendering, motion estimation and prediction, animation techniques like autonomous actors, new three-dimensional interaction techniques, and architectures that embody clear concepts and software engineering methodologies for such kind of systems.
We analyze a number of existing systems that illustrate what we believe are core concepts and difficulties to be overcome. and we attempt to formulate the state of the art in this field.

1. INTRODUCTION

Computer Animation and Visualization have been strongly influenced by technological advances such as graphics hardware and devices like head-mounted displays and body trackers. Traditional symbolic user interfaces devices limit the amount of information exchange between user and machine. We interact with the real world through highly developed skills such as our visual system. Providing an interface that involves these skills rather than artificially created interaction techniques has the potential of dramatically increasing the usability of the medium computer.

It appears that some key technologies have recently become cost-effective, and so virtual reality is extremely popular inside and outside the scientific community. To avoid confusion and expectations that come from market hype, researchers have introduced the term *virtual environments* (VE). Virtual environments should be interactive, three-dimensional simulations. They require a number of techniques to be incorporated into a single software system:

- **Multi-sensory Interaction**: The user must be able to interact with the environment in real-time. Although graphics alone are a major factor in VEs, the desired high *level of immersion* is boosted by integrating devices that interact

with the other human senses, both for input and output. Output includes visual stereoscopy, 3D spatial audio, and haptic display. Input includes hand, head, and body tracking and voice commands.

- **Real-time 3-D rendering:** Sensual response from the system must be immediate. Even small delay (in the order of 0.1sec) or inadequate smoothness of the system's output easily destroys the feeling of immersion and therefore the goal of the virtual environment. This places very high demand on the rendering hardware and software that is by no means adequate for high-quality real-time rendering yet.

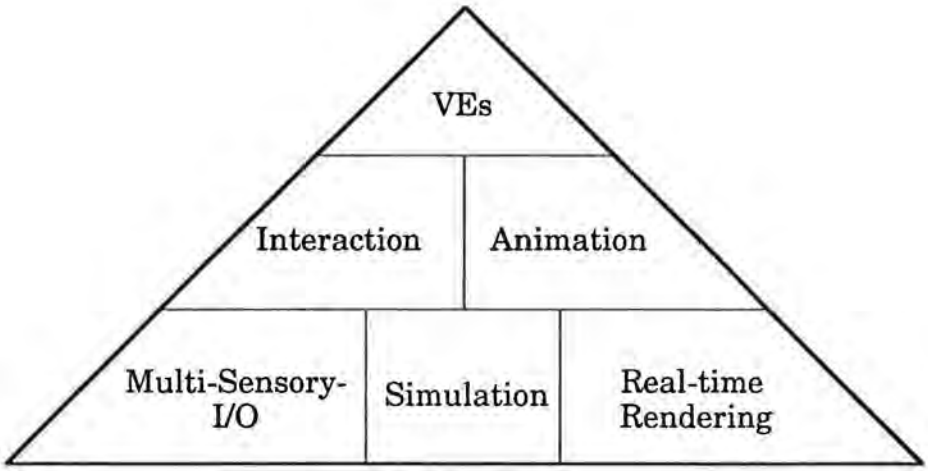

Figure 1. Virtual environment building blocks

Fig. 1 illustrates the relation of the major building blocks of virtual environment software. Most people associate the term "Virtual Reality" with exotic I/O devices such as head-mounted displays and data gloves. As can be seen, this is just a very small part on the lowest level of the architecture. The elements that are really significant are:

- **Simulation:** To step beyond interactive graphics walk-through systems, serious simulation engines are necessary that allow to fill the virtual world with meaningful content and create productive applications.
- **Animation:** Traditional animation techniques have to be adapted to fulfill the real-time need. Agents and Autonomous Actors are developed acting independent within the environment.
- **Interaction:** A three-dimensional world needs other interaction paradigms than a two-dimensional user interface on a flat screen. Tools like direct manipulation, gestures and simulated tools are used rather than pop-up menus, windows, and other two-dimensional gadgets because the switching between 3D-environment and 2D-user-interface disturbs the feeling of immersion.
- **Virtual Environment Systems:** Virtual environment software combines real-time simulation, animation, and interaction, and is therefore extremely de-

manding in the computational resources. New solution have to be found to graphics and system software to fulfill these demands.

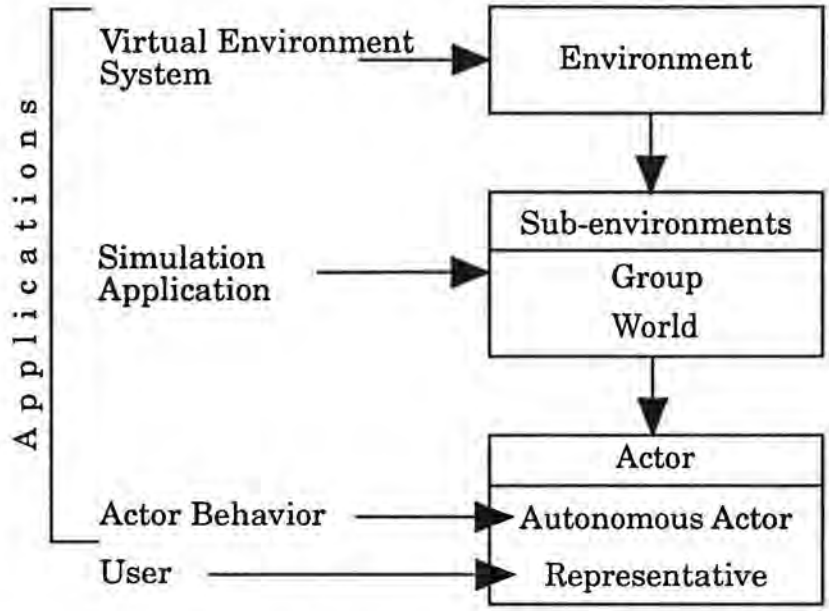

Figure 2: Terminology of Virtual Environment Systems

To think about VE in a systematic manner, it is necessary to separate *conceptual* issues from implementational considerations. Here we specify a conceptual terminology that we use in the following discussion.

- **Virtual environment:** A virtual environment is composed of active units that communicate with each other in a three-dimensional space, that can be examined by a user.
- **Actor:** The basic unit the environment is composed of is called an actor. An actor represents something from the real (or a fictional) world. The actor is visible and may change its state, in particular its visual representation, over time.
- **Representative:** In the VE, the user is represented by a special actor - a representative - that stands in for the user. The user interacts with the environment through the representative.
- **Autonomous actor:** Actors that are not controlled by users are instead controlled by some piece of software. Because they can act according to a synthetic behavior, they are called *autonomous*.
- **Actor group:** A virtual environment may be decomposed into groups. Actors are grouped to allow a common simulation application to control their behavior.
- **Virtual world:** If the grouping is made by a region (e.g.: all actors in one room) this group is called *virtual world*. Connections between worlds can be geometrical or conceptual links. The latter are called *portals*.
- **Application:** The content of the virtual world is not determined by the actors alone. Applications are needed that control aspects of the VE. An application may exercise control on the level of a world or group, or on a specific actor.

There is the potential of conflicts if the control commands of multiple applications are contradictive. This is a fundamental unsolved problem also of computer animation.

- **Object**: One of the most abused term is *object*. Every unit of discussion may be called so. We therefore use the term "object" only in the strict object-oriented sense, denoting an encapsulated union of attributes and functionality. The units that "live" in a VE are sometimes called objects [Zelt89], sometimes entities [Bric90] [Bric94], artefacts [Snow94] or actors [Ghee94]. For our discussion, we choose *actor*.

2. STATE OF THE ART

2.1. Multi-sensory Interaction

New technologies make it possible to generate acceptable quality images from VEs, but the direct feedback from input to the output is difficult to provide. There are many problems like a delay between the real event, which is generated by an input device like a motion tracker and an appropriate screen update, jittering of images, and non-constant frame update rates that cause motion sickness. Human beings are not used to such an artificial generated, bad feedback [Helm93] which is unnatural, too slow, and different from their expectations. Quite early it became clear that hardware improvement itself will not be able to solve these problems and software methods must be employed.

Head tracking is one of the main problems. The output of the head-tracking process is used to generate a properly positioned picture on the output device, e.g. a head mounted display (HMD). Even the slightest error produces noticeable output changes. Moreover, one must consider that the computer needs time to read the tracker measurements, set the new camera position and perform rendering. Because of this, the picture is presented with some delay which disturbs the feeling of immersion and causes motion sickness. Therefore a head movement prediction needs to be applied.

Different approaches have been attempted to cope with this problem. The best results up to now are achieved by predictors based on the Kalman filtering technique [Azum94][Frie92][Lian93]. There are also some other systems that achieve relatively good results but they are very hardware-intensive. They use gyros and accelerometers to measure position and orientation of the head and extremely fast, specialized rendering engines [Azum94]. A newer approach is based on a combination of motion estimation, prediction, and image deflection [Mazu95].

2.2. Real-time Rendering

Up to now, the aim of rendering and visualization research was to find fast methods to display three-dimensional scenes with highest possible quality. Sophisticated illumination models where proposed to model the laws of light energy transfer for static scenes as well as for animated sequences (see [Glas95] for a summary). This goal is no longer valid for Virtual Environments. Images have to be produced with constant frame rates and within the time the motion predictor was tuned for. Therefore, new rendering algorithms are needed [Funk93]

[Heck94], which reduces the quality of the image according to rendering time and complexity of the scene. Actors are rendered with different levels of detail, different texture mapping techniques, and different shading model, in order to achieve constant rendering time.

Virtual environments tend to be very large consisting of many many actors. Even if these actors are static, it is not very efficient to pass them all to the rendering process. Even the fastest rendering hardware and the smartest algorithms can only display a certain amount of data in a given time. Powerful pruning [Tell91][Mill93] of the potential visible parts of the environment has to be done before rendering.

Equipped with motion prediction pruning and image quality reduction constant time rendering is manageable even for complex environments.

2.3. Animation and Autonomous Actors

Animation techniques can be distinguished based on the fundamental model such as functional animation, key-framing (inbetweening), procedural animation (scripting), dynamic simulation and goal oriented animation [Watt92]. All these techniques can be combined into a single integrating concept [Gerv93][Gerv94] which can be adopted easily for VEs. A VE consists of functionally independent units, called actors, each of which can be driven by a different animation technique. A common interface between actors exists to provide a means for communication. Autonomous actors can be integrated in this framework easily using rules for specific behavior and messages to communicate with the environment. This is the base on which autonomous actors can be built.

Representatives stands for users in the virtual environment. Although not required, it may be convenient if representatives look and move like human beings. The field of human animation has been a main research topic in the last eight years. There has been significant progress in both modeling and animation of human figures (see e.g. [Badl93]), however the timing requirements of virtual environments are often to tight for human animation.

As for high-level modeling, geometric constraints are an important construct in computer animation [Barz88]. Linked figures, mechanical machines, kinematic chains, even a human skeleton model can be built, using constraints to combine some actors leaving only some of the degrees of freedom unconstrained, thus forming joints with different functionality.

2.4. Collision detection

Early physically based animation systems included collision detection and response to simulate natural behavior of rigid bodies and linked kinematic chains [Bara93]. In virtual environments collision detection is used beside physics simulation also for the avoidance of the penetration of two actors. In most cases no physically correct response is implemented. Simplifications like stopping the motion of both colliding actors are used to avoid high computational costs. Such techniques satisfy the necessity of preventing actors from interpenetration without having the overhead of computing a complete physically based simulation. Although there exists some algorithms for collision detection in linear time

[Lin92][Bara93], for a very high number of actors, like they exist in virtual environments, it can not be done in real-time.

2.5. Virtual Environment Systems

In this section we attempt to give an overview of virtual environment systems. For the sake of brevity, we limit ourselves to those systems that contribute important software architecture concepts.

2.5.1. Existing Systems

An early system with a focus on animation rather than interaction is BOLIO [Zelt89]. BOLIO uses geometric constraints as grouping mechanism. The MR toolkit [Shaw92] was the first system that introduces the idea of *decoupled simulation*, that is the functional decomposition of the system into independent processes. RB2 (Reality Built for Two) [Blan90] appears to be the first widely recognized multi-participant VE. VUE [Appi92][Code92] introduces a sophisicated concept of a dialogue for th specification of message flow in the system. VB2 [Gobe93] introduces an architecture with strong features in physically based animation. These are realized by a constraint solver. dVS [Ghee94], one of the most recognized commercial system, models the VE as a database shared by independent processes, controlled by a central director. SIMNET [Calv93], NPSNET [Mace94], and VERN [Blau92], are a family of large-scale combat-training oriented simulations supporting a large number of participants, using the standardized protocol DIS. ALICE [Paus93] is a rapid-prototyping system aimed at the quick implementation or modification of simulations by both skilled and novice users. DIVE [Carl93] is a distributed system where users and applications make concurrent updates to a database that is shared over the network. BRICKNET [Sing94] consists of a network of servers that allow clients to connect, allowing them to share information across servers by leasing out objects to other clients. VEOS [Bric94] has a very rigorous structure of hierarchical actors with arbitrary communication. It supports multiple simulation applications and multiple users. AVIARY [Snow94] supports actors modeled as concurrent processes that communicate by messages over the network. It tries to resolve the conflict of actor application and simulation application by providing actor properties on a world basis. VRML [Ball95] is a very recent attempt to bring 3-D graphics to an existing world-wide net, namely the WWW.

2.5.2. Discussion

Virtual Environment software can be characterized by two criteria: Level of distribution and level of flexibility. Existing systems can be assigned to one of the following groups:

Level of distribution

Virtual environment software requires real-time simulation and rendering and is therefore extremely demanding in the computational resources. It is therefore logical to employ some kind of concurrent computation model to resolve the computational bottle-neck; however, system complexity increases as well. Some models of distribution may be distinguished:

- **Single-threaded**: polling of user interface devices, simulation and rendering are all performed in a single loop.
- **Single-user, multi-threaded**: The key idea here is to assign parts of the application to dedicated threads (e.g. simulation, rendering) that execute concurrently and synchronize by interprocess communication (shared memory, low-level network protocols). Each dedicated thread maintains its own loop, so the update rates of the threads are independent.
- **Multi-user**: Recent systems allow several human participants to be present simultaneously and to interact with each other. Since every user has to have a console of his own, a local area network is necessary.
- **Multi-user, geometrically disperse**: In the large, multi-user environments have a somewhat different quality. Large number of user (several hundred) imply a potential low-bandwidth wide-area network that is quite different to handle than a simple close-coupled LAN.
- **Multi-world**: In addition to support for multiple users, multi-world environments implement "parallel universes", that is, more that one world is simulated at a time. The multiple worlds may have completely different contents, rules, and interaction styles.

Level of flexibility

- **Monolithic application**: Run-time framework, simulation application and interaction paradigm are tightly integrated to achieve acceptable performance for small worlds.
- **Tool-kits**: A number of modules (often a class hierarchy) is provided that provides the programmer with high-level tools (such as an efficient rendering engine, intelligent device-drivers for 3D-input or a network management module) from which to build an environment system.
- **Fixed-feature environment**: Especially in the low-cost market, virtual environment come in the form of ready-made applications. The feature set is fixed, although usually there is some kind of scripting mechanism. It is more or less predetermined how the user may manipulate the world.
- **Programmable environment**: Somewhat more powerful is an architecture that allows autonomous actors to be programmed, specifying all their properties and their dynamic behavior.
- **Dynamic simulation environment**: Still more powerful is an environment that allows the creation and manipulation of actors and the configuration worlds or groups *at runtime*. Ideally, the virtual environment features not only dynamic creation of actors but also accepts the specification of new actor *types*. Users are able to log into this environment and leave it after a time. The data in the environment is persistent.

Using these two classification schemes the above described systems can be compared (see fig.3). Distribution and flexibility are neither completely orthogonal, nor are they an exhaustive characterization, but they form a taxonomy that describes most aspects of the current systems reasonably well.

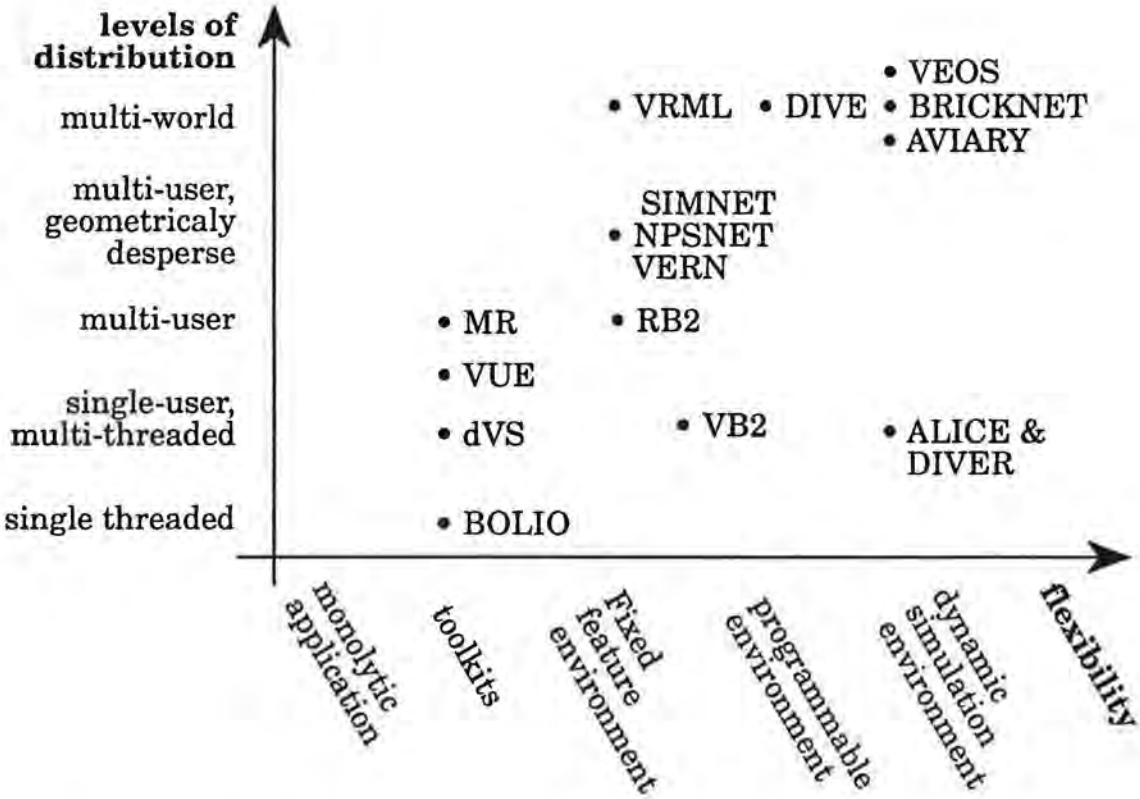

Figure 3: A classification of VE-systems based on flexibility and levels of distribution

3. CONCLUSION

We have presented the current state of the art in virtual environment systems. Virtual environments are a relatively new field and pose high demands on current hardware and software. Multi-sensory input an output, high-quality 3-D rendering, three-dimensional interaction paradigms, flexible dynamic simulation, and high level animation must be combined at real-time rates. Distributed heterogeneous networks are needed to deal with the high computational load and support multiple simultaneous users.

Virtual environments combine techniques from visualization, commuter animation, networks, modeling, distributed computation, user interfaces and human-computer interaction.

State of the art systems are both highly concurrent and flexible. They support multiple users on a wide-area networks and multiple worlds. Participants can dynamically enter and leave the environment and migrate between worlds. New actors and actor types can be defined at runtime. Arbitrary behavior can be attributed to actors.

As a conclusion, a system that meets all or most of the issues that are addressed in this paper should be well-equipped for tomorrow's virtual reality applications.

REFERENCES

[Appi92] P. A. Appino, J. B. Lewis, L. Koved, D. T: Ling, D. A. Rabenhorst, C. F. Codella: An Architecture for Virtual Worlds. Presence, Vol. 1, No. 1, pp. 1-17 (1992)

[Azum94] R. Azuma, G. Bishop: Improving Static and Dynamic Registration in an Optical See-through HMD, ACM Siggraph, pp. 197-204, 1994

[Badl93] N. Badler, N. Magnenat-Thalmann, D. Thalmann, D. Terzopoulos: Recent Techniques in Human Modeling, Animation and Rendering. SIGGRAPH'93, Course notes, No. 80 (1993)

[Ball95] G. Ball, A. Parisi, M. Pesce: VRML Draft Specification 1.0. Technical Report, http://www.eit.com/vrml/vrmlspec.html (1995)

[Bara93] D. Baraff: Non-penetrating Rigid Body Simulation. State of the Art Reports of EUROGRAPHICS'93, Eurographics Technical Report Series (1993)

[Barz88] R. Barzel, A.H. Barr: A modelling System Based On Dynamic Constraints. Proceedings of SIGGRAPH, Vol. 22, No. 4, pp. 179-188 (1988)

[Blan90] C. Blanchard, S. Burgess, Y. Harvill, J. Lanier, A. Lasko, M. Oberman, M. Teitel: Reality Built for Two: A Virtual Reality Tool. SIGGRAPH Symposium on 3D Interactive Graphics, pp. 35-38 (1990)

[Blau92] Blau B., Hughes C. E., Moshell J. M., Lisle C.: Networked virtual environments. SIGGRAPH Symposium on Interactive 3D Graphics, pp. 157-160 (1992)

[Bric94] W. Bricken, G. Coco: The VEOS Project. Presence, Vol. 3, No. 2, pp. 111-129 (1994)

[Calv93] J. Calvin, A. Dicken, B. Gaines, P. Metzger, D. Miller, D. Owen: The SIMNET virtual world architecture. Proc. VRAIS, pp. 450-455 (1993)

[Carl93] C. Carlsson, O. Hagsand: DIVE - A platform for multi-user virtual environments. Computers and Graphics, Vol. 17, No. 6, pp. 663-669 (1993)

[Frie92] M. Friedmann, T. Starner, A. Pentland: Device Synchronization Using an Optimal Linear Filter, Siggraph Symposium on 3D Interactive Graphics, pp. 57-62, 1992

[Funk93] T. A. Funkhouser, C. H. Sequin: Adaptive Display Algorithm for Interactive Frame Rates During Visualisation of Complex Virtual Environments. Proceedings of SIGGRAPH'93, pp. 247-254 (1993)

[Gerv93] M. Gervautz, R. Devide: VAST - An integrated Animation System Based on an Actor - Controller Structure, EUROGRAPHICS'93 Animation and Simulation Workshop, Barcelona, (Sept. 1993)

[Gerv94] M. Gervautz, D. Schmalstieg: Integrating a Scripting Language into an Interactive Animation System, Proc. Computer Animation '94, IEEE-Computer Society Press, pp. 156-166, (May 1994)

[Ghee94] S. Ghee, J. Naughton-Green: Programming Virtual Worlds. SIGGRAPH Course, No. 17 (1994)

[Glas95] A.S. Glassner: Principles of Digital Image Synthesis. Morgan Kaufmann Publishers, San Francisco, California (1995)

[Gobe93] E. Gobetti, J. Balaguer, D. Thalmann: VB2 - An Architecture For Interaction In Synthetic Worlds. Proceedings UIST, pp. 167-178 (1993)

[Heck94] P. Heckbert, M. Garland: Multiresolution M odelling for Fast Rendering. Proceedings of Graphics Interface '94, pp. 43-50 (1994)

[Helm93] J. L. Helman: Designing VR Systems to Meet Psysio- and Psychological Requirements, ACM - Siggraph'93 Course 23 - Applied Virtual Reality., 1993

[Hubb93] R. Hubbold, A. Murta, A. West, T. Howard: Design issues for virtual reality systems. 1st Eurographics Workshop on Virtual Environments (1993)

[Lian93] J. Liang, Ch. Shaw, M. Green: On Temporal-Spatial Realism in the Virtual Reality Environment, Virtual Reality for Visualization; IEEE Visualization, pp. 190-196, 1993

[Lin92] M. Lin, J. Canny: Efficient Collision Detection for Animation. Proc. of the Third Eurographics Workshop on Animation and Simulation, Eurographics Technical Report Series (1992)

[Mace94] M. R. Macedonia, M. J. Zyda, D. R. Pratt, P. T. Barham, S. Zeswitz: NPSNET: A Network Software Architecture for Large-Scale Virtual Environment. Presence, Vol. 3, No. 4, pp. 265-287 (1994)

[Mazu95] T. Mazuryk, M. Gervautz: Two-Step Prediction and Image Deflection for Exact Head Tracking in Virtual Environments, Proc. Eurographics'95.

[Gren93] N. Greene, M. Kass, G. Miller: Hierarchical Z-Buffer Visibility. Computer Graphics Proceedings, Annual Conference Series, pp. 231-237 (1993)

[Paus93] R. Pausch, M. Conway, R. DeLine, R. Gossweiler, S. Miale: Alice and DIVER: A software architecture for building virtual environments. Proceedings of INTERCHI, pp. 13-14 (1993)

[Pesc95] M. Pesce, P. Kennard, A. Parisi: Cyberspace. Technical report, http://vrml.wired.com/concepts/pesce-www.html (1995)

[Ragg95] D. Raggett: Extending WWW to support Platform Independent Virtual Reality. Technical Report, http://vrml.wired.com/concepts/ragett.html (1995)

[Shaw92] C. Shaw, J. Liang, M. Green, Y. Sun: The Decoupled Simulation Model for VR Systems. Proceedings of SIGCHI, pp. 321-328 (1992)

[Sing94] G. Singh, L. Serra, W. Png, Hern Ng: BrickNet: A Software Toolkit for Network-Based Virtual Worlds. Presence, Vol. 3, No. 1, pp. 19-34 (1994)

[Snow94] D. Snowdon, A. West: AVIARY: Design Issues for Future Large-Scale Virtual Environments. Presence, Vol. 3, No. 4, pp.288-308 (1994)

[Tell91] S.J. Teller, C.H.Séquin: Visibility Preprocessing For Interactive Walktroughs. Proceedings of SIGGRAPH'91, Vol. 25, No. 4, pp. 61-69 (1991)

[Watt92] A. Watt, M. Watt: Advanced Animation and Rendering Techniques - Theory and Practice. Addison-Wesley, New York (1992)

[Zelt89] D. Zeltzer, S. Pieper, D. J. Sturman: An Integrated Graphical Simulation Plattform. Proc. Graphics Interface '89, pp. 266-274 (1989)