EG UK Theory and Practice of Computer Graphics (2005)
L. Lever, M. McDerby (Editors)

VRECKO: Virtual Reality Framework

Jan Flasar, Luděk Pokluda, Radek Ošlejšek, Pavel Kolčárek and Jiří Sochor

Human-Computer Interaction Laboratory
Faculty of Informatics, Masaryk University, Brno, Czech Republic

{flasar, xpokluda, oslejsek, xkolcar, sochor}@fi.muni.cz

Abstract

In this paper, we present a framework for experimenting with virtual environments. The architecture of the system VRECKO is designed for the rapid prototyping of techniques for human-computer interaction. The architecture is flexible, but simple. Virtual environment entities and other components can be configured at run time. We demonstrate the flexibility of the approach on several examples of experiments and tools which were realized in VRECKO.

Keywords: Virtual reality, virtual environment, interaction techniques,

Categories and Subject Descriptors (according to ACM CCS): D.2.11 [Software Architectures]: Domain Specific Architectures; I.3.7 [Three-Dimensional Graphics and Realism]: Virtual reality

1. Introduction

We are designing a system suitable for creating VR applications, allowing us to define the behaviour of components and their interactions. The most important objective is the possibility to change the configuration and component features dynamicaly.

We have developed an architecture for the rapid prototyping of techniques for human-computer interaction, called VRECKO. The main goals of the related research program are:

1) identifying and supporting appropriate functions for rapid development of testing applications,
2) creating design tools that support these activities, and
3) solving fundamental technical problems to support the creation of these tools.

We are interested in various approaches to human-computer interaction based on different metaphors. Flexibility and simplicity stand in the first place as the system is used by inexperienced students as well as inexperienced teachers. The long term goal is to implement a suite of techniques and tools evaluated in many experiments. The framework also provides the means for "screenstorming" new ideas for VR applications.

2. Related Work

In [OCS03] the authors present a novel design approach called JADE (Java Adaptive Dynamic Environment). They discuss a challenging problem with regard to the development of VE applications. Besides other problems, they describe non-extensibility, non-interoperability and non/evolution of previous VE solutions. JADE is based on a component design methodology with a layered component framework. Inter-module communication is based both on the direct accesibility and on event triggering. The event distribution scheme uses EventDispatcher with a dynamic registration of subscribers. The JADE kernel and the resulting system can be configured at startup via a command line or a file containing for example, an XML description.

Similar problems were recognized in [KMC02] and led to the development of a unified component framework implemented in Java™. The framework allows to reconfigure dynamically, add, remove, and upgrade components at run time.

EVI3d [TBBB02] is a distributed architecture allowing interactions within virtual environments. This framework manages many multi-sensorial devices including haptic devices. The structure of this architecture allows a complete dispatching of device services and their clients on as many machines as required.

In [TJ01], authors describe a design model for developing virtual reality interface called VRID. They focus on the methodology for designing interface components of a VR system. They propose flexible multi-component object architecture with composite behaviour.

Collision detection (CD) techniques for large scenes with many objects are often based on bounding volumes hierarchies (BVH). The most popular BVHs are constructed with AABB's [vdB99], OBB's [GLM96], sphere trees ([PG95], [Hub96]), and k-dops [HKM96], [KHM*98]. These techniques often use space sorting structures prepared in the preprocessing phase. The research of CD techniques faces many problems related not only to a scene's flythroughs, but especially to CD in dynamically changing scenes.

3. VRECKO

The primary purpose of VRECKO is to advance the research in human computer interaction techniques in 3D virtual space by enabling researchers to work directly with VR tools. Our research currently focuses on supporting rapid prototyping of various experiments. For this reason, VRECKO is built as an open programming environment. Our long-term goal is to enable researchers to rapidly develop and test their VR experiences.

A guiding principle of our work is the intention that VRECKO build on existing design practices and tools. We are initially adapting existing interaction paradigms and solutions. As VRECKO platform supports various devices and techniques, we can easily test numerous combinations of different manipulation techniques in VR space to evaluate their usability.

3.1. Hardware

The Human Computer Interaction Laboratory is equiped with numerous VR devices . The working space is situated in front of a large stereo screen with polarized back-projection. The user's position is tracked by the Nest of Birds with 4 6DOF trackers. Other data inputs are provided by 16-DataGloves, PinchGloves, 3D mouse, tablet and a PHANToM^{TM}6DOF tracker. Force feedback devices include 2 PHANToMs 1.0 and a Reachin Display (PHANToM + mirrored stereo-display). A user may experience VR space either using polarized glasses or i-Glasses with a head tracker. With data gloves on the tracked hands, one can touch and manipulate virtual objects "in front" of the large stereowall (Figure 1).

3.2. The VRECKO Architecture

A cornerstone building block of VRECKO is an entity which can possess a number of abilities. The core entities include *Scene*, *SceneObject*, *EnvironmentObject*, *LogicalDevice*, *Scheduler*. *Scene* is a container of *EnvironmentObject*s,

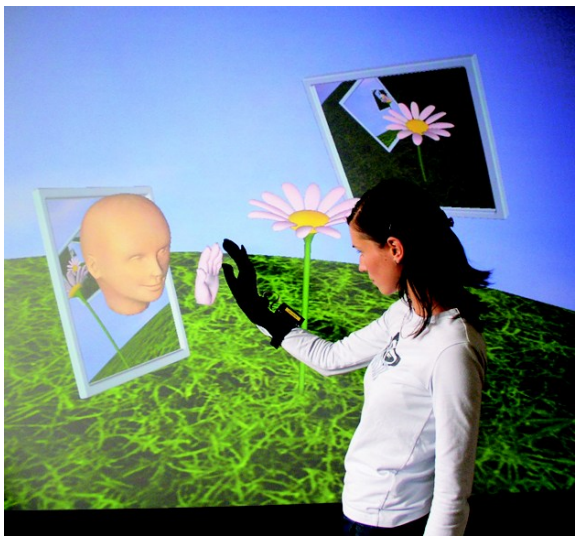

Figure 1: *Interaction with VE (Stereo Off)*

which are organized in a *SceneGraph*. Entities can accept and change the abilities and responsibilities in runtime. This is accomplished using the technology *Objects with changeable roles* [MS01, MS02], which allows controlled supplementing, changing and taking away of object roles. Accepting a role, the entity (object in a scene, tool, logical device etc.) gains the ability to communicate and fulfil new tasks.

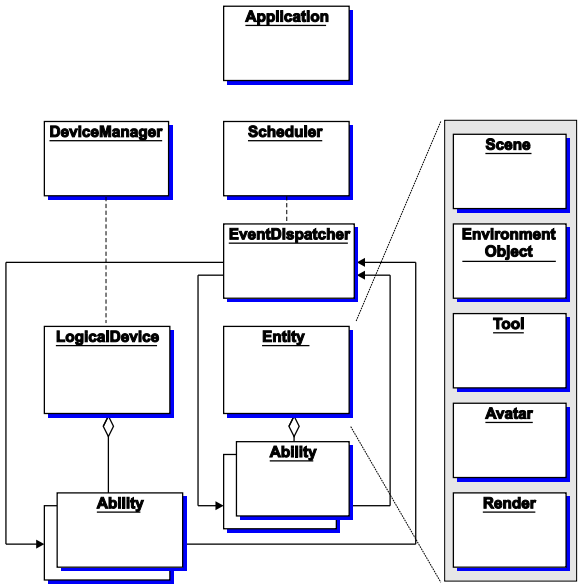

Figure 2: *Simplified system architecture*

The communication scheme (Figure 3) uses a similar approach as in [OCS03]. Entities can communicate directly by

messages, or generate events for known receiving entities. The most frequently used mechanism for communication is maintained by the *EventDispatcher* which queues input events and messages and distributes them to relevant receiving entities. The system runs in discrete time mode. Timing frequences and respective timing slots are controlled by the *Scheduler* and may differ for individual entities.

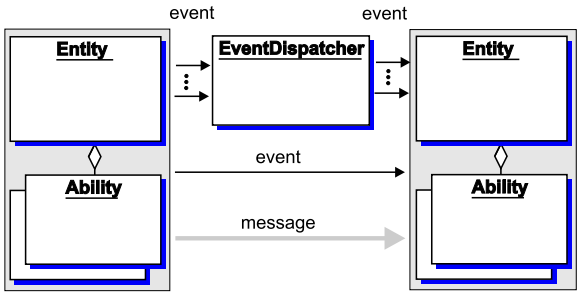

Figure 3: *3 ways of communication*

Global rendering architecture is based on graphics patterns which combine local and global illumination models [OS03]. For the rapid rendering of a scene, VRECKO uses OpenSceneGraph [OSG05], an open source 3D graphics toolkit. Written entirely in Standard C++ and OpenGL it runs on many platforms, including MS Windows and GNU/Linux. It supports several culling techniques, LOD rendering and display lists as part of the core scene graph. OpenSceneGraph also supports easy customization of the drawing process which is suitable for prototyping of new approaches.

3.3. Collision detection

For collision detection, the VRECKO system uses two external libraries. Of these, the ESG (Extensible Scene Graph) library was developed in our laboratory. It serves for obtaining spacial information about objects in a scene. Every object is accesed via adapter which encapsulates space and collision functions of the ESG library. As the objects may consist of many primitives, they are sorted in a spatial data structure. Current implementation includes bounding volume hierarchies (BVH) with spheres [Hub96], AABBs – axis aligned bounding boxes [vdB99] and 14-DOPs [KHM*98].

Via an adapter, VRECKO can invoke the CD method for a pair of objects, or to obtain their actual spatial data and relations. The following methods were implemented:

Collision detection Rapid detection based on BVH with 14-DOPs.

Distance Calculation A negative number indicates a penetration. The method can also output a probable direction of the smallest gap between objects. This direction then suggests the most probable shortest path between objects or, in the case of penetration, the shortest path of leaving penetration. The second case is useful for controlling force feedback.

Finding primitives within a limited distance Choosing objects in the neigbourhood of a given point, evaluation of force feedback.

Intersection with a ray Utilized for visibility, navigation in VE, selection of objects, etc.

The extension of an object in a given direction Useful for estimating an approximate area occupied by the object.

The ESG library does not solve dynamic collisions. In VRECKO, it is used only for calculating some interactions among objects, for example to evaluate the reflections in a virtual mirror, for grabbing and manipulating virtual tools.

For simulating rigid body dynamics, an open source, high performance ODE (Open Dynamics Engine, [ODE05]) library is used. It has integrated collision detection with friction. In VRECKO it is used for simulating a dynamic behaviour of objects in virtual reality environments. Subsequent collisions resulting from interactions are recognized and ODE library determines the new position of an object. The detected changes are transfered to a scene status.

3.4. Haptic Rendering

The current implementation contains the subsystem for smooth haptic rendering of surfaces. The force feedback device, PHANToM 1.0 (6DOF position/direction input, 3DOF force output) is controlled by multithreaded architecture [HBS99, Kab01, KS04] which enables the running of a haptics interaction loop with a high frequency – of up to 3kHz.

The haptic sensation of a surface differs from the visual one. Instead of getting all information "at once", as we do when we see, when we rely on touch we have to explore the surface through the movement of fingers. Without supplementing visual information, the haptic recognition of surface characteristics is very difficult. Therefore we are developing a technique allowing fast haptic exploration based on the suppression of details with respect to the speed of finger movement. It is a direct analogy to LOD techniques used in visual rendering but with different criteria.

3.5. Haptic tools

An important tool implemented in the haptic part, is *Virtual Fixture*. The original concept was published by Rosenberg [Ros93] and used by [KPZ*04, PS02]. Virtual fixture is a computer-generated virtual tunnel with haptics and/or graphics features. It is designed to provide guidance along a given path and to facilitate movements with higher precision.

We use the concept of virtual fixtures to provide aid for visually impaired people for orientation in a model of a building. The model of a building is rendered via a haptic device

using a polygonal mesh rendering algorithm. The computed forces resulting from the haptic rendering are added to forces generated from virtual fixtures. The list of virtual fixtures can create a path for a virtual walkthrough of a building. The user is able to turn on/off fixtures whenever he/she feels lost in a virtual space. A set of virtual fixtures is programmed as abilities which can be added to an environment object. Virtual Fixtures are also visualised when needed.

Another tool is the *virtual sensor* with different shapes which emits a signal when entered. Virtual sensors are used when building a dynamic path. The virtual sensor is placed at the end of a virtual fixture and when reached, a signal is emited. This signal activates the next virtual fixture and disables the previous virtual fixture. Finally, some auxiliary abilities can be used to support haptic exploration of a scene. The *axis lock* creates virtual walls along the selected axis; the *grid* is rendered via thin viscose walls; the *damper* simulates a viscose environment and can be used to reduce unwanted oscilations of the device.

3.6. Audio

For the sakecompleteness, the VRECKO system also includes sound abilities. The rendering of 3D sound in the VR workspace is accomplished using OpenAL library, with an output to five loudspeakers.

4. Experiments with interaction tools

Research in the laboratory is focused on HCI techniques employing various tools and their combinations. Using VRECKO as the platform for prototyping, we implemented a set of tools called World In Miniature [PBBW95, FKMK98, VC01], Magic Mirror [GC99], virtual hand for remote manipulation using go-go techniques [BH97], 3D menus etc . The system's flexibility and its simple, but powerful architecture allows us to prototype combinations of different techniques and evaluate their usability when solving a set of testing tasks. Taking into account more then 30 basic techniques for a movement, selection and manipulation of virtual objects in a scene, we obtain a large number of combinations to be tested and assessed. In Figure 4, we show a snapshot from experiment testing WIM with Magic Mirror and dynamics using ODE library.

Figure 5 shows a screenshot of testing a combination of 2-hand manipulation using Magic Mirror.

An experienced user is able to prepare a new combination of tools including new abilities in a very short time. Figure 6 depicts a screenshot from complex experiment named "Behind obstacle" where a user has to place a set of objects to a given locations behind a wall.

Current experiments concentrate on controlling a VR scene without menu of any form, for example by two-handed manipulations [LL00]. Three sensors are used to track the head and the two hands of the navigator. A user is saved from choosing the techiques and their parameters. In the course of the experiment, he/she is informed how to fulfill the given task using the activated devices and commands only. The user does not use a menu. A selection of, and manipulation with objects is solved by tracking hands and head movements, with simple gestures entered by the data gloves or contact gloves.

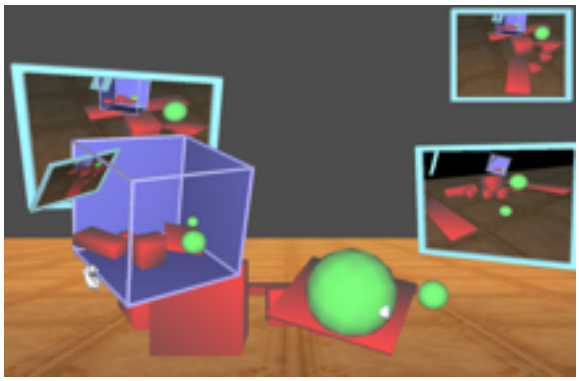

Figure 4: *Testing WIM and Magic Mirror*

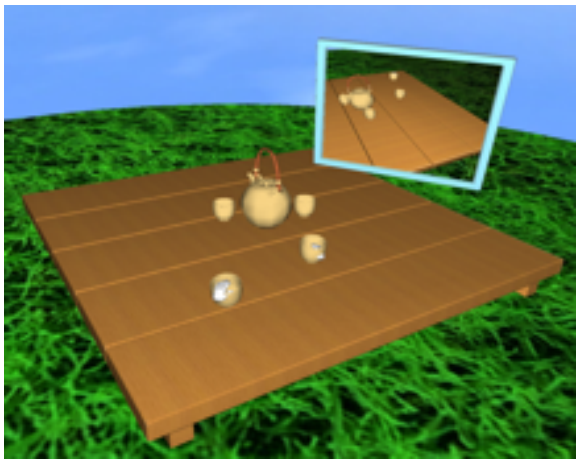

Figure 5: *2-handed manipulation using mirror*

Experiments with haptics explore new means of communication for totally blind people. An example of it is an application, which uses virtual fixtures to build a Gesture Map. The technique utilizes a symbolic map of a building presented as the graph of gestures. A gesture is performed by PHANToM's arm driven by a computer by moving along a predefined path. The user is holding device end-effector watching its movements. Gestures may express either a topology or geometry or they provide a user with some hints about available steps and processing. Figure 7 depicts the map used in our experiment.

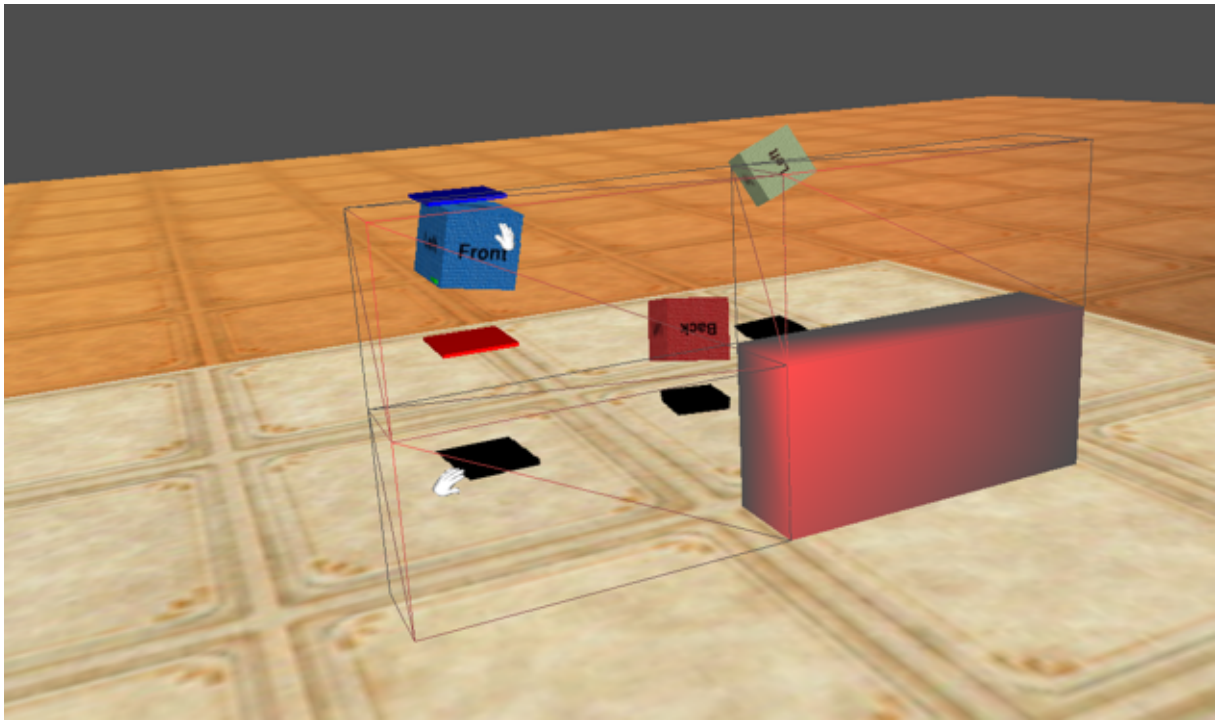

Figure 6: *Experiment "Behind obstacle". A user can change the visibility of a part of the wall and use manipulator to change his location in the scene.*

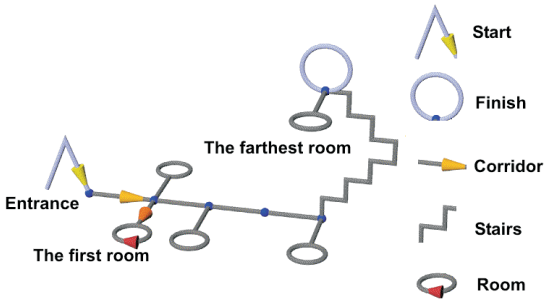

Figure 7: *Symbolic map of building using gestures*

4.1. Conclusion

We introduced the VRECKO system which we use as the platform for experimenting with egocentric and exocentric metaphors in VR. The system is under development and it is not for building large applications. We plan to stabilize the system's architecture, i.e. to (re)implement some parts in order to obtain a clear, modular solution. Among other extensions, we plan to include multiple force-feedback abilities. Currently, the system is being adapted for collaboration in a VR world shared from distant locations using a high speed network. This work is a part of a related project "Collaborative Virtual Environments" in cooperation with West Bohemia University, Pilsen, Czech Republic.

By making VRECKO accessible to many students and researchers, we can achieve its potential to create various experiments in VR scene exploration, as well as educational experiences.

4.2. Acknowledgements

This work was supported by the Ministry of Education, Czech Rep., grant no. MSM143300003 and by Faculty of Informatics Masaryk University, Brno, Czech Republic.

References

[BH97] BOWMAN D. A., HODGES L. F.: An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics* (1997), ACM Press, pp. 35–ff.

[FKMK98] FUKATSU S., KITAMURA Y., MASAKI T., KISHINO F.: Intuitive control of "bird's eye" overview images for navigation in an enormous virtual environment. In *VRST '98: Proceedings of the ACM symposium*

on Virtual reality software and technology (1998), ACM Press, pp. 67–76.

[GC99] GROSJEAN J., COQUILLART S.: The magic mirror: A metaphor for assisting the exploration of virtual worlds. In *SCCG'1999* (04 1999), UK Bratislava.

[GLM96] GOTTSCHALK S., LIN M. C., MANOCHA D.: Obbtree: a hierarchical structure for rapid interference detection. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), ACM Press, pp. 171–180.

[HBS99] HO C.-H., BASDOGAN C., SRINIVASAN M. A.: Efficient point-based rendering techniques for haptic display of virtual objects. *Presence 8*, 5 (1999), 477–491.

[HKM96] HELD M., KLOSOWSKI J. T., MITCHELL J. S. B.: Collision detection for fly-throughs in virtual environments. In *SCG '96: Proceedings of the twelfth annual symposium on Computational geometry* (1996), ACM Press, pp. 513–514.

[Hub96] HUBBARD P. M.: Approximating polyhedra with spheres for time-critical collision detection. *ACM Trans. Graph. 15*, 3 (1996), 179–210.

[Kab01] KABELÁČ Z.: Haptical rendering of triangle meshes with fixed direction hulls. In *Eurohaptics 2001* (2001), pp. 138–141.

[KHM*98] KLOSOWSKI J., HELD M., MITCHELL J., SOWIZRAL H., ZIKAN. K.: Efficient collision detection using bounding volume hierarchies of kdops. *IEEE Trans. on Visualization and Computer Graphics 4*, 1 (1998), 21–37.

[KMC02] KAPOLKA A., MCGREGOR D., CAPPS M.: A unified component framework for dynamically extensible virtual environments. In *CVE '02: Proceedings of the 4th international conference on Collaborative virtual environments* (2002), ACM Press, pp. 64–71.

[KPZ*04] KUANG A. B., PAYANDEH S., ZHENG B., HENIGMAN F., MACKENZIE C. L.: Assembling virtual fixtures for guidance in training environments. In *12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS 2004), 27-28 March 2004, Chicago, IL, USA, Proceedings* (2004), IEEE Computer Society, pp. 367–374.

[KS04] KOLČÁREK P., SOCHOR J.: Haptic rendering using velocity driven level of detail. In *Workshop in Virtual Reality Interactions and Physical Simulations* (09 2004), pp. 149–158.

[LL00] LIN C.-R., LOFTIN R. B.: Vr user interface: closed world interaction. In *VRST '00: Proceedings of the ACM symposium on Virtual reality software and technology* (2000), ACM Press, pp. 153–159.

[MS01] MARKOVIČ L., SOCHOR J.: Objects with changeable roles. In *DOA'01* (2001).

[MS02] MARKOVIČ L., SOCHOR J.: Object model unifying wrapping, replacement and roled-objects techniques. In *Workshop #09 - Unanticipated Software Evolution, 16th European Conference on Object-Oriented Programming* (June 2002).

[OCS03] OLIVEIRA M., CROWCROFT J., SLATER M.: An innovative design approach to build virtual environment systems. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003* (2003), ACM Press, pp. 143–151.

[ODE05] Open dynamics engine. `http://www.ode.org/`, 2005.

[OS03] OŠLEJŠEK R., SOCHOR J.: Generic rendering architecture. In *Conference Proceedings 2003 Theory and Practice of Computer Graphics* (June 2003), IEEE Computer Society.

[OSG05] Openscenegraph. `www.openscenegraph.org`, 2005.

[PBBW95] PAUSCH R., BURNETTE T., BROCKWAY D., WEIBLEN M. E.: Navigation and locomotion in virtual worlds via flight into hand-held miniatures. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), ACM Press, pp. 399–400.

[PG95] PALMER J., GRIMSDALE R.: Collision detection for animation using sphere-trees. *Computer Graphics Forum 14*, 2 (1995), 105–116.

[PS02] PAYANDEH S., STANISIC Z.: On application of virtual fixtures as an aid for telemanipulation and training. In *Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (2002), pp. 18–23.

[Ros93] ROSENBERG L. B.: Virtual fixtures: Perceptual tools for telerobotic manipulation. In *Proceedings of IEEE Virtual Reality Int'l. Symposium* (1993), pp. 76–82.

[TBBB02] TOURAINE D., BOURDOT P., BELLIK Y., BOLOT L.: A framework to manage multimodal fusion of events for advanced interactions within virtual environments. In *EGVE '02: Proceedings of the workshop on Virtual environments 2002* (2002), Eurographics Association, pp. 159–168.

[TJ01] TANRIVERDI V., JACOB R. J.: Vrid: a design model and methodology for developing virtual reality interfaces. In *VRST '01: Proceedings of the ACM symposium on Virtual reality software and technology* (2001), ACM Press, pp. 175–182.

[VC01] VALLANCE S., CALDER P.: Context in 3d planar navigation. In *AUIC '01: Proceedings of the 2nd Australasian conference on User interface* (2001), IEEE Computer Society, pp. 93–99.

[vdB99] VAN DEN BERGEN G. J. A.: *Collision detection in interactive 3D computer animation*. Eindhoven : University Press Facilities, 1999.