

VR Deployment: Building Interactive 3D Applications Using Image Projection and PC Clusters

By David Nahon¹
VR and Simulation Solutions Manager at [Virtools](http://www.virtools.com)

Abstract

Developers face many technical questions when building a Virtual Reality (VR) solution with large screens and/or inter-networked PCs, or "clusters." The goal of this paper is to facilitate the decision-making process to help people make the most optimal hardware choices. One key issue addresses ways of targeting the right display for the right audience (number of screens, screen shape, projection and interaction devices, type of stereoscopy). The paper also offers insight for those working with clusters, to help design the appropriate system (number of nodes, off-the-shelf or homemade, scalable or traditional).

In addition to our guide for choosing hardware systems, we conclude this paper by covering essential features of effective VR cluster-based software and discussing reasons why the Virtools Software Suite is an ideal authoring platform for VR purposes.

¹ David Nahon directed R&D at Z-A Production (<http://www.z-a.net>) for 8 years. He supervised development of networked, interactive VR applications and exhibits (see <http://david.cyber-nahon.net>) and designed the software architecture of the SAS Cube, the first PC-based, CAVE-like display which pioneered the development of the Virtools VR Pack.

Special thanks to Hillary Goidell for her invaluable English rewriting.

Going VR?

Historical Context

Virtual Reality (VR) enables the creation of virtual stimuli that mimic the natural world by creating the illusion of "real" environments. Such sensory data often explores only the visual realm, but acoustics and haptics are becoming vital in the field of VR. VR applications typically involve three essential components: real-time processing, immersion and interaction. It is however very common to speak of VR whether or not an environment is immersive, and even when only visual clues are used.

In the past, Visual Immersion has relied on headsets with rotation sensors that update visual frustum values based on head rotation. Unfortunately, good VR headsets are still relatively limited as far as resolution and field of view are concerned. High-end headsets can be very expensive and are usually extremely heavy. At the same time, however, the cost of video projectors has dropped considerably and projectors can now be found in both office and home. As a result, the market has seen a substantial proliferation of small dual projector passive stereoscopy displays, which offer many benefits – without the disadvantages of VR headsets.

Similarly, the video games market has grown exponentially, resulting in the emergence of very powerful yet inexpensive graphics cards (and PCs in general). The upshot for the VR field is that high-end visual supercomputers are no longer needed to power VR applications (see "[Why use a cluster instead of one multi-pipe, multi-processor supercomputer?](#)") – even large-scale VR displays that run multiple projectors simultaneously, can now be powered by PC clusters. Today, PCs can be assembled in cheap clusters to replace bulky systems. The advantages are clear: clusters can power an almost unlimited number of screens, and can naturally ensure both scalability and low-cost hardware upgrades.

As for software, game engines and development platforms now offer robust yet versatile options that are increasingly being used to make VR applications. The time has now come to build innovative VR solutions that merge gaming, video projection and cluster technologies.

The Right Display for You

A Complex Equation

Designing an immersive experience means striking the right balance amongst various input and output parameters.

Input:

- Number of simultaneous, active users
- Number of passive users
- Degree of interactivity (navigation / manipulation)
- Scenography: show space and control room, sound requirements and constraints

Output:

- Number of screens and screen shape (curved/flat, cubic, cylindrical, conic, spherical)
- Interaction devices
- Front or rear projection
- Stereo mode (if necessary)

Choosing the right display is a complex task that calls for a certain level of expertise. While all members of a production team need not be developers, VR content design does require in-depth understanding of hardware constraints.

This chapter provides a few tips and tricks to keep in mind: the rules of thumb to help you solve the equation described above.

Display Taxonomy

Before elaborating on VR systems on the whole, it's important to identify various categories and types of projection-based displays:



Figure 1: Visbox®

- **Simple Stereo Walls:** Barco Gemini®, VisBox®, Geowalls, etc. These are entry displays, usually based on pair of consumer or cheap LCD projectors using passive stereo. They can be bought of the shelf from manufacturers or integrators, or built following the experience of former (see [1])



Figure 2: Barco Baron®

- **Tables:** Immersadesk®, Baron®, Responsive Workbench®, Consul®, etc.
- **Large Flat Multi-Projector Displays:** CAD Walls®, Power walls®, etc.
- **Multiscreen Right-Angle Displays:** CAVE®, Workbench®, Hollobench®, SAS^{cube}®, I-Space®, VRCube®, etc.
- **Large Curved Multi-Projector Displays:** Reality Center®, Panorama®, I-Domes®, I-cone®, etc.

16 Rules to be on the Safe Side

Rule 1: Angular displays such as CAVE® environments require stereoscopy. This is the only way of eliminating the angle from viewer perception. Without it, the result will resemble a series of painted walls.

Rule 2: Even if there is a relatively large space between two screens (up to approximately 2% of the overall screen width) the stereo effect across the 2 screens can still function correctly!

Rule 3: Accurate stereo depends on head orientation. So either use head tracking or consider that viewer(s) will always be looking straight ahead. Note that recent publications discuss omnistereo [1] techniques for solving such issues.

Rule 4: When multiple users are in a CAVE®, they should all try and get as close as possible to the position and orientation of the person being tracked.

Rule 5: In a CAVE®, correctly displayed objects will appear to be "in the hands" of the tracked person, and in his hands only, at any given moment.

A straightforward solution to this inherent limitation is to switch perspective in response to requests coming from

various viewers. For example, in order to gain control of perspective, a viewer can click a device button. The limitation can thus be managed by the application or with the device driver itself (such as AR-Tracking's multiple-flystick option [6]). However this solution does not simultaneously provide correct perspective for all viewers. To remedy this, much research is being done on topics such as frame sequential, ultra-high video rates [3], which may potentially hit the market in the near future.

Rule 6: Hardware image warping is only accurate from a single point of view. This means it is incompatible with head tracking, though hand tracking will work.

For view-dependent applications, only software warping will work. Software warping makes use of the very same graphics card as the one generating the 3D imagery.

Rule 7: When you want many people in one immersive room to see your application simultaneously, use a display that is as big and as flat as possible – and never angular.

That way you'll be sure to offer correct perspective from the largest possible viewing area. Keep in mind however that the flatter the display, the less immersive it is. In trying to strike the right balance between accurate viewpoint and sense of immersion, large cones, cylinders or spheres offer a good solution (e.g. i-Cone®, Reality Center®, i-Dome®).

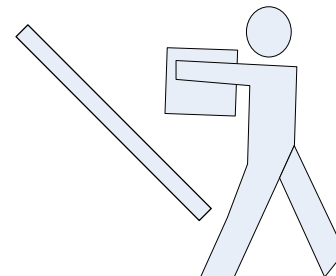


Figure 3: Tilted screen for hand centered interaction

Rule 8: Hand-centered manipulation (user has an "object in hand") works better with a tilted screen (Baron®, Idesk®) than on a vertical wall. This is partly because the latter offers more natural positioning between head, hands and screen.

Another solution that affords greater hand space is an L-shaped dual screen "table" (Workbench®, Consul®, etc.).

Rule 9: A closed (cubic) CAVE-like display with no floor is worthless for hand-centered interaction.

The user will eventually look at his hand, and in the process his gaze will be focused on the floor. This display is only valid for simulations in which the user is seated (i.e. driving or flight simulators).

Rule 10: The fifth face of a CAVE should be the top screen, not the rear. Rear doors are usually left open to avoid a claustrophobic feeling.

Rule 11: In a CAVE, if both a floor and top screen are needed, the floor must be made of glass approximately 3m x 3m, and 10 cm thick. This would potentially weigh about 3 tons: it is clearly not a solution for a mobile CAVE and may even require modifying the architecture of the building.

Rule 12: Front projections using passive, polarization-based stereo require special "silver" screens. Stereoscopy will be lost on regular screens.

Caparol Capadecor Alucryl is a paint coating that has proven to maintain polarization.

There are however passive stereo solutions which do not require special screens. These technologies (such as Infitec [5]) are embedded in the projector and use passive eye separation techniques other than polarization.

Rule 13: Avoid mixing front and rear projection using passive stereo: you'll have a hard time matching color and brightness. It's precisely for this reason that you'll rarely see 4-face CAVEs with passive stereo. Barco's PASCAD® proprietary screen [5] does provide an alternative, as does the Infitec®.

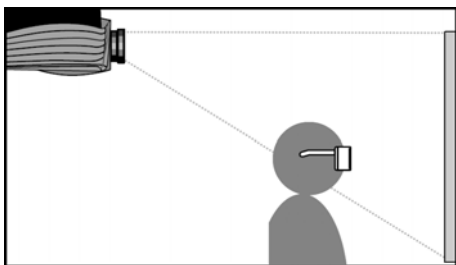


Figure 4: Front projection with shadow issue

Rule 14: If using front projection, head tracking is likely to be ineffective – the viewer will very likely cast a shadow that will spoil the projection!

Rule 15: If using head tracking with passive stereo, use circular (not rectilinear) polarizing filters to maintain polarization when the viewer tilts his head.

Rule 16: Hire professionals to assemble complex displays unless you're ready to spend lots of time (and probably money) to get things up and running.

Do I need a visualization cluster? If so, which one?

Choosing the right computer platform for visualization means:

- Deciding if a cluster is needed.
- If it is, deciding on the number of nodes to be used.
- Deciding whether or not specific hardware features are needed (genlock, framelock, rate lock).

When should I use a PC Cluster? Why not use a single PC?

The most obvious reason to use a cluster is because you don't have enough video outputs (current 3D cards are limited to 2 or at most 3).

Electronic devices called image splitters can clearly overcome this limitation by dividing a large frame buffer into smaller ones, each of which will be displayed by a different projector. Image splitters are even bundled with some projection systems.

Having a single card sequentially compute large numbers of images unfortunately comes with some obvious performance drawbacks, not to mention that a huge quantity of memory must be allocated to the frame buffer.

A simple way to parallelize rendering entails using more than one graphics card in the same PC. This will indeed provide better pixel and geometry performance, but won't scale very well in terms of CPU since processors are usually limited to 2 units per PC. For larger scenes, CPU plays an important role in the culling phase and is heavily solicited for physics or kinematics loops.

The very nature of bus architecture is another reason not to overload a single PC with too many cards. PCI-Express is beginning to replace existing AGP/PCI architecture, which could potentially increase bandwidth speed by twofold. However the bus is still likely to saturate, especially if the scene being displayed is very dynamic (with particles or mesh deformations, for example) or if all the textures and models cannot fit in the graphics card texture.

A more general argument for using clusters is that a set of clustered PCs will always be faster than a single unit of the same type PC!

Why use a cluster instead of one multi-processor, multi-processor supercomputer?

There are very good reasons to use SGI Onyx-like hardware or the Linux-based equivalent [7]. Unified memory and the unique NUMAflex™ bus architecture make it possible to visualize and manipulate huge databases with performance levels that visualization clusters just can't rival.

Yet in many circumstances, such powerful solutions are simply not necessary. Their high price tag remains a barrier for many users, even if the components themselves have benefited from mass market expansion (Intel CPUs, ATI Graphics chips, etc.) and are now more affordable. A more inexpensive solution for an initial investment and subsequent maintenance is to buy a few gaming PCs. By replacing the graphics cards and/or occasionally upgrading them, you'll have acquired a more economical solution that will also cost less to maintain over time.

On the other hand, supercomputers can display data without needing to re-export to other formats to run viewers on PC clusters. Similarly, if using a PC cluster makes the VR authoring process way too complicated, a cluster solution may wind up being cost prohibitive in the end.

For that reason, good cluster-ready authoring software must be able to absorb much of the intricacy of cluster production.

What hardware features should you look for in a cluster?

Key hardware features to look for in a graphics cluster include:

- Vertical Sync
- Genlock / (Framelock as NVIDIA calls it)
- Swaplock (Framelock as Wildcat calls it)
- Rate Lock

To better understand these concepts, we need to review a few graphics technology basics.

Raster Scan Technology

Raster scanning is used by CRT monitors, though the process is similar for LCD screens as far as graphics card behavior is concerned. An image is drawn by an electron beam that scans the screen from top to bottom and left to right, illuminating pixels along the way:

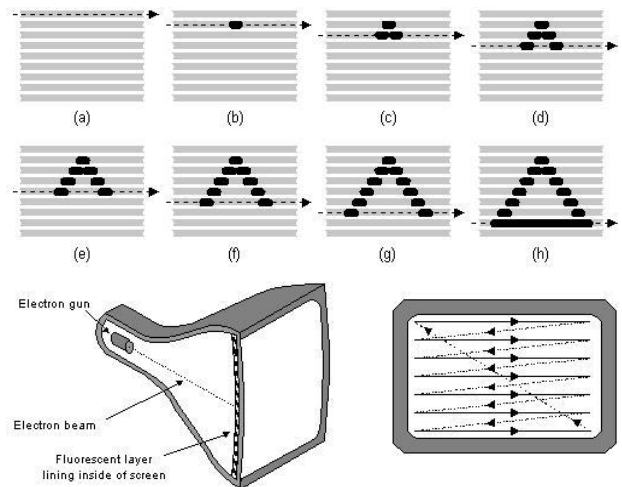


Figure 5: Raster scan technology
(Image courtesy of Wildcat)

When the beam reaches the right edge, pixel illumination goes black and the beam instantly goes back to the other edge to start the process again. Similarly, once the beam reaches the bottom of the screen, it returns to the top.

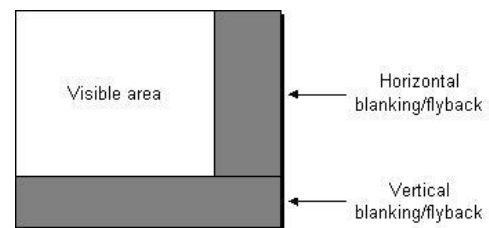


Figure 6: Monoscopic blanking

When in frame sequential stereo mode, this happens twice per frame.

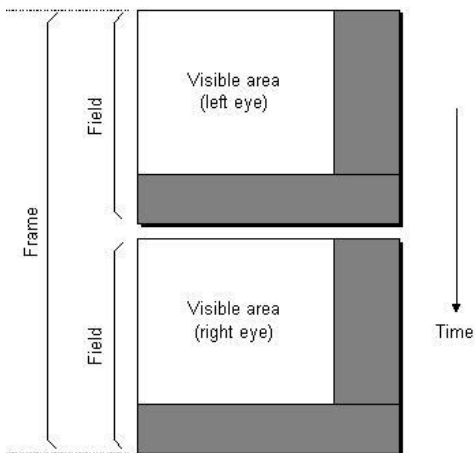


Figure 7: Stereoscopic blanking

Vertical Sync

When using double-buffering mode, images are generated in a hidden back buffer and revealed only when ready for display. Any graphics card can be instructed to not swap to the next image until the picture signal is "blanked," or invisible. This mechanism, called "vertical sync" or "vsync," prevents "horizontal image tearing" with rapidly changing images. Vsync can be activated either at the driver level or by the application. Note that when vsync is on, frame rate becomes a multiple of the refresh rate.



Figure 8: Horizontal image tearing (the image is being updated at the same time it is being drawn)

Genlock

The theoretical definition of genlock is the ability to synchronize the generation of the image based on an external signal pulse ("house sync"). In relation to visualization clusters, genlock usually means synchronizing the refresh rate using one of the clustered PCs as a reference. Note that NVIDIA calls this feature "framelock," while Wildcat uses the term "framelock" when referring to

hardware swaplock (see below). To avoid confusion, we recommend avoiding the term framelock altogether and using the term genlock instead, even when no external sync signal is involved.

Genlock can be managed via hardware. Some graphics cards offer a dedicated feature to handle genlock, thanks to a daisy chain linking all the cards together.



Figure 9: Genlockable graphics card

Using overclocked Linux kernels, some software techniques [12] have been successful in attaining genlock with consumer graphics cards.

Swaplock

Back to front frame buffer swapping may need to be synchronized when using a cluster. This is called swaplock.



Figure 10: Vertical image tearing

Swaplock prevents inter-screen discontinuities (sometimes called "vertical image tearing") caused by PCs in a cluster presenting images at different moments. Note that swaplock can be managed by the graphics card hardware using a dedicated inter-PC link (usually the same as hardware

genlock). Alternatively, a software solution for swaplock involves using the network. Latency with good software swaplock typically stays under a millisecond, which is negligible compared to frame rate latency (1/60 sec = 17 msec).

Rate Lock

Rate lock is useful when you need to keep frame rate as stable as possible. In certain cases, it is better to keep a steady rate of 30 Hz rather than constantly oscillating between 120, 60, 40 and 30 Hz. Once locked at 30 Hz, this ostensibly "low" frame rate will seem normal, since there won't be a faster frame rate to compare it to.

Although rate lock is widely utilized in driving simulators, VR applications rarely make use of it, especially with head-tracking since the benefits of stability are offset by the constant lag that results.

Do I need Swaplock?

Swaplock is needed when screens are displayed adjacent to one another, for instance in CAVEs or Reality Centers. When the gap between screens is only a few centimeters, swaplock seems less necessary. Note also that the better the framerate, the less you may need swaplock.

Do I need Genlock?

Genlock is a must with displays using multi-screen active stereo. If not used, stereo will only work correctly on the screen whose graphics are generated by the shutter glasses, and gray bars will appear on the other screens (*blanking zone*, cf. the "[Raster Scan Technology](#)" chapter).

Genlock is also needed in order to activate vertical sync and for satisfactory swaplock: If graphics cards are not genlocked, each PC will wait for its own vertical signal to trigger swapping. Without genlock to ensure this image swapping happens simultaneously on all PCs, vertical image tearing will result. This tearing is still relatively limited, since the difference in swap times still remains inferior to the vertical sync period. The latter decreases as swapping frequency increases. By and large, you should not be overly concerned by this minor discrepancy should unless you are targeting very high-end displays.

Remember that when using head tracking, the image is updated every frame since the viewer's head is never perfectly motionless. On the contrary, simulation applications without tracking may be less sensitive to poor swaplock.

Commercial Hardware: Off-the-Shelf Cluster Solutions

A visualization cluster is essentially a set of inter-networked PCs, potentially equipped with genlocked/swaplocked graphics cards.

When you need genlock/swaplock, three key commercial solutions exist:

- **3DLabs Wildcat 4 (7210) or Wildcat Realizm (200 or 800)** with G board: 3DLabs pioneered hardware solutions for PC graphics board genlock/swaplock and have been experts in the field for years. They recently released the long-awaited Realizm card, which combines their genlockable technology with modern programmable architecture (including vertex and pixel shaders).
- **NVIDIA FX3000G, FX 4400G:** NVIDIA is historically the first graphics card manufacturer to offer both shaders and genlock. As far as Windows genlock issues are concerned, NVIDIA will soon be releasing the next generation of FX 4400G cards, based on new hardware technology that will solve such problems once and for all.
- **Orad DVG** offers smart, scalable architecture that lets you use more than one computer per screen, thanks to a unique pixel bus that daisy chains graphics nodes. Orad DVGs rely on PCI/PCI-Express add-on cards and standard graphics cards that have been specially tweaked. Depending on technical needs, the DVG can link different nodes in the appropriate way: eye chaining (2 nodes create active stereo), AA chaining (slightly different views are blended to increase antialiasing quality), time division, screen space division (screen is split into zones to reduce each node's pixel and geometry load), etc. See reference [8] for more details.

Choosing the appropriate VR software

The only locking mechanism not mentioned in the section "[What hardware features should you look for in a cluster?](#)" is "scene lock," or synchronization of the scene state across the cluster. As to be expected, this is not a hardware feature but a job for VR software. Questions that arise include "How can I be sure object animation are displayed correctly on all screens?" and "What about complex physics, particle effects and even scenes using random behaviors?"

"Scene Lock" / Distribution Modes

There are two basic ways of handling scene synchronization, by using either "cause" or "effect" distribution. Stated differently, you can either synchronize

user actions and timed events, or instead share the scene state resulting from such actions and events. Both modes have their advantages and drawbacks. Base your choice on the conditions and constraints of your application and technical architecture.

Consider a scene that changes significantly with each new frame due to mesh deformations or multiple animated objects, for example. In this case, scene state sharing may be next to impossible since the network bandwidth required to send all the data is likely to saturate the actual physical network.

On the other hand, by sharing events or user interactions you may save on bandwidth but end up devoting unnecessary attention to ensure adequate host synchronization, especially for clock and random synchronization. It's also much more complex to manage PC reconnection when the full scene state isn't shared across the cluster at each frame.

VR software "must haves"

We've now covered the essentials of image generation and display, now it's time to outline vital features for VR software.

For creating interactive 3D applications using image projection and PC clusters, VR software must:

- Be cluster-ready!
- Support cluster hardware features (genlock/swaplock) and able to provide software-only swaplock if necessary.
- Allow users to scale clusters (up or down), by supporting both multiple outputs per PC and multiple PCs per output.
- Ensure that authoring and deployment are as easy with a cluster as with a single PC.
- Make it possible to deploy at various sites without any authoring changes.
- Enable separate purchase of development or runtime licenses.
- Be flexible to allow cause, effect or mixed distribution.
- Be compatible with cutting-edge graphics features (pixel and vertex shaders, vertex buffer objects, PCI-Express, etc.).
- Be compatible with any display, from HMDs to CAVEs, and even online deployment when possible.
- Support hardware and software warping, as well as hardware and software edge blending.
- Support all kinds of stereo and a wide range of VR peripherals, and be able to switch from one to another without making any authoring changes to the application.
- Be able to emulate peripherals when none are present.
- Have open architecture to let you to add new devices.

- Be built on a powerful, robust platform.
- Be reasonably priced with respect to increasingly inexpensive hardware costs.

Virtools VR Solutions

We believe that Virtools VR solutions feature all of the above. As such, it is one of the best choices as you gear up to tackle the topics discussed in this white paper (and many more!).

Virtools Software Suite

The Virtools Software Suite is a complete development platform with an innovative approach to interactive 3D content creation. The Virtools production process facilitates prototyping as well as the robust development needed for full-scale applications. Breaking away from traditional authoring environments, Virtools solutions help optimize timescales and budgets: meet your production requirements on schedule while significantly reducing production costs and overall risks.

Virtools technology combines the power of a solid authoring environment with the versatility of marketing/multimedia development platforms to provide clients with superior, ground-breaking software solutions that suit a wide range of production and trade-specific applications needs - together with a state-of-the-art render engine.

Virtools Dev



Virtools Dev is the core platform for creating highly interactive 3D applications. Virtools Dev authoring software is built on the Virtools Behavioral Engine and offers an innovative graphical user interface for intuitive

programming and production. Use Dev's standard library of behavior building blocks (BBs) to create complex interactivity, or customize and embed your own components with the Virtools Dev SDK.

Virtools VR Pack Development Edition: The VR Add-on Package for Virtools Dev

Virtools VR Pack greatly simplifies authoring complex VR systems using Virtools Dev, to build sophisticated immersive experiences using industry-standard VR peripherals and PC-based distributed computing (clusters). Virtools' VR solution specifically addresses the following needs: distribution of Virtools scene rendering on PC clusters, management of VR input devices such as 3D trackers, and management of VR displays including VR headsets, multi-screen, active/passive stereo, etc.

Virtools VR Pack Publishing Edition

Virtools VR Pack Publishing Edition is required to publish or publicly display content created with the Development Edition on virtual reality devices and displays (cubic rooms, panoramic rooms, image walls, etc.). The Publishing Edition is comprised of two separate licenses, both of which allow for an unrestricted number of applications:

- Virtools VR Pack / Publishing Cluster: valid for one cluster with an unlimited number of computers,
- Virtools VR Pack / Publishing Player: one per computer in the cluster.

References

[1] Commodity-Based Projection VR

Dave Pape – Buffalo Univ., Josephine Anstey – Buffalo Univ., Bill Sherman – Univ. of Illinois at Urbana-Champaign, Siggraph'04 workshop.
<http://resumbrae.com/s04>

[2] OmniStereo for Panoramic Virtual Environment Display Systems

Andreas Simon - Fraunhofer IMK, Randall Smith - General Motors R&D, Richard Pawlicki - R.R.P. and Associates in [IEEE Virtual Reality 2004 \(VR'04\), p 67](#)

[3] Fakespace Labs Website

<http://www.fakespacelabs.com/papers/stanford%20two%20viewer.pdf>
<http://www.fakespacelabs.com/papers/cga2004.pdf>

[4] Surround-screen projection-based virtual reality: The design and implementation of the CAVE.

Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti.. *Proceedings of ACM SIGGRAPH'93*, pages 135.142, August 1993.

<http://portal.acm.org/citation.cfm?doid=166117.166134>

[5] Barco stereoscopy brochure

http://www.barco.com/projection_systems/downloads/Barco_stereoscopic_proj.pdf

[6] Advanced Realtime Tracking, <http://www.ar-tracking.de/>

[7] SGI Prism,

<http://www.sgi.com/products/visualization/prism/>

[8] Orad Dvg,

<http://www.orad.co.il/graphic.htm>

[9] Wildcat Multiview,

<http://www.3dlabs.com/product/technology/Multiview.htm>

[10] NVIDIA Fx 3000G / Fx 4400G,

http://www.nvidia.com/page/quadrofx_3000g.html

[11] VRPN

<http://www.cs.unc.edu/Research/vrpn>

[12] Softgenlock,

<http://sourceforge.net/projects/softgenlock>