An Overview of Cluster Solutions for Immersive Displays

Anthony Steed, Mashhuda Glencross, Allen Bierbaum
Orignally published in Presence: Teleoperators and Virtual Environments, 12(4), 437-440, MIT Press.

Consumer graphics cards are now so powerful that many virtual environment groups are considering moving to a cluster solution to drive their immersive installations. The use of off the shelf components is attractive for many academic and industrial groups because of the installation, direct maintenance costs and potentially swifter upgrade path. However, cluster solutions generate their own problems in software distribution and maintenance and they are not yet common.

Several cluster solutions have been built and tested in laboratories and commercial solutions have started to appear. This review will look at three cluster solutions and compare some of the technical requirements and user experience with them: one mature installation, the Fraunhofer Institute for Industrial Engineering IAO's HiPI-6; one commercial system, the ARS Electronica ARSBox; and open source solutions based on VR Juggler.

Large wall installations have been demonstrated working in a number of different rendering configurations. In sort-first style processes, a single image is partitioned into tiles, each image generator renders one tile and then all of the tiles are composed into a single image. In sort-last style processes, each image generator is responsible for rendering, with depth, a set of graphics primitives and subsequently these images are composed with depth to form the final image. Toolkits such as Stanford's Chromium provide good support for these and other hybrid solutions for the situation where a single virtual camera can be assumed. In Chromium for example, a large subset of existing OpenGL applications can seamlessly be moved from single image generator to multiple image generators without any application-level customisation.

Immersive systems pose new challenges for cluster rendering systems. With a large wall display, a single viewpoint and third-person view control are sufficient for user interaction, In these conditions conventional desktop applications can be run unaware of the underlying rendering solution. In contrast, immersive systems usually require more than one display surface and projection system, stereo imagery and head tracking. With the egocentric and high-field of view required for immersive graphics matters, we can no longer utilise a single desktop viewing configuration because we need to make sure that a complete set of graphics primitives is generated so that any required view can be generated and nothing is culled away because of the application's reliance on view-volume or visibility culling from a single centre of projection.

Thus toolkits that target immersion usually work by distributing and synchronizing instances of an application or real-time distribution of a database of graphical primitives. Application instancing has the advantage that it seems simple to set up. However all but the most trivial applications have interactive capabilities that generate issues with synchronisation of behaviour across all processes. A common alternative is the scene-graph style distribution. In this mode, an application generates edit events on a scene-graph structure. Enforcement of a particular scene-graph paradigm restricts the application programmer's freedom, but does provide a simple framework for the distribution side.

A further level of synchronisation is required at the image level in order to make sure tearing and other artefacts are not seen across projection surface edges. Existing systems either use a software synchronisation, a custom hardware signal or a combination of both. Details of the various possibilities here, such as genlocking, swaplocking & pixel-locking, are beyond the scope of this review, but

will be the subject of a future review. Commercial solutions to this problem are becoming available now, and each of the example systems tackles synchronisation in a different way.

Fraunhofer IOA HyPi-6

The HyPi-6, which was installed in May 2001, is novel in many ways. It was one of what is still only a handful of six-walled CAVEs and it was the first that could operate in both passive and active stereo modes. The active stereo mode is driven by an SGI Onyx 3000 system with six IR3 pipes. The passive stereo mode is driven by 12 standard PCs running Linux with 1.2 GHz AMD Athlons, Geforce 3 (Elsa Gladiac 920). The active mode has a resolution of 1024x1024 per wall and the passive mode 1400x1400.

Image level synchronisation stuff is achieved with serial connection among the nodes. One node is the master and is equipped with a multiport serial card. A signal amplifier is required to get reliable signals to all nodes. This was solution developed in house and it is commercially available from a spin-off, ICIDO.

Applications for the cluster are written with in the in-house Lightning software. Lightning is based on IRIX Performer and on the SGI, Performer's own multi-processing capabilities are used to drive multiple image generators. On the cluster, Lightning uses a higher-level event distribution system to propagate application changes between cluster nodes.

The SGI is still the preferred choice for applications that require high image quality. This is both because of the inherent properties of the SGI image generators and because this is an active stereo mode. With the passive mode, there is some ghosting due to the circular polarisation needed for the passive mode. However the cluster is preferred in geometry or texture heavy applications where actual image quality is not so important as speed.

Figure 1: The HyPi-6 system. Images Courtesy of Fraunhofer Institute Industrial Engineering, Competence Center Virtual Environments

ARS Electronica Futurelab's ARSBOX

FutureLab markets the ARSBOX as a low cost CAVE and a high-end multimedia environment. Importantly, each display can be independently controlled which means that users could for example run a demo using one screen, a presentation on another, and play a video on the third.

At SIGGRAPH 2002 FutureLab demonstrated a system consisting of three screens making up a large panorama. Display graphics were generated by six Linux PCs each with nVIDIA graphics cards. Each image generator drove one projector in an active stereo mode. The video output from pairs of image generators was synchronised using a dedicated piece of hardware designed in house.

All the PCs were networked and controlled by a master node, with one additional Linux PC used to track user's input. For multi-media presentations, an optional PC with Microsoft Windows and Powerpoint is incorporated into the ARSBox. The system is controlled using a hand-held device (called the Palmist) which is based on an IPAQ but with tracker hardware. The Palmist acts both as display control, allowing the user to start and stop applications and map them to single or multiple walls, as well as control for the applications themselves by providing navigation, selection and mapping information. The ARSBOX system uses the CAVELib application

framework and Performer for scene-graph layer.

FutureLab sell the ARSBox in a variety of different configurations. They say that the system is scalable allowing up to 64 display walls. An ARSBOX similar to the one shown at SIGGRAPH costs around €35K.

Figure 2: The ARSBox Demonstration at SIGGRAPH 2002. Images Courtesy of ARS Electronica FutureLab

VR Juggler Based Solutions

VR Juggler is an Open Source suite of tools that provide a platform for VR application development. VR Juggler based systems allow for a variety of cluster solutions using all of the clustering methods discussed above.

One of the first VR Juggler clustering methods used WireGL (now renamed Chromium). This solution runs the immersive application on a single master server machine and uses the other machines in the cluster as rendering nodes. VR Juggler is used to configure and handle the input devices, display surfaces and viewing parameters. VR Juggler then in turn uses Chromium to send the correct graphics commands to each of the rendering nodes. This clustering method can be applied to current VR Juggler applications without requiring any application changes, though note the caveats for this general approach in the introduction.

VR Juggler also supports clustering based on input distribution using ClusterJuggler. ClusterJuggler starts an individual copy of an application on each node of the cluster. The system makes sure that all nodes receive the same set of device inputs each frame. This guarantees that the applications stay synchronized as long as they base all their computations upon the user inputs.

ClusterJuggler allows for a high degree of configurability. A single configuration file is used to configure the entire cluster. The configuration not only configures the normal VR system setting such as input devices and display settings, but it also selects the clustering methods used for data distribution and synchronization. Users can choose to distribute data using UDP or TCP networking. They can also choose among multiple synchronization methods including: network-based (UDP or TCP), shared serial connection, custom serial hardware (similar to the method used by HyPi-6), or a hybrid approach mixing multiple methods.

NetJuggler provides another VR Juggler based clustering solution. It was created at the Universite d'Orleans and is designed to complement more traditional computation clusters. Similarly to ClusterJuggler, it is based upon the idea of running a copy of the same application at each cluster node and distributing the user input to each application. The major difference is that it uses MPI for data distribution and synchronization. This allows the same cluster that is controlling the VR environment to be used for computation clustering for the application. This has definite advantages for applications that need the processing power and have been designed to be run in parallel with MPI.

VR Juggler based solutions that work by distributing scene graph updates also exist. These solutions usually make use of either ClusterJuggler or NetJuggler to handle the synchronization and distributing the viewing calculations.

VR Juggler based clustering solutions are being used to support a large number of immersive installations world-wide. These installations include systems at Iowa State University, Universite d'Orleans, UC Davis, University or Western Sydney, University of Kentucky, UC Davis, NRL, HRL Labratories, and the VR Media Lab. The

Iowa State University cluster was recently shown at Supercomputing 2002 and there are several groups that are planning to show systems at IEEE VR 2003.

Discussion

The most striking similarity between the systems is their reliance on low-cost, Linux-based image generators. The stability of graphics drivers and availability of quad-buffered rendering on Linux boards is driving many groups' development. Open source software is an additional boost to development. Recent hardware developments allow active stereo rather than passive stereo solutions with clusters. The option of active stereo solves some of the cross-talk issues with current passive stereo solutions.

Synchronisation is perhaps the biggest problem facing cluster solutions. Systems require both software and hardware synchronisation. Software-only solutions are likely to produce unacceptable tearing artefacts. Future reviews we will look in more depth at technologies simplify the process of setting up an immersive system.

Further Resources

A tutorial on cluster solutions was presented at ACM SIGGRAPH 2002. A workshop on clusters will take place at the IEEE VR2003 conference. We encourage readers to submit overviews of their successful experiences with clusters to www.presence-connect.com.

- ICIDO (www.icido.de)
- HyPi-6 (vr.iao.fhg.de/6-Side-Cave/index.en.html)
- VRJuggler (www.vrjuggler.org)
- ARSBox (futurelab.aec.at/arsbox/)
- CAVElib (www.vrcom.com)

Acknowledgements

With thanks to Dr. Reinhold Plösch (ARS Electronica FutureLab) and Dr. Hilko Hoffmann (Fraunhofer Institute Industrial Engineering, Competence Center Virtual Environments, Nobelstr. 12, D-70569 Stuttgart) for data and permissions to use images.

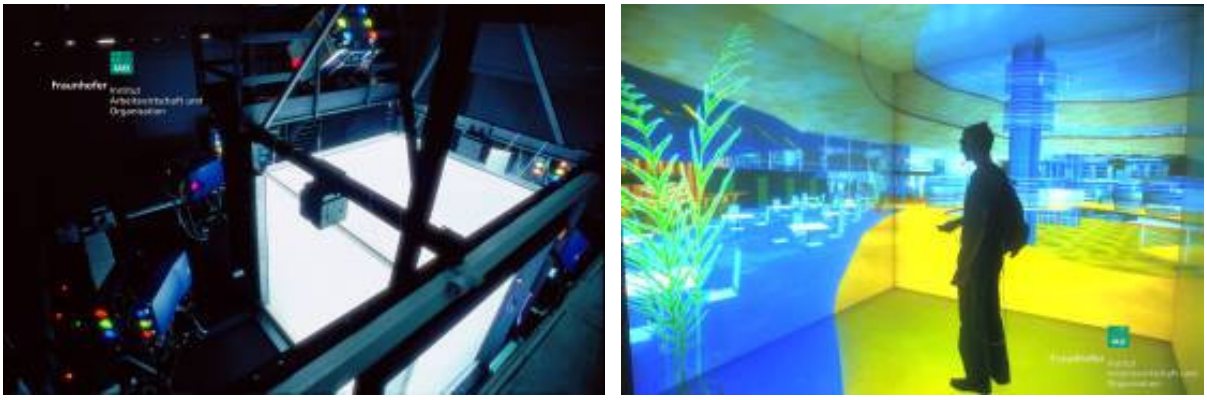

Figure 1: The HyPi-6 system. Images Courtesy of Fraunhofer Institute Industrial Engineering, Competence Center Virtual Environments

Figure 2: The ARSBox Demonstration at SIGGRAPH 2002. Images Courtesy of ARS Electronica FutureLab