Infraestructuras de bajo costo para la visualización de datos a gran escala y resolución, una opción accesible para instituciones educativas

Low cost infrastructure for large scale and high resolution data visualization, a viable option to educational institutions

Jesús Alberto Verduzco Ramírez
Instituto Tecnológico de Colima
javrtesis@yahoo.com.mx

Nicandro Farías Mendoza
Instituto Tecnológico de Colima
nmendoza@ucol.mx

Gilberto René Martínez Bonilla
Instituto Tecnológico de Colima
subdiracademica@itcolima.edu.mx

Pedro Rocha Medrano
Instituto Tecnológico de Colima
procha@itcolima.edu.mx

María Isabel Sáenz Rodríguez
Instituto Tecnológico de Colima
su_saenz16@hotmail.com

RESUMEN

La necesidad de disponer de superficies de visualización gráfica de datos de mayor tamaño y resolución que aquellas proporcionadas por el monitor se hace evidente al utilizar aplicaciones que generan cantidades significativas de datos gráficos como: la visualización científica, los entornos de la realidad virtual y aumentada, el diseño en ingeniería, el análisis de gráficas en finanzas, etcétera. El presente capítulo tiene como objetivo, en la primera parte, hacer una reseña de las tecnologías propietarias en el área de las plataformas de visualización intensiva de datos que emplean alta escala y

alta resolución. En la segunda parte, se presentan las experiencias al construir tres sistemas de visualización de bajo costo, que pueden ser utilizadas como reales alternativas a las costosas infraestructuras propietarias en apoyo a las actividades académicas en instituciones de educación con presupuesto limitado.

Palabras clave: visualización de datos, proceso gráfico distribuido, cluster, muro de imágenes, cave.

ABSTRACT

The need for surfaces of graphical data visualization of larger size and resolution that those provided by the monitor becomes evident when using applications that generate significant amounts of graphics data as: scientific visualization, the environments of virtual and augmented reality, design engineering, analysis of graphs in finance, etc. This chapter's objective, in the first part, do a review of proprietary technologies in the area of the platforms of intensive data visualization employing high scale and high resolution. In the second part, experiences are presented to build three systems of low cost, which can be used as real alternatives to expensive proprietary infrastructures in support of academic activities in educational institutions with limited budget.

Key Words: data visualization, distributed rendering, cluster, display wall, cave.

Fecha recepción: Septiembre 2013 **Fecha aceptación:** Octubre 2013

INTRODUCCIÓN

El monitor es el dispositivo normalmente utilizado para mostrar información gráfica al usuario de una computadora. Sin embargo, desde el punto de vista de sus características físicas, los monitores han evolucionado lentamente. Los monitores LCD remplazan actualmente a los monitores CRT, pero esto se produce desde hace solo algunos años. En contraste, la mayoría de los componentes de la PC, tales como la memoria, el procesador y los discos han visto sus capacidades prácticamente duplicarse todos los años, mientras que la resolución de los monitores se ha incrementado solamente a un ritmo del 5 % por año los dos últimos decenios (Chen Y., Chen H., Clark, D.W., Liu, Z., Wallace, G. & Li, K ,2001)

La resolución y dimensiones del monitor constituyen un obstáculo para cierto tipo de aplicaciones que tienen como característica principal la generación de importantes cantidades de datos gráficos. La visualización científica de datos, los entornos de la realidad virtual y aumentada, el diseño en ingeniería y el análisis de gráficas en finanzas, son ejemplos de aplicaciones que requieren superficies de visualización gráfica de mayores dimensiones que aquellas proporcionadas por el monitor. Ejecutándose en un sistema con reducida potencia gráfica, una aplicación no puede explotar toda su capacidad en la generación del entorno gráfico. Como consecuencia, la visualización de gráficas complejas y el proceso de interacción con el usuario son fuertemente perturbados. Por ejemplo, un usuario al interactuar con una aplicación que despliega mapas, se ve obligado a visualizar fracciones de la imagen total si desea analizar detalles de la imagen, esto implica la pérdida de la visión del contexto global, situación que se agrava aún más si participan múltiples usuarios.

El interés por aumentar las dimensiones y resolución de la superficie de visualización ha estado presente en años recientes. Las soluciones actuales utilizan una arquitectura centralizada o una distribuida. El enfoque centralizado consiste en dotar a un solo nodo de múltiples tarjetas gráficas y con ello alimentar a más de un monitor. Como ventaja, disponemos de una alternativa de bajo costo y de fácil implementación; sin embargo, este esquema sufre de algunos inconvenientes. Un solo nodo efectúa el proceso gráfico (*renderización)*, lo que implica baja eficiencia. Un problema mayor es el número limitado de puertos PCI y AGP disponibles en el nodo. Una solución más atractiva es dividir el proceso gráfico entre un grupo de nodos interconectados por una red de comunicación de datos y distribuir las imágenes y las primitivas gráficas utilizando un protocolo eficiente. En este modelo los problemas principales son el consumo excesivo del ancho de banda de la red y la sincronización de las actividades de los nodos. A pesar de estos inconvenientes, actualmente los proyectos en desarrollo emplean como solución la distribución del proceso gráfico.

Los sistemas de visualización gráfica basados en video proyectores tales como: Display Wall (Li K., Chen H., Chen Y., Clark D.W., Cook P., 2000), WorkBench (Wolfgang Krüger, Christian-A. Bohn, Bernd Fröhlich, Heinrich Schüth & Wolfgang Strauss, 1995) y CAVE (Cruz Neira C., Sandin Daniel J. & Defanti Thomas A. 1993), han sido ampliamente utilizados en la visualización de datos.

Estos sistemas permiten visualizar las imágenes de aplicaciones en alta resolución y gran tamaño acompañadas de visión estereoscópica para incrementar el grado de inmersión de los usuarios. En las secciones siguientes se hace un análisis de estas tres soluciones que utilizan este enfoque.

Display Wall

Un Display Wall es un sistema capaz de presentar imágenes en alta resolución y gran tamaño (Li K. et al. 2000). La imagen global es generada sobre una o varias superficies por un conjunto de video proyectores, los cuales son alimentados por las salidas gráficas de una supercomputadora tipo SGI Onyx. En la figura 1 se muestra un Display Wall compuesto por ocho superficies independientes iluminadas por video proyectores. Cada video proyector es responsable de generar una fracción de la imagen global proyectada. Un Display Wall permite visualizar imágenes 2D en grandes superficies, lo que facilita el análisis detallado, por ejemplo, en medicina, imágenes satelitales, arquitectura, etcétera.

Figura 1. Display Wall generado por un arreglo de ocho video proyectores

Responsive Workbench

El Responsive Workbench (Krueger W. et al., 1995), es una herramienta que permite generar imágenes estereoscópicas 3D que son proyectadas en dos superficies de visualización perpendiculares con la ayuda de un sistema formado por un proyector y espejos. El usuario puede interactuar con las aplicaciones desarrolladas por medio de

guantes y un ratón 3D. Además, un sistema de seguimiento monitoriza la posición de la cabeza del usuario para que este visualice el ambiente virtual desde el ángulo correcto.

El workbench es una herramienta muy apreciada en el trabajo con maquetas electrónicas por la cercanía que tiene con las manos del usuario. La Figura 2 muestra a un usuario utilizando el Workbench.

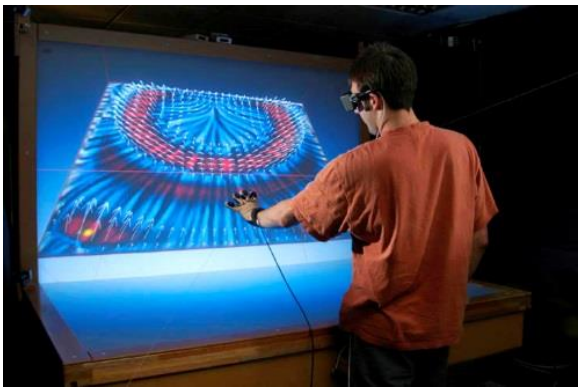

Figura 2. Interacción con una aplicación 3D ejecutándose en el Workbench

CAVE

La CAVE es un sistema de visualización gráfica 3D de alto desempeño constituida por un cubo de seis caras. Las caras son iluminadas desde el exterior por video proyectores alimentados por los canales gráficos de una supercomputadora, en la cual se ejecutan las aplicaciones que producen las gráficas desplegadas sobre las superficies de la CAVE. Los usuarios situados al interior de la CAVE provistos de lentes 3D, prácticamente se encuentran rodeados de imágenes acompañadas de estéreo lo que contribuye a incrementar el grado de inmersión. Las aplicaciones de las CAVE se centran en las visualización de modelos a escala de fenómenos físicos.

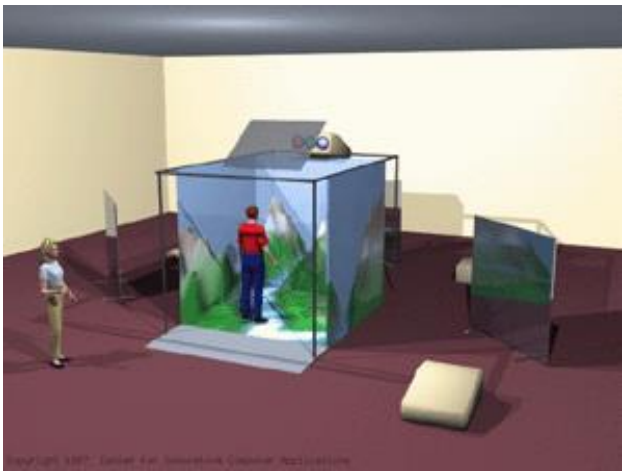

Figura 3. Estructura de una CAVE

SOLUCIONES DE BAJO COSTO

Tradicionalmente para aplicaciones de visualización intensiva de datos se han utilizado equipos de alto desempeño dotados de tecnología propietaria. Los tres sistemas analizados en la sección anterior CAVE, Display Wall y Workbench, comparten una característica en común: utilizan tecnología propietaria tales como video proyectores 3D y supercomputadoras de la clase SGI Onyx que redunda en un significativo incremento en sus costos haciéndolos inaccesibles para organizaciones con presupuesto reducido. Una alternativa atractiva, es el uso de Clúster de computadoras, es decir, utilizar el poder de procesamiento agregado de un conjunto de equipos de cómputo, así como combinar su capacidad de almacenamiento y procesamiento gráfico como una alternativa económica a las costosas supercomputadoras.

La rápida evolución, incremento del desempeño y tendencia al bajo costo de los componentes de las PC incluyendo las tarjetas gráficas, convierten a los Clúster en una alternativa viable para aplicaciones que requieren procesamiento intensivo de datos. Las infraestructuras de este tipo, además, resultan económicas debido a que es posible

reutilizar equipo existente en las organizaciones, por ejemplo, en las instituciones educativas.

PLATAFORMAS PARA LA VISUALIZACIÓN DISTRIBUIDA DE BAJO COSTO

Actualmente existen varias plataformas que siguen el enfoque de visualización distribuida integrada con componentes de bajo costo, una de las más sobresalientes es Grimage (Jérémie Allard, Clément Menier, Bruno Raffin, Edmond Boyer & François Faure, 2007), la cual consiste en cuatro elementos: un clúster de procesamiento gráfico, un sistema de adquisición de imágenes, una librería para el desarrollo de aplicaciones paralelas y una superficie de visualización de grandes dimensiones. En esta plataforma, es posible adquirir un objeto para que sea instantáneamente modelado en 3D e ingresado en un mundo virtual con el cual es posible interactuar. La Figura 2 presenta imágenes de Grimage.

Figura 2. Funcionamiento de la Plataforma Grimage

En la figura 3 se presenta con detalle un esquema del funcionamiento de Grimage. Primero, se capturan flujos de video provenientes de un conjunto de cámaras, se adquiere únicamente el objeto requerido y se sustraen todos los elementos de fondo, de esa manera se crea un modelo en 3D del objeto capturado y se renderiza en una superficie de visualización de grandes dimensiones.

Figura 3. Esquema de funcionamiento de Grimage

Grimage requiere para su funcionamiento un equipo de alto rendimiento, la configuración existente en el INRIA Rhone Alpes consiste en 6 cámaras de alta definición, un muro de imágenes de 16 video-proyectores y 2 Clústers interconectados, uno con 11 dual-Xeon a 2.6 GHz para la aplicación y otro con 16 dual-Opteron a 2 GHz para la parte de visualización. Para la distribución de los procesos y el envío de mensajes entre estos, Grimage utiliza una librería de software llamada FlowVR, la cual se describe en la siguiente sección.

FlowVR

FlowVR (Jérémie Allard, Valérie Gouranton, Sébastien Liimet, Emmanuel Melin, Bruno Raffin & Sophie Robert, 2004) es una librería que provee a los usuarios con las herramientas necesarias para desarrollar y ejecutar aplicaciones interactivas de alto desempeño en Clúster s y Grids computacionales. Las principales aplicaciones de esta librería son aquellas soluciones de realidad virtual y visualización científica. FlowVR

facilita la programación modular que disminuye las dificultades de ingeniería de software mientras permite ejecuciones de aplicaciones de alto desempeño en ambientes paralelos y distribuidos. La arquitectura y aplicaciones de FlowVR se muestran en las y 7.

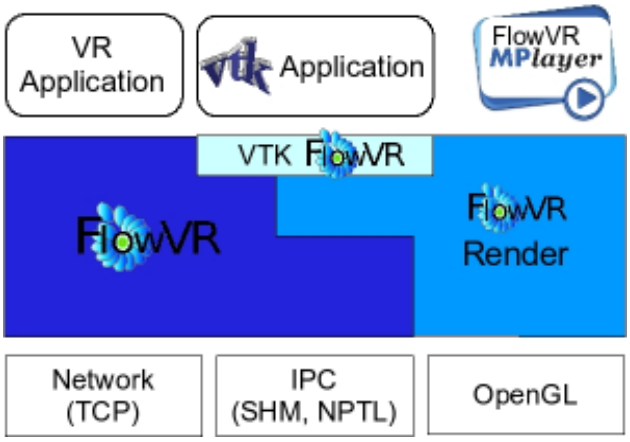

Figura 4: Arquitectura de FlowVR

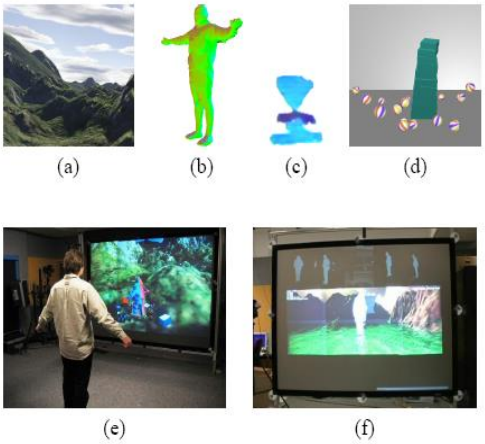

Figura 5. Diferentes aplicaciones de FlowVR

En la Figura 5 se muestran diferentes aplicaciones de FlowVR:

(a) renderizado de terrenos, (b) modelado de un cuerpo con cámaras, (c) distorsión de objetos, (d) simulación de cuerpos rígidos, todas pueden ser combinadas en una aplicación ejecutándose en un Clúster.

IMPLEMENTACIÓN DE SISTEMAS DE VISUALIZACIÓN DE BAJO COSTO

En esta sección se describe la construcción de tres prototipos que utilizan para su funcionamiento componentes de bajo costo. Nosotros definimos el bajo costo, como la utilización de componentes de arquitectura abierta tales como PC, video proyectores,

Clúster de computadoras, también llamados *commodity components*, como una alternativa a las costosas infraestructuras propietarias tales como las supercomputadoras.

Prototipo 1: Muro de imágenes de bajo costo

En esta sección describimos el trabajo desarrollado para implementar un muro de imágenes utilizando componentes de bajo costo. Un Muro de Imágenes es un sistema capaz de visualizar imágenes en alta resolución sobre grandes superficies. La imagen global es generada sobre una o varias superficies por un conjunto de video proyectores, los cuales son alimentados por las salidas gráficas de los nodos que constituyen un Clúster. En la figura 8 se muestra un Muro de Imágenes compuesto por cuatro superficies independientes (nombradas *tiled*). Cada nodo del Clúster es responsable de generar la imagen correspondiente a cada *tiled* y la red de transmisión de datos traslada primitivas gráficas y comandos de sincronización entre ellos. En el Clúster existen uno o más nodos que permiten la interacción del usuario con las aplicaciones, ya sea utilizando el teclado y el ratón o por otros medios como el reconocimiento de gestos, el rastreo de la posición del usuario, etcétera.

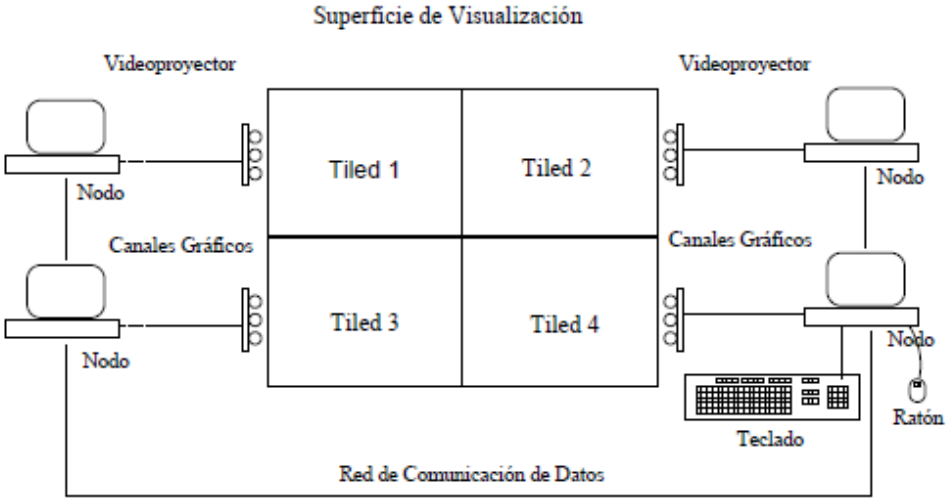

Figura 6. Arquitectura de un muro de imágenes de bajo costo

La solución proXy

En esta sección nosotros describimos el trabajo desarrollado para distribuir el display del Servidor X Window (Robert W. Scheifler & Jim Gettys, 1986) y visualizar las gráficas de una aplicación X Window (clientes X) sobre múltiples nodos de un clúster. Los mensajes producidos por los clientes X necesitan ser modificados y distribuidos entre el grupo de nodos. Similarmente, los mensajes de retorno, producto de la interacción del usuario deben de ser recuperados y dirigidos al cliente X. La arquitectura material del sistema (ver Figura 9) está constituida por:

Nodo maestro: Aloja una instancia en ejecución del proXy. A este nodo los clientes deben imperativamente conectarse, utilizando para ello el *socket* en el cual escucha el proXy.

Nodos de renderización: Ejecutan una instancia del Servidor X Window y tienen como misión recibir las peticiones gráficas enviadas por el cliente por medio del proXy, efectuar la *renderización* y su posterior visualización.

Cada nodo efectúa la *renderización* de la fracción de la imagen que le corresponde visualizar. Con este procedimiento, el conjunto de monitores o de videoproyectores, ordenados bajo un arreglo particular presentan una sola superficie de visualización en la que se despliegan las imágenes de los clientes.

Nodo de interacción: Habilita los dispositivos de entrada de datos, teclado y ratón, mediante los cuales el usuario puede interactuar con las aplicaciones. Este nodo también efectúa la visualización de gráficas.

A. Display virtual

La primera tarea de nuestro prototipo es efectuada al momento de su lanzamiento, consiste en definir la configuración del *display virtual*. El usuario proporciona en la línea de comando la lista de nodos de renderización y la disposición que guardarán en el *display virtual*. El proXy envía *primitivas gráficas* de apertura de display (*XOpenDisplay*) a cada servidor de renderización con la finalidad de recuperar sus características gráficas, particularmente las dimensiones del display. Con esta información, el proXy construye y determina la resolución del *display virtual*, por ejemplo: 4 servidores de renderización A,B,C y D cuyos displays son de *1,280x960* pixeles y la disposición forma un arreglo de *2x2*, formarán un *display virtual* de *2,560x1,920* pixeles, como el que se muestra en la figura 9. Al momento en que un cliente X se conecta, el proXy le envía las características del *display virtual*. Con esta información, el cliente X cree disponer de una superficie de visualización de mayores dimensiones que aquella proporcionada por el monitor de la PC en la cual está en ejecución.

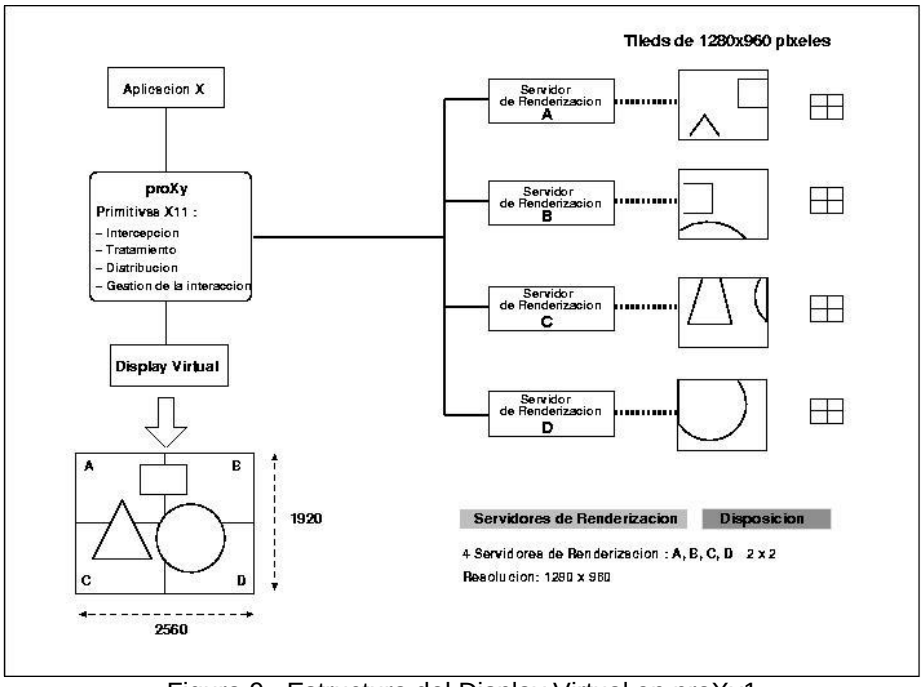

Figura 9. Estructura del Display Virtual en proXy1

B. El flujo de renderización

Nosotros desarrollamos dos estrategias para el tratamiento de las *primitivas gráficas*. La primera estrategia es aplicada por el prototipo llamado proXy1 y consiste en la intercepción y adaptación de las coordenadas que transportan las primitivas *gráficas* con el fin de adaptarlas al contexto del *display virtual*. Esta solución se expresa por la expresión siguiente:

$$X(u,v) = Xvirtual - U * X \qquad (1)$$

$$Y(u,v) = Yvirtual - V * Y \qquad (2)$$

Donde (1) X, Y identifican la resolución del *display virtual (u,v)* la posición del servidor de renderización numerada a partir de (0,0) , (X*virtual*, Y*virtual*) las coordenadas de un punto en el *display virtual* y (X*(u,v)*, Y*(u,v)*) las coordenadas correspondientes con

respecto al servidor de renderización *(u,v).* Gracias a las acciones de la función *clipping* los nodos de renderización realizarán únicamente la renderización de la parte del *display virtual* que cada uno administra. La figura 9 muestra este procedimiento.

La (2) segunda solución es utilizada por el prototipo llamado proXy2, esta propone en términos generales descentralizar las funciones de la adaptación de las *primitivas gráficas* hacia los nodos de renderización. Para este fin, proXy2 construye una ventana sobre el *display* de cada nodo de renderización. La característica especial de esta ventana es que cuenta con una dimensión similar al *display virtual.* Esta ventana es posicionada en cada nodo de renderización con el intervalo de coordenadas del *display virtual* que cada *nodo renderización* es responsable de visualizar, el resto de la ventana es eliminada por la función *clipping.*

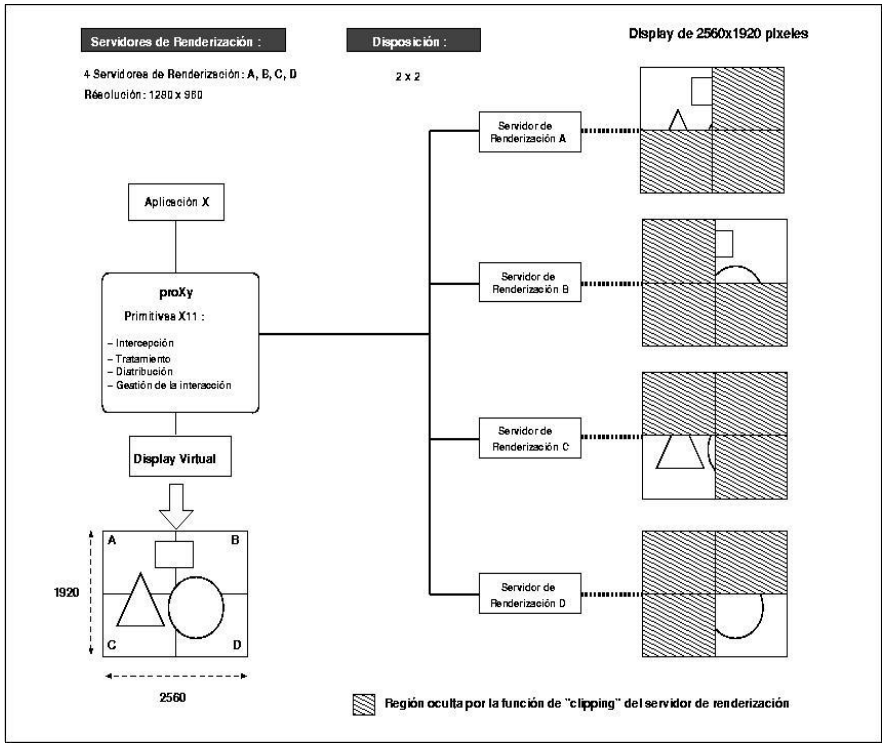

Fig. 10. Construcción del Display Virtual en la solución proXy2

C. El flujo de interacción

Para recuperar las acciones del usuario sobre el ratón y el teclado, se interceptan los mensajes provenientes del nodo de interacción. El servidor X dispone del mecanismo de *focus* para determinar la ventana hacia la cual los resultados de la renderización e interacción deben ser dirigidos.

Mientras que el *focus* permanezca confinado en el *tiled* del nodo de interacción, los eventos del teclado y del ratón son tratados localmente. La situación cambia cuando el usuario arrastra el cursor del ratón a un *tiled* vecino. Entonces los eventos generados en el servidor de interacción deben ser dirigidos hacia el nodo de renderización que posee el *focus* en ese momento. Nosotros utilizamos la extensión XTEXT presente en el servidor X Window que permite enviar eventos hacia un *display* remoto. Nuestra solución está basada en este código. Los resultados se muestran en la Figura 11.

Fig. 11 Muro de imágenes de bajo costo en el Tecnológico de Colima

Discusión de resultados

En esta sección mostramos el desarrollo de un servidor proXy que recibe mensajes de clientes del sistema X Window y las distribuye hacia un conjunto de servidores X, los cuales cooperan para construir la imagen del cliente, obteniendo como resultado gráficas de mayores dimensiones y mayor resolución.

Para distribuir el display del servidor X Window, nosotros utilizamos dos estrategias. La primera consiste en adaptar las *primitivas gráficas* al contexto del *display virtual*, esta tarea es realizada en el prototipo llamado proXy1. La segunda estrategia es implementada en el prototipo proXy2 y consiste en realizar el proceso de adaptación en los nodos de renderización.

Con esta implementación, el usuario se beneficia al disponer de una herramienta que despliega las gráficas de las aplicaciones del sistema X Window sobre grandes superficies, aumentando la cantidad de información que puede ser percibida y mejorando el proceso de interacción con el usuario. Una de las ventajas de esta solución, es que puede implementarse con elementos presentes en instituciones académicas, por ejemplo, el Instituto Tecnológico de Colima, con ello se dispondrá de una infraestructura para visualizar imágenes 2D en grandes dimensiones.

II.- Prototipo 2: Plataforma de bajo costo para interacción con objetos virtuales

Esta sección presenta nuestro segundo desarrollo, una plataforma para ejecutar aplicaciones de realidad aumentada.

Nuestra solución está estructurada en dos módulos funcionales:

A. Un módulo para la adquisición y distribución de las imágenes capturadas con cámaras Web.

- Un módulo de renderización para el tratamiento del flujo de video y mezcla con escenas en 3D interactivas.

En esta primera fase, para comprobar la factibilidad de nuestra idea, se implementó un prototipo integrado por solamente dos nodos, el nodo de adquisición y el nodo de renderización, interconectados por una red, como se muestra en la Figura 12.

El Nodo de adquisición

El nodo de adquisición conecta una o dos cámaras Web, las cuales adquieren imágenes del entorno. Un servidor de video almacena y transmite dichas imágenes hacia el nodo de renderización.

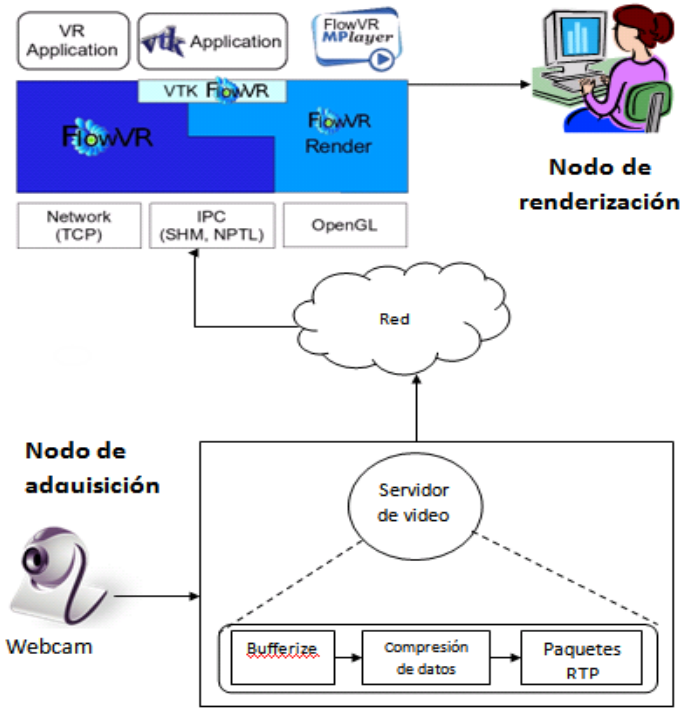

Figura 12. Arquitectura del prototipo de realidad aumentada de bajo costo

El Nodo de renderización

El nodo de renderización ejecuta una instancia de FlowVR, la cual recibe los cuadros de video, efectúa la renderización y posterior visualización por medio del MPlayer inmerso en FlowVR.

Funcionamiento

El nodo de adquisición captura las imágenes y envía el flujo de video hacia el nodo de renderización. Como servidor de video, utilizamos el software Spook (SPOOK, 2008), un software de libre distribución que permite el envío de flujos de video por redes IP. El software Spook toma las imágenes almacenadas en el buffer, las ordena secuencialmente, comprime los datos y los envía por medio de la red. Por su parte, en el nodo de renderización, el cliente inicia el reproductor MPlayer inmerso en FlowVR. De esta manera, es posible visualizar las imágenes capturadas por medio de la cámara Web en el nodo de renderización.

La mezcla de las imágenes capturadas con la cámara Web y la generación de escenas 3D corre a cargo del componente FlowVR Render, el cual permite la combinación del flujo de video con un ambiente virtual renderizado por FlowVR.

Descripción de la aplicación

Para comprobar la funcionalidad de nuestra plataforma, nosotros hemos desarrollado algunos ejemplos en los que se demuestra el reconocimiento de los contornos de los objetos captados por la cámara Web y la interacción con esos objetos sin emplear ningún dispositivo de interacción. Para ambas operaciones se ha utilizado la librería OpenCV (Intel Corporetion, s.f.).

Reconocimiento de contornos

Un contorno es una lista de puntos que representan de una manera u otra, la curvatura de una imagen. Esta representación puede ser diferente dependiendo de las circunstancias. Existen muchas formas de representarlos. En OpenCV los contornos son representados por secuencias en donde cada entrada de dicha secuencia codifica información acerca de la localización del siguiente punto en la curva.

Para conseguir mostrar los contornos de una imagen, primero se necesita un método para el manejo de la memoria. OpenCV cuenta con una entidad llamada Memory Storage, la cual es un método para administrar la ubicación de objetos dinámicos en la memoria. El tipo de objeto que puede ser almacenado dentro de un memory storage es una secuencia. Las secuencias son listas enlazadas de otras estructuras de datos.

El primer ejemplo desarrollado consiste en el reconocimiento del contorno de una serie de letras y dibujos a mano alzada. El resultado se muestra en las siguientes imágenes.
.

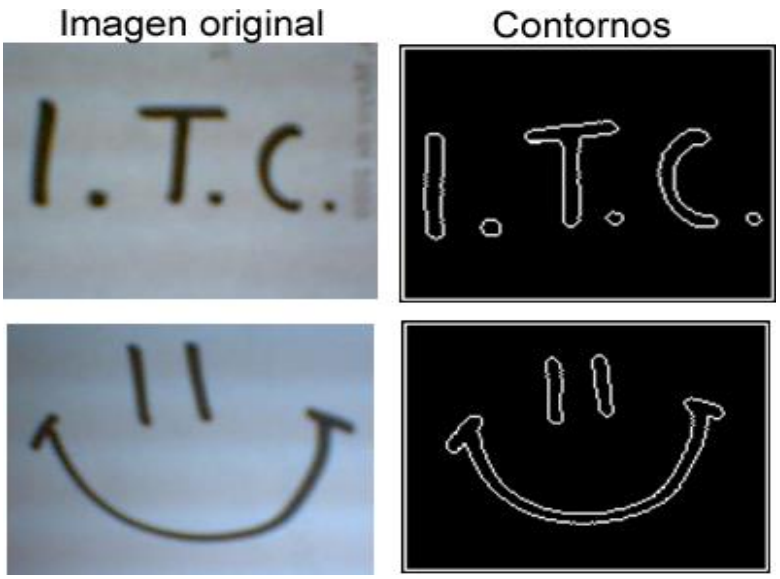

Figura 12. Detección de contornos

Interacción con objetos virtuales

Utilizando este prototipo, es posible capturar objetos utilizando la cámara Web e incrustarlos en una aplicación 3D, lo cual ofrece posibilidades para interactuar con los objetos virtuales que integran la escena 3D. En una aplicación de este tipo nosotros identificamos tres elementos, mostrados en la Figura 7.

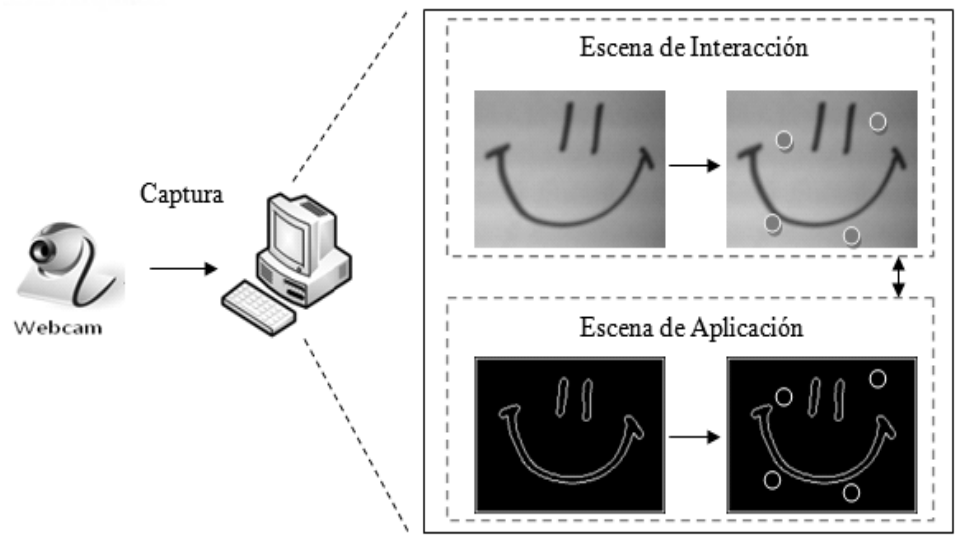

Figura 7. Arquitectura de las aplicaciones desarrolladas

A. La escena de interacción: es representada por el conjunto de imágenes captadas por la cámara.
B. La escena de aplicación: es representada por el conjunto de imágenes de las que se encuentran algunas características de la escena de interacción. Por ejemplo: contornos, esquinas y colores.
C. Finalmente, los objetos virtuales: son los objetos incrustados en las escenas de interacción y aplicación mediante programación, que reaccionan a eventos

dentro de la escena de aplicación y son representados a través de imágenes o texturas en la escena de interacción.

Para comprobar la interacción con objetos virtuales, se desarrolló un par de ejemplos con la intención de mostrar la manera en la que un usuario puede interactuar con ellos. La primera aplicación muestra un conjunto de burbujas que el usuario puede hacer explotar al hacer movimientos en la escena de interacción. La escena de la aplicación consiste en la diferencia entre una imagen y su antecesora. De esta manera, una sombra blanca es formada.

La segunda aplicación de ejemplo muestra un conjunto de pelotas que rebotan dentro de la escena de interacción, cuando el usuario toca alguna o cuando chocan entre sí, modificando su dirección.

La escena de aplicación consiste en los contornos de las figuras que aparecen dentro de la escena de interacción, que al hacer contacto con los objetos virtuales, estos ejecutan una rutina para modificar su dirección de movimiento. Un ejemplo de estas dos aplicaciones se puede observar en la Figura 8.

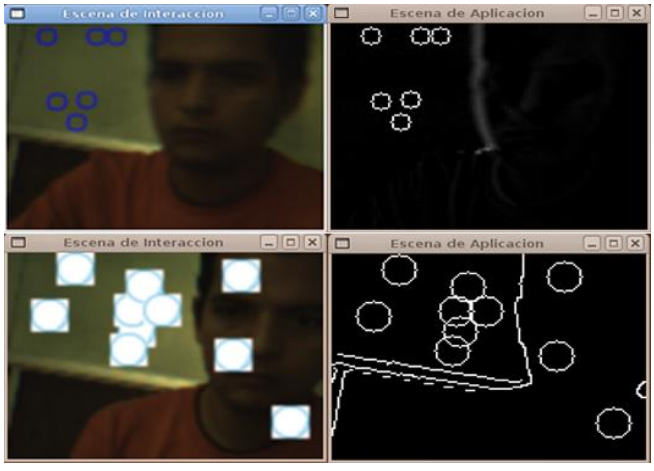

Figura 8. Aplicación que combina imágenes adquiridas y escenas virtuales

III.- Prototipo 3: Implementación de una CAVE de bajo costo

En esta sección describimos el desarrollo de un prototipo de CAVE de bajo costo. Para alcanzar el objetivo de implementar una CAVE, nosotros dividimos el trabajo en cuatro subsistemas que se describen en detalle en las siguientes secciones.

A. Área de visualización

El área de visualización está integrada por un cubo de cinco caras. Nosotros construimos una estructura metálica en forma de cubo de 1.80 m por lado, posteriormente recubrimos este cubo con una tela especial para permitir la proyección de imágenes desde el exterior. Para el ingreso de los usuarios al interior de la CAVE, se implementó un acceso en una de las caras dotado de espacio suficiente para permitir el paso de un usuario a la vez. En esta área de visualización seis usuarios pueden permanecer de pie con cierta comodidad, los detalles se muestran en la Figura 9.

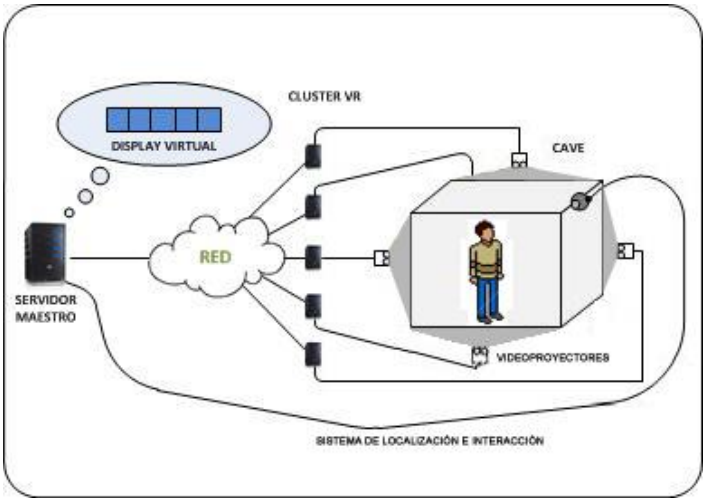

Figura 9: Subsistemas de la CAVE de bajo costo

B. Clúster VR

La ejecución de aplicaciones para la CAVE está soportada en un Clúster VR integrado por seis PC equipadas con 2 GB de memoria RAM y procesador de 1.8 Ghz. Las tarjetas gráficas de las PC son genéricas y no proporcionan soporte para la visualización estéreo. Las PC son interconectadas por una red Fast Ethernet a 100 Mbps. Con la finalidad de reducir los retardos asociados al costo de comunicación en el Clúster VR, nosotros optamos por el modelo ejecución distribuida de copias sincronizadas. En este modelo, cada PC del Clúster VR mantiene una copia completa en ejecución de la aplicación; si bien se tiene la desventaja del consumo excesivo de memoria en cada nodo, se gana en cuanto a la reducción del costo de comunicación. Únicamente es necesario intercambiar mensajes de sincronización entre las copias en

ejecución alojadas en los nodos del Clúster VR, por lo general estos mensajes son de tamaño reducido lo que origina pequeñas latencias. Para el soporte de las aplicaciones en la CAVE, nosotros utilizamos el software *syzygy* (Schaeffer, Goudeseune, 2003) que permite distribuir aplicaciones 3D paralelizando la renderización de las escenas gráficas.

C.- Sistema de proyección

Está constituido por un conjunto de cinco video proyectores localizados convenientemente al exterior de la superficie de visualización, a una distancia y distribuidos en un arreglo de tal manera que iluminen totalmente cada una de las caras de la CAVE. Cada video proyector está conectado a la salida gráfica de cada nodo del Clúster VR. Nosotros utilizamos video proyectores que normalmente se utilizan en las aulas para soporte en la impartición de cursos.

La resolución manejada es de 1,024X768 pixeles. Es importante señalar que estos video proyectores no disponen de la capacidad estéreo.

D.- Sistema de localización e interacción

Para permitir la interacción de los usuarios con las aplicaciones ejecutándose en la CAVE, fue necesario implementar una alternativa a los sistemas de tracking clásicos. Nuestra solución determina la posición de un usuario utilizando las imágenes adquiridas por una cámara Web interconectada al servidor maestro del Clúster VR. Un usuario provisto de un casco en el cual está sujeto un led de alta intensidad pilotea la interacción con la aplicación ejecutándose en ese momento. El área interna de la CAVE adquirida por la cámara Web representa el total de la superficie en la cual se desplazan

los usuarios. Cualquier cambio en la posición del usuario es detectado por la cámara Web y esto significa la regeneración de la escena 3D y su posterior visualización.

Para detectar las modificaciones en las imágenes adquiridas por la cámara, nosotros implementamos un pequeño programa con la ayuda de la librería OpenCV. De esta maner, los usuarios desplazándose al interior de la CAVE pueden navegar por las aplicaciones 3D. En este primer prototipo solo se implementó el soporte para la interacción de un solo usuario.

Pruebas de funcionamiento del prototipo

Para medir la funcionalidad de nuestra CAVE, utilizamos algunas aplicaciones de la librería OpenGL (Mark Segal & Kurt Akeley, 1994). Posteriormente, lanzamos la ejecución en el Clúster VR y encendimos los video proyectores para iluminar la CAVE. Los resultados fueron sorprendentes. A pesar de no utilizar dispositivos con soporte para estéreo, el realismo de las imágenes al interior de la CAVE y la inmersión fueron particularmente alentadores. La ejecución de las aplicaciones en el Clúster VR tuvo un desempeño bastante aceptable. Con respecto al manejo de la localización del usuario y la interacción con las aplicaciones, la solución implementada se desempeñó de manera correcta. En las Figuras 14, 15 y 16 se muestran imágenes de aplicaciones en ejecución en la CAVE.

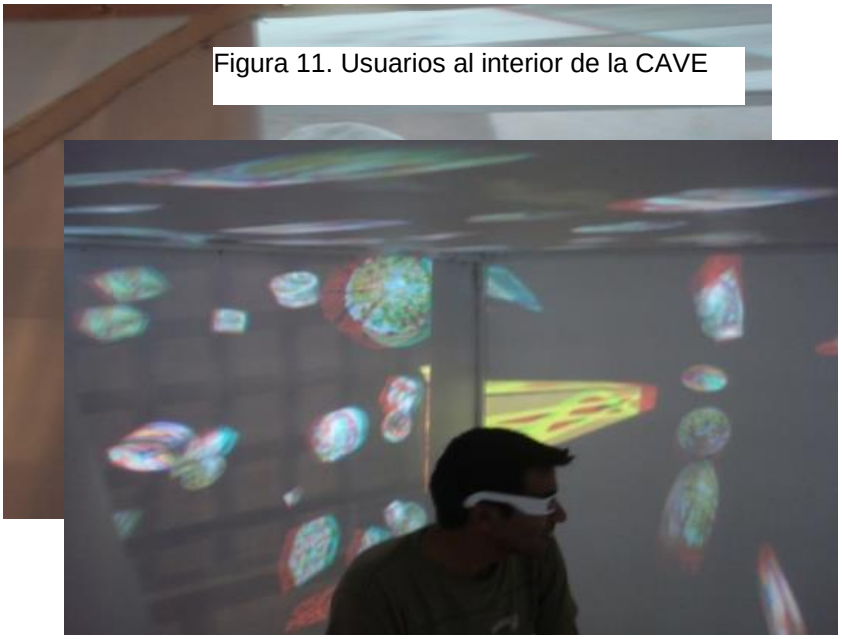

Figura 11. Usuarios al interior de la CAVE

Figura 10: Visualización de moléculas en la CAVE

Figura 15. El sistema de localización e interacción de la CAVE

Discusión de resultados

Esta sección describe las experiencias obtenidas al implementar un CAVE con elementos de bajo costo, tales como PC, Red, video proyectores, que trae como

resultado una reducción importante de los costos. Con esta infraestructura, el Instituto Tecnológico de Colima dispone de un sistema de visualización de alto desempeño para el desarrollo y ejecución de aplicaciones orientadas a la educación.

CONCLUSIONES

En este documento se ha presentado el desarrollo de tres sistemas destinados a la visualización de gráficas en grandes superficies acompañadas de alta resolución. Lo notable de estos desarrollos consiste en que se han empleado elementos de bajo costo en su implementación. Los resultados obtenidos muestran que son una real alternativa a los costosos sistemas de visualización que emplean tecnologías propietarias y que son inalcanzables para organizaciones con reducido presupuesto.

Las soluciones aquí expuestas pueden ser utilizadas por instituciones educativas en aplicaciones destinadas a la educación, tomando como base su reducido costo y el hecho de que muchos de sus elementos pueden ser reutilizados de los laboratorios de dichas instituciones.

Bibliografia

Chen, Y.; Chen, H.; Clark, D.W.; Liu, Z.; Wallace, G. & Li, K (2001). "Software environments for cluster-based display systems," Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on , vol., no., pp. 202,210.

Cruz, Neira C; Sandin Daniel, J. & Defanti Thomas, A. (1993). "Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE". Proceedings SIGGRAPH , 135-142. Intel Corporetion (s.f.) "OpenCV". Recuperado el 2 de Febrero de 2009, de http://opencv.org/

Jérémie, Allard; Valérie, Gouranton; Loick, Lecointre; Sébastien, Liimet; Emmanuel, Melin; Bruno, Raffin & Sophie, Robert (2004). "FlowVR: A Middleware for Large Scale Virtual Reality Applications". Euro-Par 2004 Parallel Processing, Lecture Notes in Computer Science, Volume 3149, 2004, pp 497-505.

Jérémie, Allard; Clément, Menier; Bruno, Raffin; Edmond, Boyer and François, Faure. (2007). "Grimage: markerless 3D interactions". In *ACM SIGGRAPH 2007 emerging technologies* (SIGGRAPH '07).

Li K.; Chen H.; Chen Y.; Clark D.W.; Cook P.; Damianakis S.; Essl G.; Finkelstein A.; Funkhouser T.; Housel T.; Klein A.; Liu Z.; Praun E.; Singh J.P.; Shedd B.; Pal J.; Tzanetakis G. & Zheng J. (2000). "Building and using a scalable display wall system," Computer Graphics and Applications, IEEE , vol.20, no.4, pp.29,37, Jul/Aug 2000.

Mark Segal and Kurt Akeley (1994). "The Design of the OpenGL graphics interface", Silicon Graphics Computer System.

Robert, W. Scheifler and Jim Gettys (1986). "The X window system". *ACM Trans. Graph.* 5, 2 (April 1986), 79-109.

Schaeffer, B.; Goudeseune C. (2003), "Syzygy: native PC cluster VR", Virtual Reality, 2003. Proceedings. IEEE , vol., no., pp.15,22, 22-26 March 2003.

Wolfgang, Krüger; Christian-A, Bohn; Bernd, Fröhlich; Heinrich, Schüth; Wolfgang, Strauss, and Gerold, Wesche (1995). "The Responsive Workbench: A Virtual Work Environment". *Computer* 28, 7 (July 1995), pp. 42-48.