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

Introduction

The monitor is the device normally used to display graphical information to the user of a computer. However, from the point of view of its physical characteristics, the monitors have evolved slowly. Currently the LCD monitors replace CRT monitors, but this happens only some years ago. In contrast, most of the components of the PC, such as memory, processor and disk have seen their capabilities practically doubled every year, while resolution monitors has increased only at a rate of 5% per year over the past two decades (Chen Y., Chen H., Clark, D.W., Liu, Z., Wallace, G. & Li, K ,2001)

The resolution and dimensions of the monitor are an obstacle for certain types of applications that have as main feature the generation of significant amounts of data. Scientific data visualization, virtual and augmented reality environments, engineering design and the analysis of graphs in finance, are examples of applications that require surface of graphical visualization of larger than those provided by the monitor. Running on a system with reduced graphics power, an application can not exploit full capacity on the generation of the graphical environment. As a result, the visualization of complex graphics and the process of interaction with the user are strongly perturbed. For example, a user to interact with an application that displays maps, is forced to visualize fractions of the overall image if you want to scan the image details, this involves the loss of the vision of the global context, situation further compounded if multiple users are involved.

The interest in increasing the size and resolution of the display surface has been in recent years. Current solutions use a centralized or a distributed architecture. The centralized approach is to provide a single node of multiple graphics cards and thus feeding more than one monitor. As a bonus, we have a low-cost and easy to implement; However, this scheme suffers from some drawbacks. One node performs the graphic process (rendering), which means low efficiency. A major problem is the limited number of available PCI and AGP ports on the node. A more attractive solution is to divide the graphics processing among a group of nodes interconnected by a data communication network and distribute the images and graphics primitives using an efficient protocol. In this model the main problems are excessive consumption of bandwidth of the network and synchronization of the activities of the nodes. Despite these drawbacks, developing projects currently employed as graphic solution distribution process.

Graphic display systems based video projectors such as:. Display Wall (Li K., H. Chen, Y. Chen, DW Clark, P. Cook, 2000), Workbench (Wolfgang Krüger, Christian-A Bohn, Bernd Fröhlich Heinrich Schüth & Wolfgang Strauss, 1995) and CAVE (C. Neira Cruz, Daniel J. Sandin & Thomas A. Defanti 1993), have been widely used in data visualization.

These systems allow applications display pictures in high resolution and large accompanied by stereo vision to increase the degree of immersion of users. The following sections provide an analysis of these three solutions using this approach becomes.

Wall Display

A Wall Display is a system capable of displaying high-resolution images and large (Li K. et al. 2000). The overall image is generated on one or more surfaces of a set of video projectors, which are fed by graphic type outputs a supercomputer SGI Onyx. In Figure 1, a Display Wall consists of eight independent video surfaces illuminated by spotlights shown. Each projector is responsible for generating a fraction of the projected global image. A Wall Display can display 2D images on large surfaces, which facilitates detailed, for example, in medicine, satellite imagery, architecture, etc. analysis.

Figura 1. Display Wall generated by an arrangement of eight video projectors

Responsive Workbench

The Responsive Workbench (W. Krueger et al., 1995), is a tool to generate 3D stereoscopic images that are projected on two surfaces perpendicular view with the help of a system consisting of a projector and mirrors. The user can interact with applications developed using gloves and a

3D mouse. Furthermore, a monitoring system monitors the position of the user's head so that this display the virtual environment from the right angle.

The workbench is a highly appreciated working with electronic models for the closeness you have with the user's hands tool. Figure 2 shows a user using the Workbench.

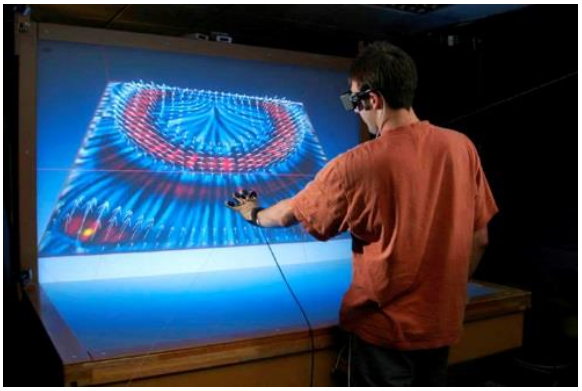

Figura 2. Interaction with a 3D application running in the Workbench

CAVE

The CAVE system is a 3D high-performance graphics display consists of a six-sided cube. The faces are illuminated from the outside by video projectors fed by channels graphics supercomputer, in which applications they produce the graphic displayed on the surfaces of the CAVE run. Users located within the CAVE provided 3D glasses, they are practically surrounded by stereo images accompanied contributing to increase the degree of immersion. The applications focus on CAVE display scale models of physical phenomena.

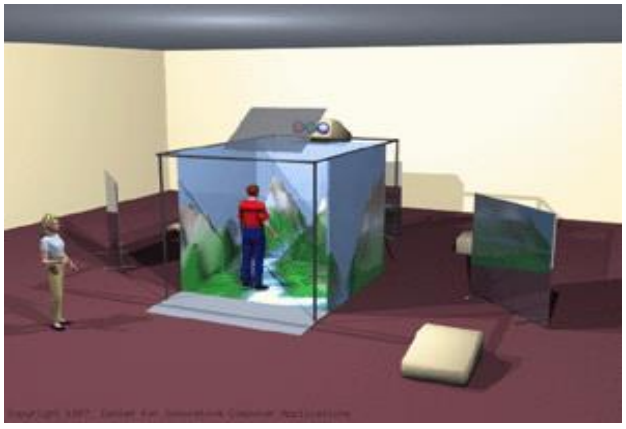

Figura 3. Estructura de una CAVE

Low cost solutions

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

Currently there are several platforms that follow the distributed components integrated with low-cost, one of the most outstanding is Grimage (Jérémie Allard, Clément Menier, Bruno Raffin, Edmond Boyer & François Faure, 2007) display approach, which consists of four elements : a cluster of graphics processing, a system imaging, a library for developing parallel applications and a display surface large. On this platform, you can purchase an object to be instantly entered

and 3D modeling in a virtual world with which you can interact. Figure 2 shows images of Grimage.

Figura 2. Operating Platform Grimage

In Figure 3 is shown in detail a diagram of the operation of Grimage. First, video streams from a set of cameras are captured, the required object does not arise and every element of background is subtracted, thus a 3D model of the captured object is created and rendered in a display surface large.

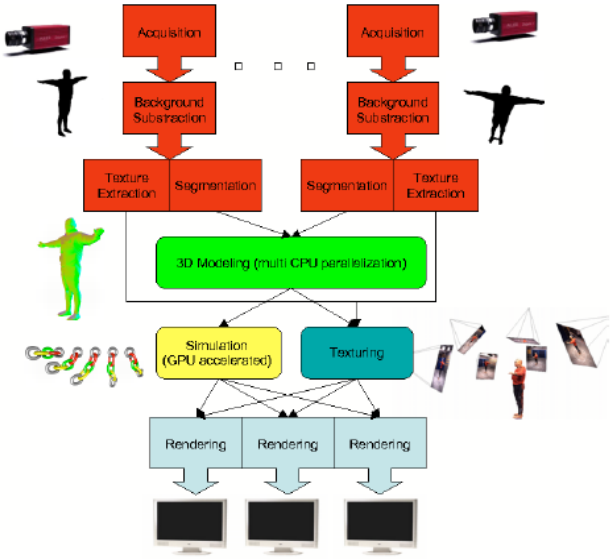

Figura 3. Operating Scheme Grimage

Grimage required to operate a team of high performance, the existing configuration at INRIA Rhone Alpes consists of 6 high definition cameras, a wall of images from 16 video projectors and 2 clusters interconnected, one with 11 dual-Xeon 2.6 GHz for application and another with 16 dual-Opteron 2 GHz for the display part. For the distribution of processes and messaging between these, Grimage uses a software library called FlowVR, which is described in the next section.

FlowVR

FlowVR (Jérémie Allard, Valérie Gouranton Sebastien Liimet, Emmanuel Melin, Bruno Raffin & Sophie Robert, 2004) is a library that provides users with the tools to develop and execute interactive applications in high performance computing Grids s Cluster tools. The main applications of this library are those solutions of virtual reality and scientific visualization. FlowVR facilitates modular programming that reduces software engineering difficulties while allowing executions of high-performance applications in parallel and distributed environments. Architecture and applications are shown in FlowVR and 7.

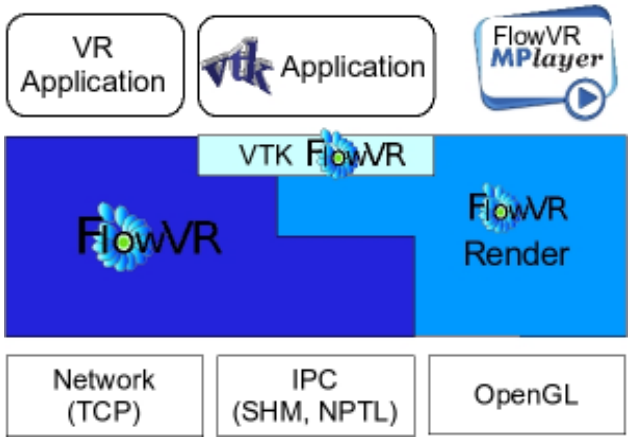

Figura 4: Arquitectura de FlowVR

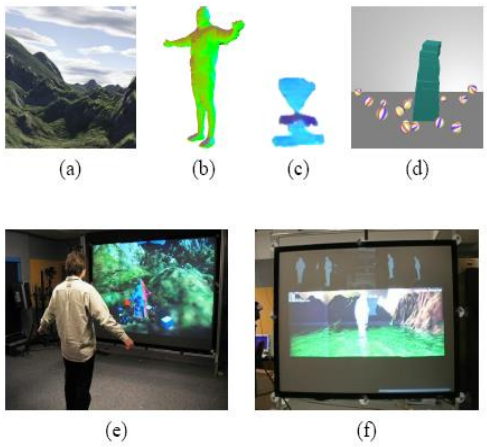

Figura 5. Diferentes aplicaciones de FlowVR

In Figure 5 different applications FlowVR shown:

(A) rendering of land, (b) modeling of a camera body, (c) distortion of objects, (d) simulation of rigid bodies, they can all be combined into an application running on a cluster.

DISPLAY SYSTEMS IMPLEMENTATION OF LOW COST

In this section the construction of three prototypes using components for low-cost operation is described. We define the low cost, as the use of open architecture components such as PCs, video projectors, computer cluster, also called commodity components, as an alternative to expensive proprietary infrastructure such as supercomputers.

Prototype 1: Wall low cost image

In this section we describe the work done to implement a wall of images using inexpensive components. A wall of images is a system capable of displaying high-resolution images on large surfaces. The overall image is generated on one or more surfaces of a set of video projectors, which are fed by graphic outputs of the nodes constituting a cluster. In Figure 8 a wall of images composed of four independent surfaces (named tiled) shown. Each cluster node is responsible for generating for each tiled image and data transmission network moves graphics primitives and synchronization commands between them. In the cluster there are one or more nodes that allow user interaction with applications, either by using the keyboard and mouse or by other means such as gesture recognition, tracking the user's position and so on.

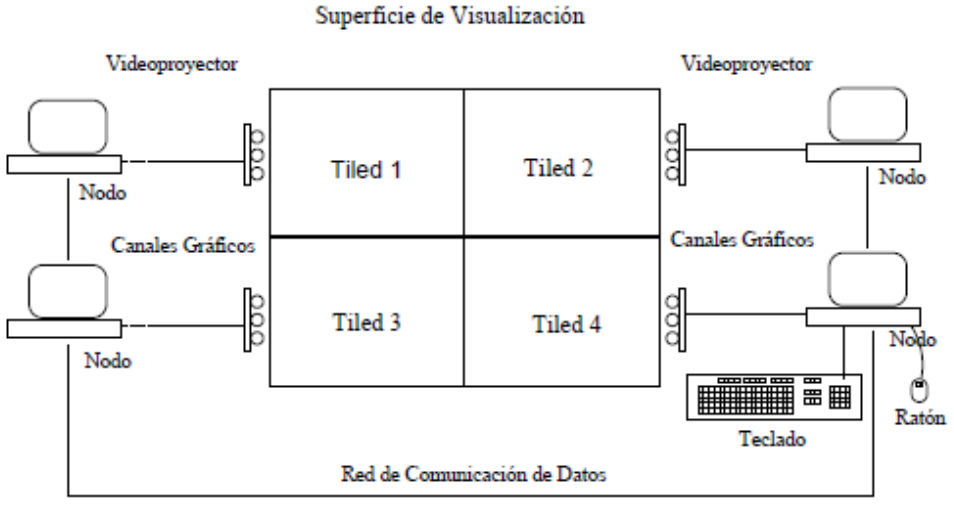

Figura 6. Building a wall of low cost image

The proxy solution

In this section we describe the work done to spread the display of the X Window Server (Robert W. Scheifler & Jim Gettys, 1986) and view graphs of X Window (X clients) application on multiple nodes in a cluster. The messages produced by the X customers need to be modified and distributed among the group of nodes. Similarly, return messages, product user interaction should be recovered and led to customer X. The physical architecture of the system (see Figure 9) is constituted by:

Master node: Host a running instance of the proxy. This node must imperatively connect customers, using the socket in which the proxy listens.

Rendering nodes: Running an instance of the X Window server and have the mission to receive the graphic requests sent by the client through the proxy, perform the rendering and later viewing. Each node performs the rendering of the fraction of the image that corresponds to view. With this procedure, the set of monitors or projectors, arranged under a special agreement have one display surface on which images are deployed clients.

Interaction node: Enables data input devices, keyboard and mouse, by which the user can interact with applications. This node also performs the graphic display.

A. **Virtual Display**

The first task of our prototype is made at the time of its release, it is to define the configuration of the virtual display. The user provides command line in the list of nodes and the provision rendering stored in the virtual display. The proxy sends graphics primitives opening display (XOpenDisplay) rendering each server in order to recover their graphic features, particularly the dimensions of the display. With this information, the proxy constructs and determines the resolution of the virtual display, for example: 4 rendering servers A, B, C and D whose displays are available 1,280x960 pixels and forms a 2x2 arrangement, will form a virtual display 2,560x1,920 pixels, as shown in Figure 9. At the time an X client connects, the proxy sends the characteristics of the virtual display. With this information, the X client believes to have a display area larger than that provided by the monitor of the PC on which it is running.

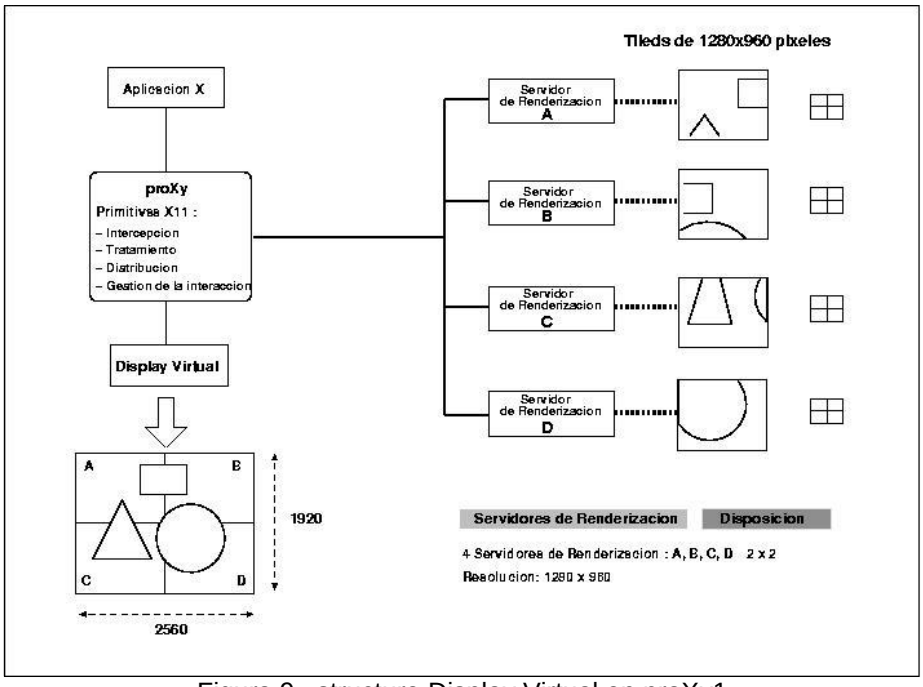

Figura 9. structure Display Virtual en proXy1

B. Rendering the flow

We developed two strategies for the treatment of graphics primitives. The first strategy is applied by the prototype called Proxy1 and involves the interception and coordinate adaptation carrying graphic primitives in order to fit the context of the virtual display. This solution is expressed by the following expression:

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

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

Where (1) X, Y identifies the virtual display resolution (u, v) position rendering server numbered from (0,0), (Xvirtual, Yvirtual) the coordinates of a point in the virtual display and (X (u, v), Y (u, v)) the corresponding coordinates with respect to the rendering server (u, v). Thanks to the actions of the clipping function render nodes made only the rendering of the virtual display each administered. 9 shows this procedure.

The (2) second solution is used by the prototype called Proxy2, this proposed broadly decentralize functions adapting to the nodes graphics primitives rendering. To this end, Proxy2 builds a display window on each node rendering. The special feature of this window is that it has a similar dimension to the virtual display. This window is positioned in each node rendering the interval virtual display coordinates of each node is responsible for rendering display, the rest of the window is eliminated by the clipping function.

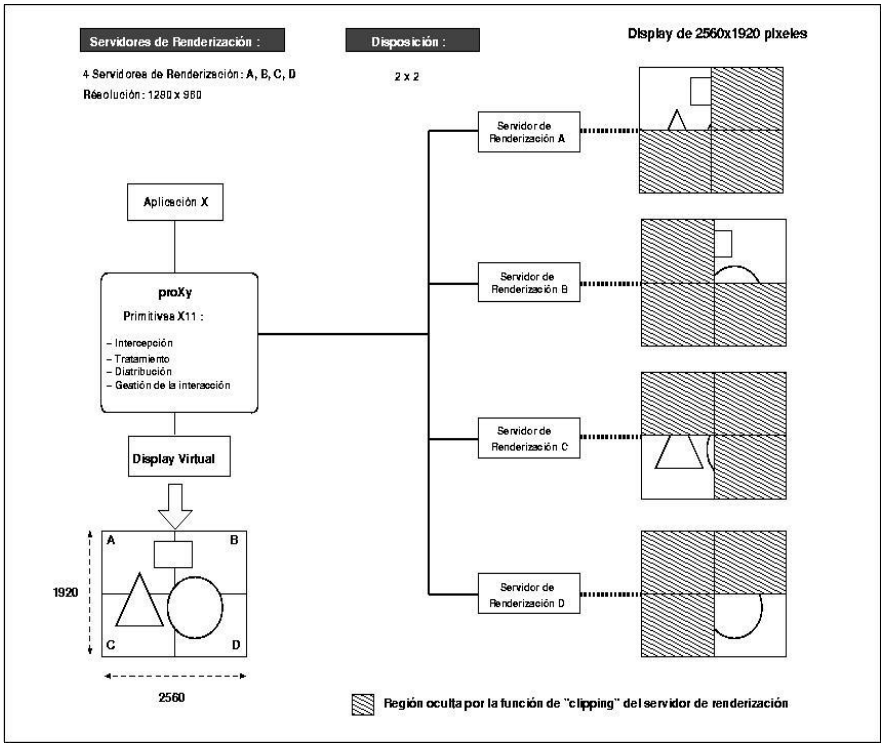

Fig. 10. Construcción del Display Virtual en la solución proXy2

C. Flow interaction

To retrieve the user's actions on the mouse and keyboard messages from node interaction intersect. The X server has the focus mechanism to determine the window to which the results of rendering and interaction should be directed.

While the focus remains confined to the node tiled interaction events keyboard and mouse are treated locally. The situation changes when the user drags the mouse cursor to a neighbor tiled. Then the events generated in the interaction server must be addressed to the node possessing

rendering the focus at that time. We use the extension xtext present in the X Window server which allows you to send events to a remote display. Our solution is based on this code. The results are shown in Figure 11.

Fig. 11 Wall pictures inexpensive Tech Colima

Discussion of Results

In this section we show the development of a proxy server that receives messages from the X Window System clients and distributes them to a set of X servers, which cooperate to build the image of the client, obtaining as a result of larger graphics and higher resolution.

To deploy the display of the X Window server, we used two strategies. The first is to adapt the graphic context of the virtual display primitive, this task is performed on the prototype called Proxy1. The second strategy is implemented in the prototype Proxy2 is to perform the adaptation process of rendering nodes.

With this implementation, the user benefits by having a tool that displays graphs of the X Window System applications over large areas, increasing the amount of information that can be perceived and improving the process of interaction with the user. One advantage of this solution is that it can be implemented with elements present in academia, for example, the Technological Institute of Colima, with this infrastructure will be available to view 2D images in large sizes.

II.- Prototype 2: inexpensive platform for interaction with virtual objects

This section presents our second development, a platform to run augmented reality applications.

Our solution is divided into two functional modules:

A. A module for the acquisition and distribution of the images captured with webcams.

• A module for the treatment rendering video stream and mixed with interactive 3D scenes.

In this first phase, to test the feasibility of our idea, consisting of only two nodes prototype was implemented, node acquisition and rendering node interconnected by a network, as shown in Figure 12.

The acquisition Node

The acquisition node connects one or two Web cameras, which acquire images of the environment. A video server stores and transmits these images to node rendering.

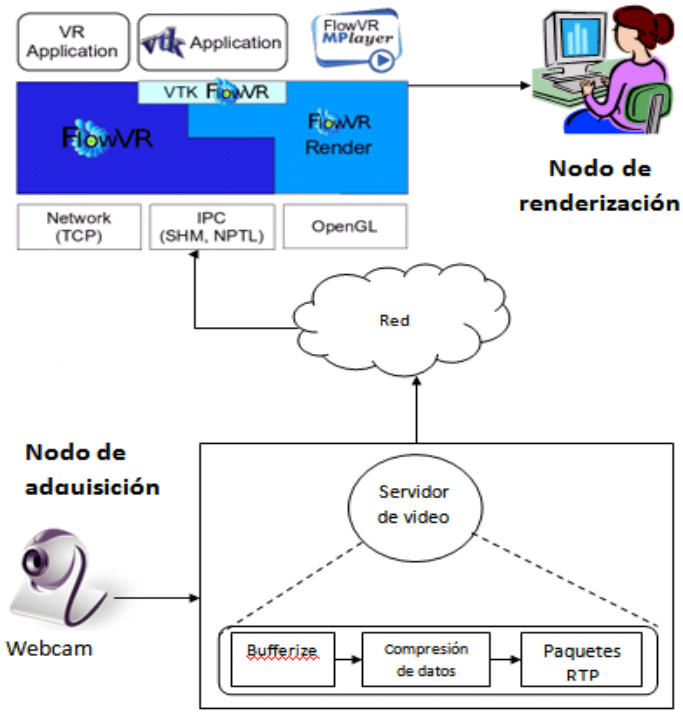

Figura 12. Architecture augmented reality prototype inexpensive

Node rendering

The render node running an instance of FlowVR, which receives video frames, performs the rendering and later viewing through FlowVR immersed in MPlayer.

Performance

The acquisition node captures images and sends the video stream to the rendering node. As video server, we use the software Spook (SPOOK, 2008), a free software that allows sending video streams over IP networks. The Spook software takes images stored in the buffer, the commands sequentially, compresses the data and sends it through the network. Meanwhile, in rendering node, the client initiates the player immersed in FlowVR MPlayer. In this way, you can display the images captured by the webcam at node rendering.

The mixture of the images captured by the webcam and the generation of 3D scenes is provided by the component FlowVR Render, which allows the combination of streaming video with a virtual environment rendering FlowVR.

Application description

To test the functionality of our platform, we have developed some examples where the recognition of contours captured by the webcam and interaction with these objects without using any device interaction objects is shown. For both operations have used the library (Intel Corporetion, nd) OpenCV.

Contour recognition

An outline is a list of points that represent one way or another, the curvature of an image. This representation may be different depending on circumstances. There are many ways of representing. In openCV contours are represented by sequences wherein each entry of said sequence encodes information about the location of the next point on the curve.

For displaying the contours of an image, first a method for memory management is needed. OpenCV has an entity called Memory Storage, which is a method for managing the location of dynamic objects in memory. The type of object that can be stored in a memory storage is a sequence. The sequences are linked lists of other data structures.

The first example is the recognition developed the outline of a series of letters and freehand drawings. The result is shown in the following images.

.

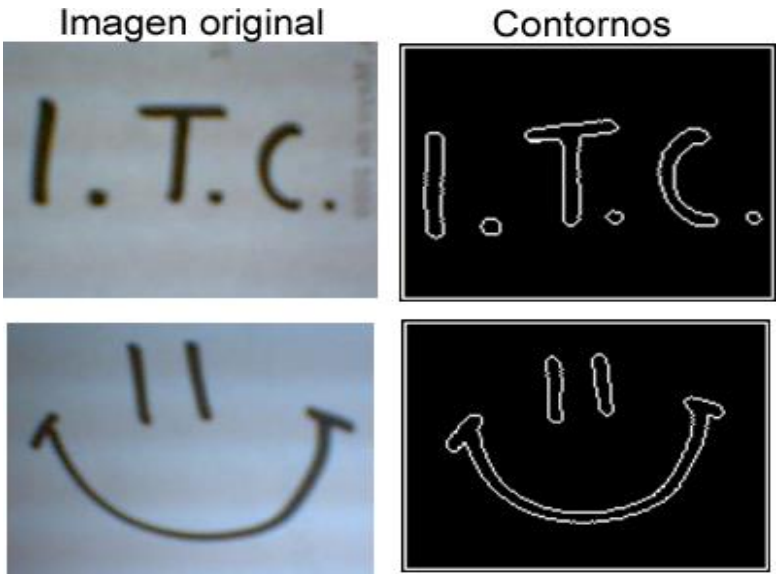

Figura 12. Edge Detection

Interaction with virtual objects

Using this prototype, you can capture objects using webcam and embed them in a 3D application, which has the potential to interact with virtual objects that make up the 3D scene. In an application of this type we identified three elements shown in Figure 7.

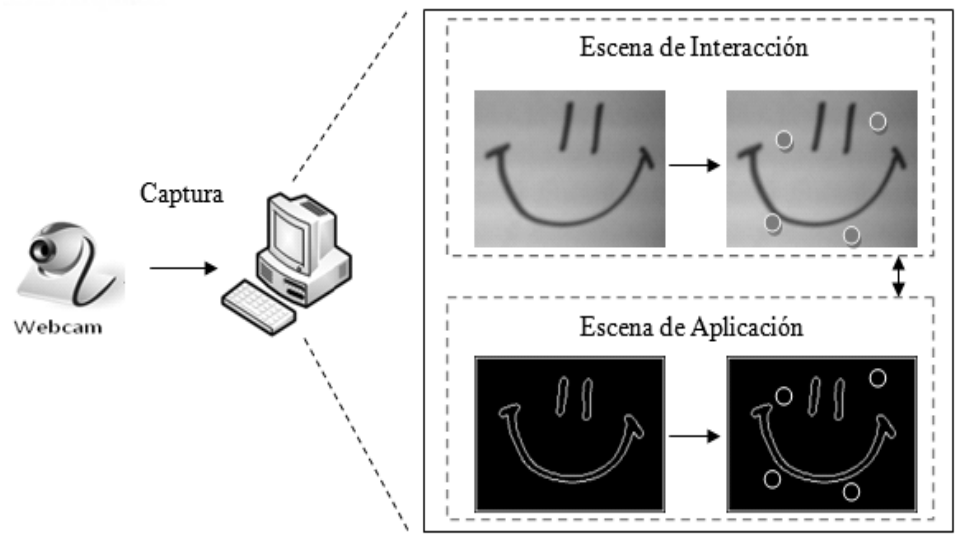

Figura 7. Architecture of the developed applications

A. The scene of interaction is represented by a set of images captured by the camera.

B. The scene of application is represented by the set of images that are some features of the scene of interaction. For example: outline, corners and colors.

C. Finally, the virtual objects: they are embedded in the scenes of interaction and application programmatically objects that react to events in the scene of application and are represented through images or textures in the scene of interaction.

To check the interaction with virtual objects, he developed a couple of examples intended to show the way in which a user can interact with them. The first application shows a set of bubbles that the user can exploit to make moves at the scene of interaction. The scene of the application is the difference between an image and its predecessor. Thus, a white shadow is formed.

The second sample application shows a set of balls that bounce in the scene of interaction, when the user touches any or when collide, changing its direction.

The application consists scene contours of the figures that appear in the scene of interaction, which on contact with virtual objects, they perform a routine to change its direction of motion. An example of these two applications can be observed in Figure 8.

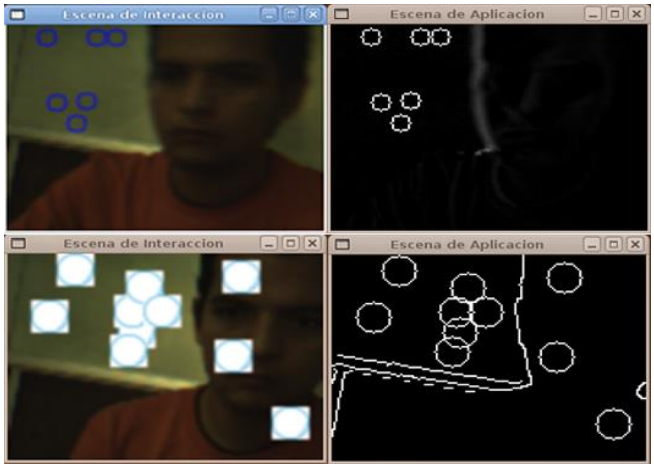

Figura 8. Application that combines images acquired and virtual scenes

III.- Prototype 3: Implementation of a low-cost CAVE

In this section we describe the development of a prototype low-cost CAVE. To achieve the goal of implementing a CAVE, we divide the work into four subsystems are described in detail in the following sections.

A. Display area

The display area is composed of a five-sided cube. We build a metal cube-shaped structure of 1.80 m per side, then we cover the bucket with a special fabric to allow the projection of images from the outside. For the entry of users into the CAVE, an access provided on one of the faces sufficient to allow passage of a user both implemented space. In this display area six users can stand with some comfort, details are shown in Figure 9.

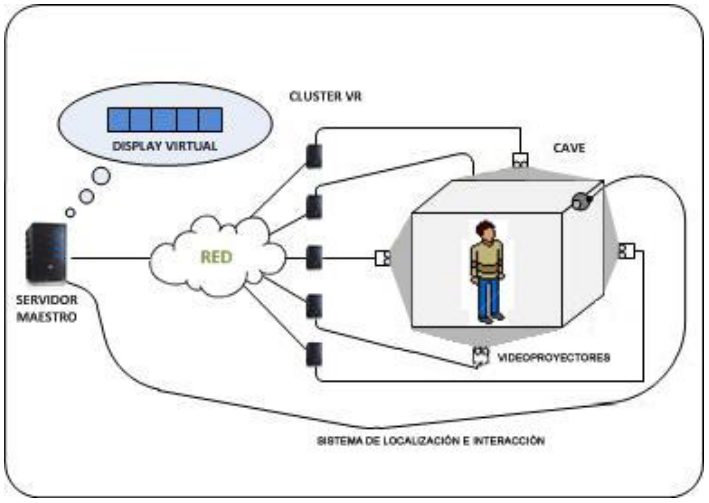

Figura 9: CAVE subsystems inexpensive

A. Clúster VR

Running applications for the CAVE is supported on a cluster consisting of six VR PC equipped with 2 GB of RAM and 1.8 GHz processor. The PC graphics cards are generic and do not provide support for stereo viewing. PCs are interconnected by a Fast Ethernet network at 100 Mbps. In

order to reduce the delays associated cost of communication in the cluster VR, we opted for the distributed execution model synchronized copies. In this model, each PC Cluster VR maintains a complete copy of the application running; although it has the disadvantage of excessive consumption of memory in each node, you gain in terms of reducing the cost of communication. You only need to exchange synchronization messages in the copies hosted on the cluster nodes VR implementation, these messages usually are of small size which causes small latencies. To support applications in the CAVE, we use the software syzygy (Schaeffer, Goudeseune, 2003) that allows parallelizing distribute applications rendering 3D graphics scenes.

C. Projection System

It consists of a set of five video projectors conveniently located outside the display surface, at a distance and distributed in an array such that completely illuminate each of the sides of the CAVE. Each projector is connected to the outputs of each cluster node VR. We use video projectors typically used in classrooms to support the delivery of courses.

The resolution is 1,024x768 pixels handled. Importantly, these video projectors do not have the stereo capability.

D. Location System and interaction

To enable user interaction with applications running on the CAVE, it was necessary to implement an alternative to traditional tracking systems. Our solution determines the position of a user using the images acquired by a webcam interconnected to the VR master server cluster. A user equipped with a helmet in which a high-intensity LED is subject pilots interacting with the application running at the time. The internal area of the CAVE acquired by the webcam represents the total area in which users move. Any change in the user's position is detected by the webcam and this means regenerating the 3D scene and later viewing.

To detect changes in the images acquired by the camera, we implement a small program with the help of the OpenCV library. In this maner, the moving into the CAVE users can navigate 3D applications. In this first prototype only support for a single user interaction was implemented.

Prototype function tests

To measure the functionality of our CAVE, use some OpenGL applications (Mark Segal & Kurt Akeley, 1994) library. Subsequently, we launched the cluster running on the VR and video projectors lit to illuminate the CAVE. The results were surprising. Despite not use devices with support for stereo, the realism of the images inside the CAVE and dip were particularly encouraging. Running applications on the cluster VR had a quite acceptable performance. Regarding the management of the user's location and interaction with applications, the solution implemented properly served. Images of running applications CAVE shown in Figures 14, 15 and 16.

Figura 11. Users within the CAVE

Figura 10: Molecules in the display CAVE

Figura 15. The location system and interaction CAVE

Discussion of Results

This section describes the experiences gained by implementing a CAVE with inexpensive items, such as PC, Internet, video projectors, which results in a significant reduction in costs. With this infrastructure, the Technological Institute of Colima has a display system for high performance development and implementation of education-oriented applications.

CONCLUSIONS

This paper has presented the development of three systems for the graphic visualization of large areas accompanied by high resolution. The remarkable thing is that these developments have been used inexpensive items in their implementation. The results show that they are a real alternative to expensive visualization systems that use proprietary technologies and are unreachable for organizations on a budget.

The solutions presented here may be used by educational institutions in applications for education, based on its low cost and the fact that many of its elements can be reused in the laboratories of these institutions.

Bibliography

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.