ISSN 2320-9984

Volume 1, Issue 2, July 2013 ISSN: 2320-9984 (Online)

International Journal of Modern Engineering & Management Research

Website: www.ijmemr.org

Cloud Rendering System

Yogesh Khullar
Department of Computer Science & Engineering,
Gyan Ganga College of Technology,
Jabalpur (M.P.). [INDIA]
Email: sjkhullar@gmail.com

Dr. Mukta Bhatele
Professor & Head of the Department
Department of Computer Science & Engineering,
Jai Narain College of Technology,
Bhopal (M.P.). [INDIA]
Email: mukta_bhatele@rediffmail.com

Abstract*—*Cloud Rendering system depicts rendering of multimedia contents like dicom, virtual reality, augmented reality, high resolution images. Cloud rendering system provides the cost effective, flexible, scalable, extensible solution for high resolution multimedia contents. There are varieties of software available in the market. The Chromium is one such software which elegantly renders multimedia contents to a large format display, by partitioning and sending individual OpenGL primitives to each client per frame. A system containing large number of rendering nodes, it would not be good idea to transfer large number of data over a network, as it will badly impact the scalability and performance. We will demonstrate the rendering system that allows an application to render a large tiled display. The method integrates improved graphics API forwarding, scene change rendering, content based rendering methods for high resolution multimedia contents etc for displays.

Keywords: *Cloud Rendering, Scalable rendering, High rendering system, Tiled Displays, Remote Graphics*

1. INTRODUCTION

The effective rendering of multimedia content depends upon the size of content and display area. The high resolution of contents are important for visualization. The display area grow drastically over a period of time, as a result the public displays will require wider viewing capability. The considerable effort will be required to construct, maintain hardware system.

The display resolution offered by today's graphics hardware is typically not sufficient to visualize these large datasets. Large format displays such as the PowerWall [15] and CAVE [16] support resolutions well beyond the common desktop but are limited in their scalability. Previous attempts to solve the resolution challenge have faced a number of limitations. Many visualization applications however required large display areas and high display resolutions simultaneously to provide both detail and context. Virtual Reality environments, scientific visualization, etc. are examples where large display size with high resolutions can be useful. Such displays can be built using specialized hardware or using a cluster of computers.

With the advancement of technology, the replica of monolithic display can be achieved by clubbing large number of displays providing the feel of monolithic display system. The technological changes in the area of distributed computing/rendering, networking, display hardware and the emergence of integrated computing display products aided to cluster rendering research.

Cloud based rendering is the concept; many researches have been carried out to provide the effective, scalable and load balancing system. The scene graph

applications used the static primitive information, which can be passed on to the rendering node before displaying. But unfortunately it could not support dynamic changes in the scene graph application. The dynamic changes in the rendering content is the demand of many multimedia application.

The evaluation of rendering system is evolved over a year and many research has been done in the rendering area. There is tradeoff between resolution and dimension of display. The computing power and networking advancement contributed to high resolution displays. These technological trends provides the foundation for well recognized applications solutions available in the market like Hyperwall, NASA etc. Besides, application like Garuda depicts a cluster based tiled display wall for interactive graphics applications.

Garuda's application is built using open scene graph application program interface. The application built using open scene graph able to render transparently on the tiled display without any modification. The Gruda implements the cache manager and distribute the geometry primitive at the rendering clients to exploit temporal and special coherence in the scene. The software system Garuda provide the scalable clustering environment to cater large number of display without impacting the performance of system. Besides, there are software available in the market which Predominently depends upon the network bandwidth. Chromium generates the texture on the server node and distribute to the rendering nodes connected in the LAN. The geometry information is also passed on to the rendering node. Wallace describes the design issues behind the Princeton display wall from the point of view of scalability to videos, images, graphics and audio.

Large number of applications can be supported using the proposed solution. The cloud server will responsible for controlling the display connected across the network. Cloud server will provide the configuration primitives and will pass the multimedia information to the LAN server tiled displays. LAN server will responsible for generating and passing the primitive information to the display node.

The LAN server can able to support the multimedia contents ranging from high resolution images to high dimensional multimedia contents. The framework can support wide range of applications like virtual reality, augmented reality without impacting the scalability of a system. The applications commonly associated with immersive, highly visual, 3D environments etc. The interactivity can be provided between the LAN server and the rendering node. The user can interact virtual reality contents at LAN server and corresponding changes will get reflected at rendering node of display wall.

2. BACKGROUND AND RELATED WORK

We will depict the background of research carried out in the area for the rendering of multimedia contents to provide the virtual reality, augmented reality, Medical imaging, rendering of high resolution images, video etc.

The technology advancements in the field of networking offers high speed data transfer amounting to mega byte or gigabyte. The consistent efforts have been made in the past to provide the software systems to operate efficiently over a network. We can visualize the technological trends in the area of graphics, cloud rendering, cluster rendering which created the inroads for cloud rendering system.

Xinerama

Xinerama[1] software which can be used to club two or more physical devices and provide the simulation environment of one single display unit. The Xinerama software system uses the extension to the X Window System which enables X applications and window managers to achieve the desired results.

Figure 1: Xinerama displaying the text spanning across all the physical devices.

Xinerama have some inherent drawbacks, Xinerama requires that all of the display should have the same bit depth, it does not support the hetrogenous system. The physical displays do not need to be the same resolution, and the virtual display area is not necessarily rectangular if the component physical displays are not the same size.

As the size of display wall increases it performance on the system degrades gradually. This method is not suitable for large clusters of systems as Xinerama makes the rendering really slow in such situations

Parallel Rendering

Cluster rendering systems have a growing interest in the recent past. Tiled display system have exposed the parallel rendering system and delivered the robust software system. The application built using WireGL framework allowed the signal application to a tiled display over a network. WireGL used the sort first parallel rendering methodologies to provide the large display wall with minimum effect upon the performance of a system. The disadvantage of these systems is poor utilization of graphics resources available in the network.

The sub system currently available focus on cost-based model to achieve load balancing among the nodes in the cluster. It redistribute non-overlapping pixel-tiles to drive a tiled display. All of these algorithms required the full replication of the scene database on each node in the cluster, so further work was done to only require partial replication, trading off memory usage for efficiency

As we are aware the applications badly consumes the network resources to achieve the cluster rendering. In the order to reduce the bandwidth usage there exist a VRJuggler solution which distribute the user's input over the network and direct the application to run in different view mode at each node. In VRjuggler the server node distributes configuration information to clients rather than requiring users to maintain per-machine configuration arrangements. Also, it is no longer necessary to identify, which cluster node is hosting a device because all devices have to be hosted by the server. These user input and viewing parameters will be synchronized. These operations will use the little amount of network bandwidth. The applications built will have the intelligence for user's input and deterministic algorithms to achieve the consistency across the nodes. This requires each node has the power to run the same full application and needs parallel accesses to common resources which can lead to performance bottlenecks. This approach works efficiently only for specific applications that use VRJuggler library in their code

3. SYSTEM ARCHITECTURE

The cloud rendering system provides the Scalability and elasticity to render the multimedia contents over a large display. Large number of display can be supported without impacting the performance of a system, Performance is monitored, consistent and loosely coupled architectures is constructed using web services as the system interface. The application supported cloud rendering systems are easily maintainable, because they do not need to be installed on each user's computer and can be accessed from different places. The cloud server will be providing the configuration console and contents repository. The application deployed at cloud server will inherit the cloud computing capabilities like Agility, Geographic Distribution, Virtualization, Massive, Scale, Resilient Computing, Multitenancy. The cloud server was proposed to address the various tiled rendering challenges like Seamless Display, Easy configurability & use, Extensibility for High

resolution imaging applications, Balanced Distributed rendering, Network bandwidth utilization, Embedded display Disk space, Embedded display rendering capability, Providing Interactivity to the Display Wall etc.

Figure 2 demonstrate the cloud rendering system, the cloud server will provide the console, configuration, repository etc of multimedia contents. Cloud server will able to control the rendering node via LAN server. The multimedia contents like virtual reality, augmented reality, medical imaging, etc will be transferred to corresponding LAN server. The content transferred will happens only once until contents are modified at server. The LAN server will control the rendering node and will responsible for providing the required content information at the display node. The screen change information will be triggered from cloud server and corresponding changes will be getting reflected at the rendering node with the help of LAN server. Each LAN server will keep the cloud server updated regarding the health status of its rendering node.

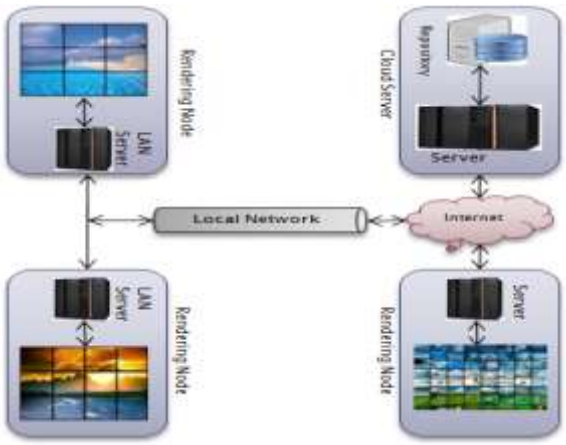


Figure 2: Architecture diagram of cloud server. The rendering node containing LAN server and tiled displays (3*3 display wall) connected in a LAN. Cloud server contains the server node and its corresponding repository.

Chromium is a framework for distributed display rendering. Chromium is generic and extensible mechanism for manipulating streams of OpenGL API commands. There are lots of applications which are built using the framework and achieved the cluster rendering.

Chromium implements sort-first, sort-last and hybrid algorithms to distribute the rendering of polygonal geometry across the cluster. Chromium replaces the existing OpenGL library and fakes the calls coming to it and passes it to the rendering node OpenGL. Chromium directly operates on the stream of the OpenGL graphics commands issued by the application. Each Stream Processing Units (SPU) has its input-streams of graphics commands, performs operation on these commands and passes them on. The SPUs for tiled rendering are tilesort, render and pack. Tilesort SPU implements the sort-first algorithm. The tilesort SPU sorts the OpenGL commands into tiles so that they are packed and sent over a network to the nodes handling the tiles. Render SPU passes the stream to the node's local OpenGL implementation. Pack SPU packs the stream into a buffer for transmission to cluster servers. The frame buffer is subdivided into rectangular tiles which may be rendered in parallel by the hosts of a rendering cluster.

The disadvantage of chromium is the high network bandwidth usage for transfer of texture information. The compressed image is converted in the server side and the texture is distributed to all the rendering node. As the size of multimedia contents increases the raw data sent over the network increases. If there is repetitive display of multimedia contents then the same raw data will be sent which results in the redundant usage of resources.

In our subsystem, we overcome this issue where we pass the encoded information over the network. Hence reduce the usage of network bandwidth and will generate the required information at the rendering node. All the operation related to rendering of multimedia contents will take place at the rendering node. The geometry information of the tile will be passed from the server. The rendering node will cull the required part of information and render the same. All OpenGL operations will supposed to happen on the encoded image file at the server side now will happen the client side. The Application faker

(crfaker.dll) will forward all the calls to the corresponding library at the rendering node. OpenGL call forwarding mechanism enables all the operation to be performed at the encoded multimedia contents.

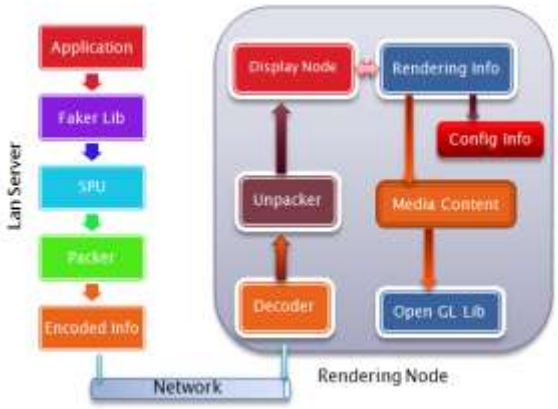


Figure 3: Architecture diagram demonstrating the communication between the LAN server and rendering node. Rendering info module will be responsible for extracting the primitive information and render the corresponding date of tile in cluster.

Algorithm for video rendering:

- Server send the video file to rendering node via TCP or UDP.
- Application faker library forwards all the GL commands from server and at the rendering node OpenGL system library generates the texture information from the video file present at its node.
- View Frustum Cull and forward the coordinates primitive depending on the view of the particular tile.
- Control the texture by changing its view port and change using the OpenGL primitive.

High Definition video rendering

The full HD videos are one of the complex multimedia contents to be rendered on the tiled display. The rendering nodes either download the contents from server or listen to particular port for stream to render. In the prior scenario server will configure the rendering node and each rendering node will display the contents after being downloaded from the server. The each node will be responsible for cropping, scaling and rendering of corresponding contents. The texture will be generated locally at the rendering node, the configuration primitives will be passed from server. Server node will responsible for taking care of synchronization among the rendering.

The current method does not consume much of the network resources, because the display contents will be transferred before the start rendering and subsequently only primitive instruction are passed. In certain scenarios where there is deficiency of space at the rendering node, this method may not prove fruitful.

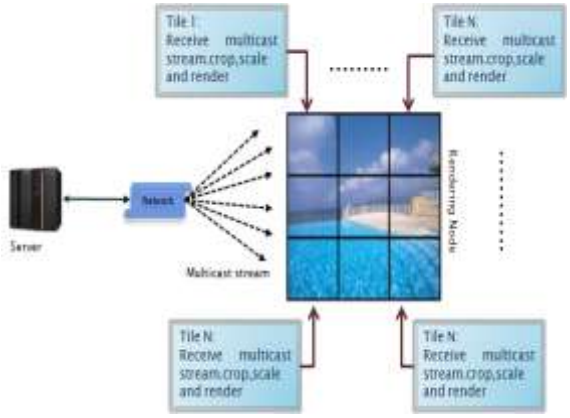
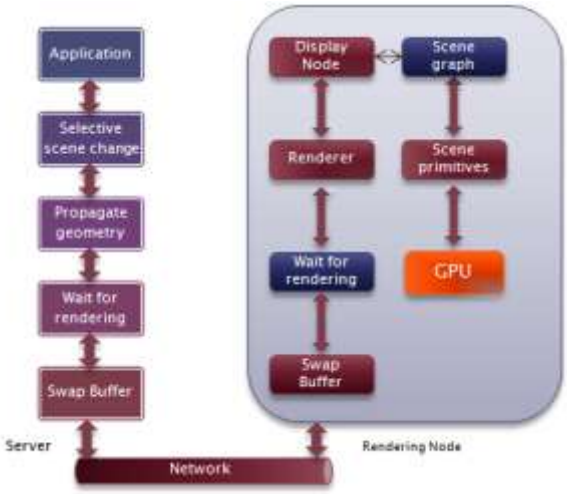


Figure 4: Server node send the multicast stream over a LAN, tiles at the rendering node will be responsible for processing the stream, extract the desired portion of stream and render it on the display.

To overcome this limitation, the server node will send the multicast stream over the network; the stream will be encapsulated under RTP packet. The rendering node will listen to desired stream, crop the stream and render it on the display node. Each rendering node will render the stream based upon the display time stamp of each frame, so as to achieve the synchronization among the tiles. The real time rendering of video contents is achieved at the cost of network bandwidth. However, it overcomes the space deficiency issue that may occur at the rendering node.

Scene based Application Tiled rendering

The OpenSG approach is derived from the VRML based files. In this case the OpenSG library has been modified to provide seamless display for multi window.



The OpenSG application encapsulates the windows on which it renders, moreover each window comprises of viewport (the rectangular areas inside window). Applications can have an arbitrary number of Windows, each of which can have one or more viewports. The OpenSG library needs to be modified to support the rendering system. All the scenarios of multi tile rendering needs to considered. In the OpenSG system application will be seamlessly tiled even if it is written for single file. OpenSG application uses fake library, the server and rendering node should able to support the tile rendering. The rendering node will be notified, selected area of scene got changed, hence needs the rendering (redrawing).

4. CONCLUSION

We have discuss the various multimedia contents which can be rendered in the cloud environment. The generic scalable, robust and extensible framework proposed, which can support large number of display node as we decentralized the rendering responsibility. The pros and cons have been discussed and we have proposed the our solution to support the display rendering system. In case of full HD contents we have showed the solution which have miser usage of network bandwidth and at the same time effective rendering of contents. The solution can be applied to various multimedia contents. The Improvements were achieved with respect to rendering time and network bandwidth using the proposed approach.

5. FUTURE WORK

We have proposed the generic framework to supports the large number of displays. It provides the foundation for applications to support the scalable environment. As the technology changes in no time, we may need to supports different multimedia contents to provide the virtual reality, augmented reality, medical imaging etc. The framework can be worked out to be applied in the various areas like entertainment, logistics, advertisement, gaming etc., The proposed framework provides the opportunity to the existing rendering applications to migrate to cloud infrastructure making the full utilization of technology.

REFERENCES :

[1] Xinerama: http://en.wikipedia.org/wiki/Xinerama

[2] VR Juggler: http://www.vrjuggler.org

[3] OpenSG Library : http://www.openSG.org

[4] Chromium Project: http://chromium.sourceforge.net

[5] Distributed Multi Head X : http://dmx.sourceforge.net

[6] Timothy A. Sandstrom, Chris Henze and et. al - The hyperwall - Proceedings of The Coordinated & Multiple Views in Exploratory Visualization IEEE (2003).

[7] Grant Wallace, Otto J. Anshus et. al- Tools and Applications for Large-Scale Display Walls, IEEE Computer Society (July 2005)

[8] Bruno Raffin and Luciano Soares - PC Clusters for Virtual Reality - Cloud Rendering System Proceedings of the IEEE Virtual Reality Conference (2006)

[9] Rajvikram Singh, Byungil Jeong and et.al - TeraVision: A Distributed, Scalable, High Resolution Graphics Streaming System, IEEE Computer Graphics and Applications (2006).

[10] MOLNAR S., COX M., ELLSWORTH D., FUCHS H.: A Sorting Classification of Parallel Rendering. IEEE Computer Graphics and Algorithms, pages 23–32, July 1994.

[11] DUCA N., KIRCHNER P.D., KLOSOWSKI J.T.: Stream Caches: Optimizing Data Flow. In Visualization Clusters. Commodity Cluster Visualization Workshop, IEEE Visualization, 2002.

[12] Buck I., Humphreys G., Hanrahan P.: Tracking graphics state for networked rendering. In Eurographics/ SIGGRAPH Workshop on Graphics Hardware, 2000.

[13] Byungil Jeong, Ratko Jagodic and et.al : Scalable Graphics Architecture for High-Resolution Displays, Proceedings of IEEE Information Visualization Workshop (2005).

[14] Nirnimesh, Pawan Harish and et al. Garuda: A Scalable, Tiled Display Wall Using Commodity PCs - IEEE transactions on visualization and computer graphics (2007).

[15] University of Minnesota. PowerWall.http://www.lcse.umn.edu/ research/powerwall/powerwall.html.

[16] Caroline Cruz-Neira, Daniel J. Sandin, and Tom DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the CAVE. Proceedings of SIGGRAPH 93, pages 135–142, August 1993.