Workload Balancing in Distributed Virtual Reality Environments

Michael Ditze[1], Filipe Pacheco[2], Berta Batista[2], Eduardo Tovar[2], Peter Altenbernd[1]
[1]C-LAB, 33094 Paderborn, Germany, [2]ISEP-IPP, Polytechnic Institute of Porto, Portugal
[1]Michael.Ditze, Peter.Altenbernd@c-lab.de [2]emt,ffp,bbatista@dei.isep.ipp.pt

Abstract

Virtual Reality (VR) has grown to become state-of-the-art technology in many business- and consumer oriented E-Commerce applications. One of the major design challenges of VR environments is the placement of the rendering process. The rendering process converts the abstract description of a scene as contained in an object database to an image. This process is usually done at the client side like in VRML[1] a technology that requires the client's computational power for smooth rendering.

The vision of VR is also strongly connected to the issue of Quality of Service (QoS) as the perceived realism is subject to an interactive frame rate ranging from 10 to 30 frames-per-second (fps), real-time feedback mechanisms and realistic image quality. These requirements overwhelm traditional home computers or even high sophisticated graphical workstations over their limits. Our work therefore introduces an approach for a distributed rendering architecture that gracefully balances the workload between the client and a cluster-based server. We believe that a distributed rendering approach as described in this paper has three major benefits: It reduces the clients workload, it decreases the network traffic and it allows to re-use already rendered scenes.

1. Motivation

Conceivable applications scenarios that strongly benefit from the use of VR include architectural design analysis, distributed learning environments and travel management settings where vacationers can walk through potential hotels in advance. Imagine a walkthrough scenario consisting in the simulation of a well-known commercial street to be used by residential users. Here, the expectation is to produce a high quality simulation environment that highly resembles the original. The 3D nature of the simulation should allow the users to interact with the environment in quasi real-time: change their point of view in the three-dimensional space, zoom in on details and trigger pre-recorded actions by means of hot spots. When interacting with such a virtual environment, realism depends mainly on three factors: realistic images, interactive frame rate (10 to 30 frames per second) and real-time feedback (motions, behavior, etc.).

By itself, the generation of photo-realistic images from a 3D-object database; i.e. the *rendering process* is computationally extremely expensive, and still imposes major research challenges, whereas the complexity of lighting phenomena associated to interactive usage further calls for powerful and predictable computing in order to met the user expected time constraints.

On the other hand, walkthroughs of large information spaces face the task of generating images from a model containing a huge amount of elements.

Given this a complete framework will include the integration of existing and, when required, development of new solutions to several challenging real-time problems, such as:

- real-time distributed client-server networking providing proper guarantees;
- real-time distributed computation of parallelised rendering tasks for clusters of workstations networked via commodity RT LANs (rendering engine at the server side);
- timely scheduling and execution of rendering tasks and media-players running on the client;
- timely scheduling and execution of multiple client requests (front-end server at the server side);
- the adaptation of current workload, including client-server balancing of rendering load.

2. MPEG-4 as a Client-Server connection

Traditional VR systems usually form a Client-Server architecture where the information of objects is on request transferred from the server to the client. The rendering process itself that may result in large computational overhead is then left to the client. An idea how to transform the conservative client-server methodology to form a distributed rendering approach can now be realized by using MPEG-4.

MPEG-4[2] is a common video compression standard originally targeted at video streaming applications used in environments with very restrictive bandwidth at disposal. It was developed by the Motion Pictures Experts Group and finalized as a standard in 1998. In future, MPEG-4 will be used for video streaming applications in UMTS. In contrast to preceding MPEG standards (MPEG-1 and

MPEG-2), MPEG-4 follows an object-oriented approach. Video scenes are decomposed into single arbitrarily shaped objects called audio-visual objects that are separately encoded and transmitted over the network to the client. Examples for these objects range from primitive media objects like audio or still images to complex object representation in 3D environments. Besides fast encoding mechanisms, the major benefits of MPEG-4 are its scalability in terms of gracefully adjustable video quality with regard to network capacity, reusability of video objects across different video scenes and platforms and QoS support for network service providers.

The advantages of MPEG-4 very well serve the idea of distributed rendering in VR environments. In contrast to the traditional solution where computationally expansive rendering is completely done at the client side, the server which has usually more computational power may now partly pre-compute complex rendering scenes and encode the rendered scene as a MPEG-4 audio-visual object which is then transmitted to the client. Alternatively, the rendering could also be adopted to some other client with vacant resources. The advantages of this new approach are obvious: The client is greatly relieved from the computational overhead it has to spent for rendering and the rendered MPEG-4 object qualifies for re-utilization across various clients that wish to display the same rendered scene. Moreover, network traffic is significantly reduced since a MPEG-4 object that is targeted at low bitrates will consume less bandwidth than complex object descriptions which, in traditional VR environments, still must be transmitted to the client.

3. Network Management Unit

In order for the server to determine which scenes are to be pre-rendered, a network management unit is required. The NMU keeps track of the current network workload as well as of the computational workload on all clients that participate in the VR. If the workload of a particular client exceeds the resources at disposal as determined by a Response Time Analysis (RTA)[4], the NMU may decide to re-distribute the rendering process to the server or to an alternative client with free resources at disposal. This necessitates the client to apply a RTA and a scheduling algorithm that takes the specifics of MPEG-4 and rendering into account. A scheduling algorithm and RTA for MPEG-2 streams that may be adopted to suit MPEG-4 is presented in [5].

Moreover, the NMU is also responsible for reserving the required bandwidth on the respective link in order to guarantee that MPEG-4 video objects as well as VR object descriptions can be transferred within given timing constrains. RSVP provides such a set of communication rules that allows channels or paths on the Internet to be reserved for the transmission of video and other high-bandwidth messages [6]. In order to determine weather a particular network link is still capable of transmitting additional data before a pre-determined deadline, a RTA for the network link must be performed. [7] presents such a RTA for RSVP.

4. Peer to Peer Networking

VR environments usually allow for interaction with virtual objects or other virtual persons represented through clients in the same VR environment. Apparently, in case of multiple clients that navigate the same VR environment, information that represent this environment may be redundant and respective objects, e.g. the background of a scene, may already have been rendered by other clients in the VR. Peer to Peer Networking offers a great opportunity to determine clients or servers that have already rendered this scene and stored in a MPEG-4 video object by addressing the NMU. Instead of repeating the rendering process, the client in need requests this particular MPEG-4 object through the network and displays it. Note, in each case at least the NMU knows where to find particular objects as it is responsible for the distributed workload balancing.

5. Cluster-based Rendering

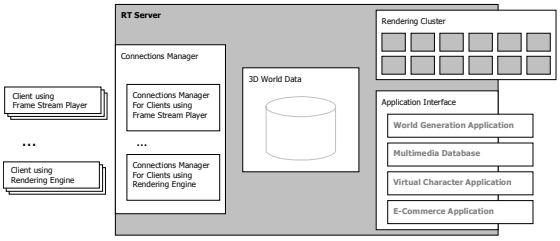

Figure 1 – A VR Server Architecture

For complex scenes or high-quality images, the rendering process is computationally intensive. This is particularly acute for a rendering server, which will have to serve multiple clients. The RT VR Server may consist of a "front-end" machine, for managing the client-server balancing and working as an interface to the NMU, and a cluster of networked personal workstations (PWS) acting as the server's rendering engine. Such a cluster will provide cost-effectiveness for both performance and scalability[8], which are main platform requirements. Additionally, maturity and robustness of Linux/RT-Linux[9] and *de facto* standardization of message-passing via Parallel Virtual Machine (PVM[10]) and Message Passing Interface (MPI[11]) are enabling the design of systems which are entirely made up of COTS technology.

However parallelism problems in rendering are usually regarded as intractable[12]. In fact, although the rendering process contains ample parallelism at different levels of the rendering pipeline, it is not easy to efficiently distribute the processing between different units, mainly due to the enormous sharing of information in the rendering process. The question of integrating parallel renderers into the broader computing environment has often been neglected, and in most cases explicitly ignored[13]. Nonetheless, diverse research works have been published focusing on parallel rendering[12][13][14].

A final important requirement is efficient real-time LAN technologies for the rendering cluster. Even though clusters of PWS are used for parallel rendering in at least one commercial rendering package[15], its actual implementation is hampered by the lack of efficient networking technologies. This will be detailed in the next section.

The Server Architecture may also supports additional client-server functionalities (including application extensions interfaces) that will not be detailed in this paper.

6. The RT Cluster Network

The RT cluster implementation must consider several issues in order to achieve the adequate behaviour/performance, namely the cluster interconnect, the message passing scheme and the operating system.

Traditionally expensive interconnects are proprietary and rely on special purpose hardware losing the cost benefit offered by the commodity market. In this class GigaNet[16] and Myrinet[17] are two of the leading interconnects for clusters of commodity computer systems.

The building block of a Myrinet network is a 16-port switching chip. It can be used to build a 16-port switch, or can be interconnected to build various topologies of varying sizes (albeit not all allow easy finding of contention-free routes). The core of the switching chip is a pipelined crossbar that supports non-blocking cut-through routing of packets. The routing algorithm is based on source-routing according to the information present at the variable-length packet header. Myrinet provides reliable, connection-less message delivery between communication end-points. This is achieved by maintaining reliable connections between each pair of hosts in the network and multiplexing the traffic between end-points over these reliable paths [18]. Simulation results showed, however, that Myrinet latency, under heavy load, suffers due to the blocking in the distributed wormhole routing scheme [19]. There is no efficient support of broadcast communication as well.

GigaNet is a connection-oriented interconnect. No message can be exchanged between communication processes until a VC has been established. Each VC corresponds to the allocation of buffer queues, routing table entries and other resources in the network and, at the host, that limits the size of the cluster. With the connection-oriented communication semantic, circuit-based switching and end-to-end flow control scheme are naturally adopted for GigaNet.

The most popular Local Area Network (LAN) technology is Ethernet. Today standards ensure bandwidths of 10-, 100- (or fast), 1000-Mb/s (or gigabit) and 10000-Mb/s and there are already discussions of 100-gigabit per second Ethernet, which could provide the next generation parallel computers with a smooth upgrade path to their communication subsystem. Ethernet, in addition to bandwidth enhancement present in the last implementations, and particularly in the full-duplex mode, allows switched access at full channel capacity without the limitation of CSMA/CD. Therefore, we believe that, given a scalable switching architecture, Ethernet can be a cost-effective solution for cluster computing.

Conventional Ethernet switches are not fully scalable because they use designs based on a backplane bus or crossbar switch, so cascading is required to build a cluster beyond the size of the upper limit imposed by the number of nodes the switch supports, and latency is increased. The spanning tree algorithm is used to calculate a loop-free tree that has only a single path for each destination, using the redundant paths as stand-by links. At present, due to the remaining lack of switch scalability we believe that applications using e.g. a conventional Gigabit Ethernet switch fabric are limited to the smaller parallel systems in which this application includes to.

The combination of message passing middleware and high-speed interconnect is one of the crucial components for building high-performance commodity clusters. But the performance of capable network technologies can be severely influenced by the overhead in the host cluster computing, as it is also demonstrated in [19]. So there have been proposals where the role of the operating system was much reduced and user applications are given direct access to the network interface, which resulted in the industry standard for user-level communication VIA[20]. Some studies quantified the impact of user-level communication against network bandwidth on the performance of a content-aware server, comparing TCP/IP and VIA over Fast Ethernet and a higher bandwidth network. Results demonstrated that reduced processor overhead, remote memory writes, and zero-copy can all provide performance gains, whereas network bandwidth is not as important [21].

7. Conclusions

This paper presented a work in progress that aims at finding a new approach for the distributed rendering in Virtual Reality environments. Unlike traditional approaches that usually apply a client-server architecture where the computationally expensive rendering process is completely done at the client side, an new idea is introduced that enables the server or some other client to adopt pre-rendering to relieve overloaded clients. It exploits the MPEG-4 standard that allows to decompose video scenes into single individual objects that are encoded and transmitted separately.

Furthermore, a Network Management Unit has been presented that determines vacant computer and network resources and distributes the workload accordingly. It exploits RSVP for bandwidth reservation and RTA as Admission Control and along with a RTA executed at each client, it guarantees QoS all along the data path form the source to sink.

A mechanism for Peer-to-Peer networking was described that allows for efficient re-use of already encoded and stored MPEG-4 objects to VR object representation.

Finally the rendering cluster issues including the internal network were discussed.

8. References

[1] ISO. *Information technology - VRML97. Information technology - Computer graphics and image processing: The Virtual Reality Modeling Language (VRML), Part 1: Functional specification and UTF-8 encoding, 1997, ISO IEC ISO/IEC 14772-1* 1997.

[2] ISO. *Information technology - coding of moving pictures and audio, Overview of the MPEG-4 standard, Final, ISO IEC JTC 1/SC29/WG11 N4668*, March 2002.

[3] Barkai, P.*Peer to Peer Computing*. Intel Press, ISBN 1-55860-475-8,2001.

[4] Burns, A., Wellings, A. *Real-Time Systems and Programming Languages (second edition)*. Addison-Wesley, 1997.

[5] Ditze, M., Altenbernd, P. “A Method for Real-Time Scheduling and Admis- sion Control of MPEG-2 Streams.” *7 th Australasian Conference on Parallel and Real-Time Systems,* Sydney, 2000.

[6] Braden, R. Ed., Zhang, L., Berson, S., Herzog, S., Jamin, S.: “Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification. Request For Comments 2205” http://www.ietf.org/rfc.html.

[7] Schneider,S., Altenbernd, P. “Combining Multimedia Response-Time Analisysis and the Resource Reservation Protocol for Efficient Network Scheduling of Media Streams” *7th Australasian Conference on Paralellel and Real-Time Systems.* Sydney, 2000.

[8] Merkey, P. *Beowulf Introduction*. http://duce.mcs.kent.edu/~farrell/equip/beowulf/intro.html. 1998

[9] Epplin, J. “Linux as an Embedded Operating System” *Embedded Systems Programming. October.* RT Linux: http://www.rtlinux.org. 1997

[10] Geist, A., Beguelin, A., Dongarra, J., Jian, W., Machek, R. and Sunderam, V. *PVM: Parallel Virtual Machine*. MIT Press. 1994

[11] MPI Forum “MPI-2: Extensions to the Message-Passing Interface” *Technical Report MPI 7/18/97*, Message-Passing Interface Forum. 1997

[12] Bartz, D., Schneider, B. and Silva, C. “Rendering and Visualisation in Parallel Environments”. *SIGRAPH 2000, course on Rendering and Visualisation in Parallel Environments.* 2000

[13] Crockett, T. “Beyond the Renderer: Software Architecture for Parallel Graphics and Visualisation”. Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, ICASE Report No. 96-75. 1997

[14] Schneider, B. “Parallel Rendering on PC Workstations” *Proceedings of 1998 International Conference on Parallel and Distributed Processing Techniques and Applications.*1998

[15] Hubbell, J. “Network rendering”. *Autodesk University Sourcebook* Vol. 2, pp. 443–453. Miller Freeman. 1996

[16] Giganet, Inc. *Giganet cLAN Family of Products*, http://www.giganet.com/products, 1999.

[17] Myrinet, *Myrinet* http://www.myri.com/myrinet/overview/index.html,1999.

[18] Myrinet, Inc. *The GM Message Passing System*, http://www.myri.com, 1999.

[19] Chen H., Wyckoff P., “Simulation studies of gigabit ethernet versus myrinet using real application cores". *Presented at CANPC00 workshop of HPCA*, January 2000.

[20] *Virtual Interface Architecture Specification*, http:// www.viarch.org

[21] Carrera E. V., Rao S., Iftode L., and Bianchini R., "User-Level Communication in Cluster-Based Servers", *In Proceedings of the 8th International Symposium on High-Performance Computer Architecture*, IEEE Computer Society, Cambridge - MA, pp. 275-286, February 2002.