A Framework for Realistic Real-Time Walkthroughs in a VR Distributed Environment

F. Pacheco[1], E.Tovar[1], H. Hansson[2], P. Altenbernd[3]

[1] ISEP-IPP, Polytechnic Institute of Porto, Portugal
e-mail: {ffp, emt}@dei.isep.ipp.pt

[2] MRTC, Malärdalen University, Sweden
e-mail : han@idt.mdh.se

[3] C-LAB, Siemens Business Services, Germany
e-mail : peter@c-lab.de

Abstract

Virtual and augmented reality (VR/AR) are increasingly being used in various business scenarios and are important driving forces in technology development. However the usage of these technologies in the home environment is restricted due to several factors including lack of low-cost (from the client point of view) high-performance solutions. In this paper we present a general client/server rendering architecture based on Real-Time concepts, including support for a wide range of client platforms and applications. The idea of focusing on the real-time behaviour of all components involved in distributed IP-based VR scenarios is new and has not been addressed before, except for simple sub-solutions. This is considered as "the most significant problem with the IP environment" [1]. Thus, the most important contribution of this research will be the holistic approach, in which networking, end-systems and rendering aspects are integrated into a cost-effective infrastructure for building distributed real-time VR applications on IP-based networks.

1. Motivation

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 user to interact with the environment in quasi real-time: change his 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, behaviour, 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 impose major research challenges, whereas the complexity of lighting phenomena associated to interactive usage further call 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. Theoretically speaking, the database of the virtual world can either be available before application start (kept on local CD-ROM, such as seen in computer games), or downloaded just before usage, such as current VRML (Virtual Reality Modelling Language) browsers do. None of these solutions satisfy the envisaged type of applications. The first option is completely unreasonable while the second is severely constrained by low network throughput and large database sizes. Extended waiting periods also make incremental downloads infeasible, so exploratory behaviour of 3D data spaces using these approaches is practically impossible. Additionally the computational demand for the rendering process is essentially placed at the client side (VRML browser).

The presented framework aims at enabling networked walkthroughs of large information spaces for interactive applications. This will be done with new concepts to the field of networked multi-user virtual environments, including a distributed adaptive real-time rendering approach.

Our 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. Research Topics in Client-Server Real-Time Networking

2.1 Time-critical Network Multimedia

Most multimedia systems cannot be considered as hard real-time systems, but if customers are used to a certain local multimedia performance, the service provider is under hard pressure to really guarantee acceptable Quality of Service (QoS) levels over networks. As a general rule, real-time techniques not only give a-priori guarantees, but also refrain from compensating timing problems by over-dimensioning hardware to be used. Hence, the use of known and new real-time research results will lead to QoS guarantees as well as to efficient operation models for servers, local area network components, and clients. We will extend the real-time research to network scheduling, admission control (including schedulability analysis), and real-time rendering. Initial attempts to apply recent real-time research results to multimedia traffic in networks have shown promising results [2]. Applying these results to IP-based and distributed VR scenarios, as intended by our research framework, has a potential of causing a major change in the way streamed media are handled. Though this part of work will focus on IP networks with bandwidth reservation (full QoS), we will also handle no/low QoS networking by our advanced adaptive rendering concept.

2.2 Response-Time Analysis for VR media on IP-QoS Networks

The main innovation in the network area is to develop an analysis method that can be used to precisely control the transmission of media (e.g., MPEG-4 streams) and improve the utilisation of network resources (embedded in a VR distributed system). The main reason for the superiority of the analysis, when compared to ordinary QoS models, is the employment of an alternative way to describe and analyse data traffic (based on Response-Time Analysis [3]). The set of parameters used to describe the traffic includes timing parameters that allow analysing the variable behaviour of the data streams more precisely than with common description models (taken from traditional QoS research). In the definition of IntServ's QoS model RSVP Guaranteed Service, a variation of the Token Bucket Model [4] is used to describe the traffic. In the proposed research framework, an alternative set of input parameters related to Response-Time Analysis is used to improve the analysis. This method will be an extension of Sjödin's Multimedia Response-Time Analysis [2] used to analyse the traffic profile and determine a sharper upper bound for the end-to-end delay than the method described in [4].

The original Multimedia Response-Time Analysis was developed for use with Asynchronous Transfer Mode (ATM) networks. However, in this project the analysis will be tailored to deal with IP-QoS networks. The higher precision of the analysis method and its orientation towards real-time applications represent an improvement over traffic shaping mechanisms like the token bucket method, with the draw-back of spending more effort into analysis. When the receiver tries to determine the bandwidth it needs to reserve using a QoS mechanism like RSVP [5], computation of the end-to-end delay, that can be achieved with a given bandwidth, is a crucial factor for efficient use of network resources. Other methods to analyse the end-to-end delay can be found in [4] for the Guaranteed Service, and in [6] for a Guaranteed Rate service.

The first result on worst-case response time analysis of non pre-emptive deadline-scheduled tasks in uniprocessor systems appeared in 1996 [7]. Since then, only few research has been performed to extend the analysis to the case of distributed real-time systems, and those few apply only to particular real-time embedded network protocols [8]. To our knowledge this project is the first approach to employ response-time analysis in the context of computing end-to-end delay bounds for IP-QoS-based networks, handling complex VR media data. This way the advantages of both using IP as the de-facto standard for multimedia applications and those derived from traditional RT processing will be combined. While there will be a focus on IntServ (having real guarantees) in the networking part of the project, the approach will be open to address reduced QoS, like DiffServ or MPLS-based QoS (which will be used in the trial), using advanced adaptive rendering.

We will combine this work with other well-know protocols taken from the IP-QoS domain (including both TCP and UDP, hence covering OSI Level 3 and higher) because these represent *de facto* standards in the field of IP-based multimedia. RTP [9] will be

used for packetizing the media data for transmission, combined with a time stamp for easier recombination at the client. Further, there has to be some monitoring (RTCP, [9]) in order to check (particularly on low/no QoS networks) if the packets have really arrived at the client (needed for adaptation). The set-up for such a network session is usually done by a protocol called RTSP [10], which provides the protocols mentioned above with data like bandwidth, length of the media, title, publisher, etc. RSVP, which represents the signalling protocol for IntServ, will be employed as a socket-based API, independent from the underlying QoS model, using pre-defined transitions between different QoS models, as needed for a user trial, in which MPLS may be used.

3. Research Topics in Rendering

Virtual Reality (VR) is strongly connected to the concept of immersion of a person into a simulated surrounding produced by a computer. The goal is to make the synthesised experience created by presenting artificial stimuli to the human senses convincing enough to ultimately making the real and virtual surrounding indistinguishable. Vital to VR is rendering: the process by which an abstract description of a scene is converted to an image. For interactive 3D graphics the generation of images by the computer must be fast enough so the user perceives animation and not a sequence of images. To achieve this, a frame rate of 10 to 50 frames per second is needed, depending on the application.

Rendering VR objects represented as polygons can be accomplished by a rendering pipeline, which includes database traversal, polygon processing and pixel processing. In database traversal visible objects are selected. Polygon processing includes co-ordinate transformation and visibility calculation, lighting calculation and 2D-triangle set-up. Finally, in pixel processing operations such as depth buffer testing, anti-aliasing and texturing are performed. For generation of images from the virtual world, there is a wide range of surface-based techniques available, such as raytracing [11] and radiosity [12]. These techniques are computationally extremely expensive, but can generate photo-realistic quality images. Simpler polygon-based rendering requires less processing, but the generated images appear synthetic [13].

3.1 Real-Time and Interactive Rendering

Timely computations in rendering have been scarcely exploited by researchers. Some authors base their efforts in exploiting Levels of Detail (LODs) of the virtual world description. The idea is to limit the number of polygons [14] or to use the computation time from previous frames to adapt the LOD of the next frame [15] to achieve a quasi-constant frame rate. More recent works, e.g., [16], use more advanced time calculation heuristics to support higher changes in computation needs from frame to frame. Another group of solutions use priority based rendering algorithms (e.g., PLP [17] or QSplat [18]) that can produce fast results, albeit with some errors.

Computer graphics researchers see "real-time" more as a set of speed-up techniques. Our research is, on the contrary, devoted to time control of interactive rendering by using techniques from traditional real-time research. The first challenge in this area is to devise efficient time-controllable rendering algorithms (raytracing or radiosity-like). The worst-case execution time of those algorithms (and variants) can change enormously depending on the viewpoint and small scene changes. The goal will be to find a balanced trade-off between quality of image and worst-case execution time. This can be done using incremental rendering algorithms, a technique that has not been addressed before with hard real-time concerns. Important challenges will be not only to parallelise the rendering pipeline but also to apply the recently introduced concept of feedback control scheduling [19] to the image rate control of RT interactive rendering. The later has not yet been applied to rendering but only to transmission/playing of streamed multimedia data [20]. Other problems that need to be handled in this context include: sharp variations in the system's workload when scheduling the rendering on a shared multi-tasking system, dramatic variation of execution requirements depending on variations in viewpoint as well as the client-server workload balancing. This means that computation time for rendering an image of a viewpoint will also vary depending on the time instant (depending on the server's and client's workload). All these properties of the rendering process make the use of feedback control applied to scheduling both challenging and promising for real-time rendering.

Another promising line of research for real-time rendering is to represent 3D environments with sets of images [21], making it possible to produce new images from arbitrary viewpoints by interpolation techniques. Since this makes rendering computation proportional to the number of output pixels rather than to the number of geometric primitives, the rendering rate and the rendering latency can be decoupled from the user’s changing viewpoint. Not much research has been performed on image-based rendering (IBR). Although this technique has some known disadvantages [22], it will be assessed in future work, envisaging a possible mixed IBR/geometry-based rendering system, as recently explored (however without RT concerns) [23].

3.2 Real-time Parallel Rendering

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

will consist of a "front-end" machine, for managing the client-server balancing, 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 [24], which are main platform requirements. Additionally, maturity and robustness of Linux/RT-Linux [25] and *de facto* standardisation of message-passing via Parallel Virtual Machine (PVM [26]) and Message Passing Interface (MPI [27]) are enabling the design of systems which are entirely entirely made up of COTS technology.

However parallelism problems in rendering are usually regarded as intractable [28]. 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. Most of the effort to date has focused on the rendering algorithms themselves and their interactions with specific architectural platforms. The question of integrating parallel renderers into the broader computing environment has often been neglected, and in most cases explicitly ignored [29]. Nonetheless, diverse research works have been published focusing on parallel rendering [28, 29, 30]. Essentially the problem with parallel rendering in a cluster deals with how to exploit time efficiency (we have a strict time limit to present an image to the user in interactive systems) using parallelisation.

One 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 [31], its actual implementation is hampered by the lack of efficient networking technologies. Recent developments of Ethernet switches [32] provide flexible and scalable solutions that overcome the traditional Ethernet non-determinism through the use of micro-segmentation and full-duplex operation (leading to a collision-free environment). Simultaneously, IEEE 802.1p gives Layer 2 switches the ability to prioritise Ethernet traffic and the IEEE Std 802.3x Flow Control provides means to control the generated traffic. Myrinet [33] has also been recently introduced, and is becoming a commodity high performance scalable RT LAN technology, with particular application to interconnect clusters of workstations, PCs, servers, or single-board computers. In this research framework we will exploit the results of a current national-level funded project investigating timing guarantees for message passing in clusters of PWS networked by these commodity RT LANs.

Another important issue is finding the optimal parallelisation for the rendering process. This is a complex problem which includes: rendering pipeline parallelisation, object (3D geometry) partitioning and image-partitioning. As in any parallel algorithm implementation, the performance depends critically on balancing the load between the parallel processors. It is expected that the development of time-controlled rendering (see previous sub-section) will constitute an important breakthrough to the problem of parallelised rendering.

A final issue is that the RT Server will have to assist multiple clients. This classical RT problem of scheduling tasks in multiprocessor systems has not yet been at all addressed by RT researchers for the case of parallel rendering.

4. Research Topics in Client-Server VR Architectures

4.1 Hybrid RT Client-Server Architecture

Networked multi-user virtual environments enable the interaction of remote users in a shared common world. Examples include networked walkthroughs of large information spaces and interactive applications. Commonly, such systems face the task of generating images from a model containing millions of elements [34]. With technologies such as 3D scanning, graphical models are becoming increasingly complex. The current trend for addressing this question is based on geometry replication: a local representation of the geometric database is stored locally (at the client) for access by the rendering process (at the client). The database can be downloaded just before usage, such as current VRML browsers do [35]. This approach is however severely constrained by low network throughput and large database sizes. Extended waiting periods also make simple incremental downloads useless.

To overcome this limitations we will explore the recently introduced concept of remote rendering pipeline [36], in which a geometry management scheme delivers 3D data over the network “just in time” to the client for the rendering process. It will be the job of server to transmit the required data to the client on demand. On the client side there will be a 3D data cache holding for immediate rendering. The goal is to keep its content at all times equivalent to the current viewpoint. Important tasks in the client rendering engine will be level of detail (LOD) selection and checking whether the 3D data cache lacks visible items or items that are likely to become visible in the near future.

However to support less powerful clients, image-based client/server services will take advantage of the multi-client RT Rendering Cluster.

4.2 Platforms for the RT Media Client and RT Front-end Server

Continuous media processing applications such as video and audio tend to vary significantly in their

request for system resources, making conservative scheduling algorithms and admission control inefficient as they do not take the particular needs of single applications into account (variable bit rates (VBR)) [37].

New media streaming formats like MPEG-4, that have emerged recently, make additional high demands on process scheduling. Real-time synchronisation methods for audio-visual objects must be considered as well as traditional characteristics of multimedia traffic like bursty resource requirements and dramatically changing workloads. These supplementary imposed requirements cannot be satisfied by traditional methods for multimedia scheduling that usually do not take the specifics of media streaming formats (like precedence) into account. Hence, new methods will be developed that allow for efficient resource utilisation, thus guaranteeing QoS for the whole client system.

Even though there are well-defined approaches for making network reservations, the scheduling of the processor / OS resources has not been paid full attention in much of the previous QoS work. Hence, there may be no guarantee for a jitter-free and non-delayed playback even though the data was properly transmitted over a reserved network. For the above mentioned scheduling in the context of complex VR data we will use RT Linux [25], in order to include the client's resources into the scenario as well. The same OS concept will be run on the server side.

5. Implementation Topics

Figure 1 illustrates the envisaged client-server VR distributed infrastructure. In the following sub-sections we make some considerations on the implementation of the end system platforms and internetworking.

5.1 Network Interfaces

One important aspect of our architecture is the underlying network. In the implementation of a system the following aspects are worth studding.

- *The description of interfaces for using network I/O*, to enable smooth support of different network architectures and easy support of RT concepts.
- *Concepts for media data mapping*, to the transport protocol RTP since some of the media formats used probably are not yet specified by the IETF.
- *Tthe transition between different network types*, used in the project. Even though it is expected in the development phase that IntServ will be available for the reservation of network resource, one cannot be sure this will be the case for later employment of the framework. Therefore, the networking interface will remain unchanged no-matter the underlying network QoS model will be (e.g. DiffServ, best-effort QoS).
- *Concepts for real-time network I/O scheduling and admission control*. When using IntServ, mathematical analysis of the data transmission is possible, because clients are then able to reserve a certain level of bandwidth that is used for their data stream. Depending on the expected traffic and the reserved bandwidth, end-to-end delay bounds can be determined before the actual transmission starts. The computation of these delay bounds is essential for the operation of real-time interactive client/server rendering system. An analysis that can be used to increase the resource utilisation profits strongly from using methods known from real-time research. Some previous work has already shown promising results in terms of increased use of available network bandwidth. Therefore, it is very likely that the overall system will profit from the use of real-time analysis methods, as well.

5.2 The RT Media Client

The proposed RT client architecture (figure 2) will reflect the flexible nature of the RT Remote Rendering concepts. On one hand, a full-featured client will have its own rendering engine that will perform most of the rendering locally. This component will be closely related to the server side Rendering Engine. However the RT rendering for the client side will be for single processor and user. The client/server balancing will be based on RT 3D data-on-demand with LOD requests to the server.

On the other hand, less powerful clients will be image-served by the RT VR Server and will have a media player for the purpose of video decoding and

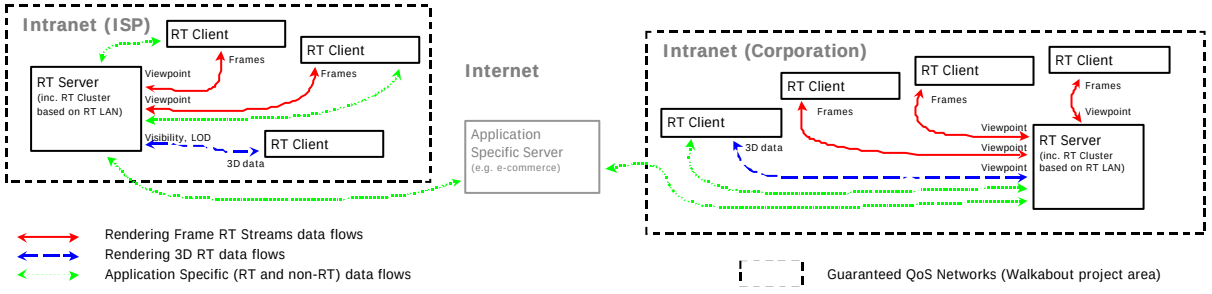

Figure 1 - Envisaged Networking Approach

display. This will be done either on the basis of integration of an existing player, or by the use of a corresponding development library. An example of the later (in the context of MPEG1) is the open-source SMPEG library [38] that is based on SDL (Simple Direct Media Layer) [39].

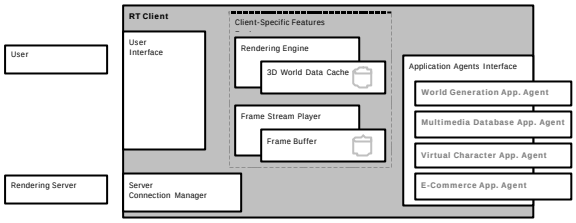

Figure 2 – Proposed Client Architecture

A Server Communication Manager will be responsible for the timely update of the 3D World Data Cache or Frame Buffer. For the RT management it will use information from the network, the RT VR Server, and about user interactions. The system will also feature User Interface elements for quick application deployment.

In order to support efficient application-specific client/server processing an Application Agent Interface will be defined and implemented.

5.3 The RT VR Server

Our base proposal for the RT VR server (figure 3) is a "front-end" server and a rendering cluster of commodity personal workstations connected by a high performance commodity RT LAN. In this design, the front-end server and the rendering engine (rendering cluster) are separate.

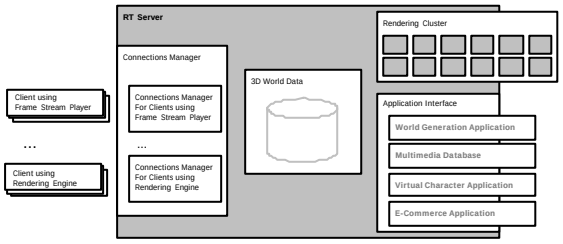

Figure 3 – Proposed Server Architecture

The front-end server will have different modules responsible for managing the server rendering facilities according to the type of client/service RT adaptation modes, to the available network resources to the rendering cluster usage, and to the client requirements. At least two types of managers will be implemented. A manager will serve connections with clients using a frame stream mode. This will incorporate advanced RT multiprocessor scheduling techniques to allocate requests to the rendering cluster and tune timing controls of the RT parallelised rendering algorithm. Another type of manager will serve connections with advanced rendering capabilities. This will incorporate mechanisms to handle RT 3D data and LOD switching client/server management. Managers will also serve optional server extensions using the application interface and will enable straightforward interconnection of the applications to the Application Agents on the client side.

Our system will need a RT cluster implementation. From the infrastructure point of view this area needs a study of its own: tight pre-run time schedulability worst-case response times for state-of-the-art commodity RT LANs (e.g., switched-Ethernet and Myrinet) will be combined with message passing standards (e.g., MPI and PVM) for RT parallel programming on top of RT operating systems (e.g. RT Linux) to provide a powerful and cost-effective cluster platform for demanding RT parallelised rendering operations. RT Linux will be considered, based on the work performed for the RT Media Client, but adapted to the server-specific characteristics including multi-user environment, more demanding network interface, interconnection to the rendering cluster, etc.

Optimal parallelisation and client/server balancing of the rendering features in the server and its implementation will be addressed in this task. One important topic will be choosing the appropriate parallelism to be implemented for the rendering pipeline. Other important aspects include the RT Geometry Selection mechanism based on viewpoints and LOD to be used by the Clients in Geometry Rendering Mode.

Application interfaces enable the support of server extensions like multimedia databases for predefined behaviour of virtual characters. This integration must be flexible enough not only to enable interactions between the applications and the server rendering process but also to enable client/server implementations using Application Agents on the RT Client side.

6. On Going Work

The presented research framework is the basis for a 3 year project proposal: Walkabout - Realistic Real-Time Walkthroughs. The consortium involved in this project proposal includes: the project co-ordinator, Siemens Business Services (Germany); Malärdalen University (Sweden), Polytechnic Institute of Porto (Portugal); Bertelsmann mediaSystems (Germany); Eptron Multimedia (Spain); Laboratory for Mixed Realities (Germany) and Q-Systems. Work has already started in five areas of activity, which are crucial to carry out in order to achieve the project's objectives:

i. *Methods for efficient resource reservation and scheduling for network I/O*. The developed

mechanisms will be evaluated and demonstrated in the following trial.

ii. *Field trial to evaluate the applicability of the developed methods*, involving a prototype application: The Virtual Street. This prototype will be tested in a major user trial. It represents a premium service application in which timing plays a dominant role, so traditional soft real-time or reduced QoS is not really appropriate. The test will be accompanied by a user survey for market analysis. The end-to-end real-time behaviour will be measured and verified during the user trial.

iii. *Development of algorithms for real-time client/server and parallel rendering*. The efficiency of the algorithms and their implementations will be evaluated and demonstrated in the above mentioned trials.

iv. *To provide deterministic behaviour of the applications on both the server and client sides.*

v. *To perform a market analysis*, with focus on gathering of the technical requirements for Walkabout and on the investigation of commercial exploitation of the Walkabout results. The objective of this activity is to show how Walkabout can be used commercially. The analysis will be accompanied by user surveys and user trials.

7. References

[1] Ovum study, *Next Generation IP Networks*, 1999.

[2] Sjödin, M. (2000). *Predictable High-Speed Communications for Distributed Real-Time Systems*. Dissertation, Uppsala University, Information Technology, DoCS 00/117.

[3] Joseph, M. and Pandya, P. (1986). *Finding Response Times in a Real-Time System*. In The Computer Journal, Vol. 29, No. 5, pp. 390-395.

[4] Shenker, S., Partridge C. and Guérin, R. (1997). *Specification of Guaranteed Quality of Service*. Internet RFC 2212. http://www.ietf.org/rfc.html.

[5] Braden, R. *et al.* (1997). *Resource ReSerVation Protocol – Version 1*. Internet RFC 2205. (RSVP: http://www.isi.edu/div7/rsvp/).

[6] Georgiadis, L., Guérin, R., Peris, V. and Rajan, R. (1996). *Efficient Support of Delay and Rate Guarantees in an Internet*. SIGCOMM 1996, pp.106-116.

[7] Spuri, M. (1996). *Analysis of Deadline Scheduled Real-Time Systems*. Technical Report No. 2772, INRIA.

[8] Tovar, E. and Vasques, F. (2000). *Non Pre-emptive Scheduling of Messages on SMTV Token-Passing Networks*. Proc. of the 12th Euromicro Conference on Real Time Systems, pp. 209-218.

[9] Schulzrinne, H. et al. (1996). *RTP – A Transport Protocol for Real-Time Applications*. Internet RFC 1889. (RTP : http://www.cs.columbia.edu/~hgs/rtp/).

[10] Schulzrinne, H. et al.(1998). *Real-Time Streaming Protocol (RTSP)*. Internet RFC 2326. (RTSP: http://www.cs.columbia.edu/~hgs/rtsp/).

[11] Enzmann, A., Kretzschzmar, L. and Young, C. (1994). *Ray Tracing Worlds with POV-RAY*. Waite Group Press.

[12] Doi, A. and Itoh, T. (1998). *Acceleration Radiosity Solutions through the use of Hemisphere-base Formfactor Calculation*. Jour. of Visualization & Computer Animation, Vol. 9, No. 1, pp. 3-15.

[13] Leung, A. (1997). *Real-Time Interactive Client-Server Terrain Rendering*. PhD Thesis, Syracuse University, New York, USA.

[14] Hesina, G. and Schmalstieg, D. (1998). *A Network Architecture for Remote Rendering*. Proceedings of Second International Workshop on Distributed Interactive Simulation and Real-Time Applications, pp. 88-91.

[15] Funkhouser, T. and S´ equin, C. (1993). *Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments*. In SIGGRAPH ’93, pp. 247–254.

[16] Gobbetti, E. and Bouvier, E. (1999). *Time-critical Multiresolution Scene Rendering*. IEEE Visualization '99.

[17] Klosowski, J. and Silva, C. (1999). *Rendering on a Budget: a Framework for Time-Critical Rendering*. IEEE Visualisation'99, pp. 115-122.

[18] Rusinkiewicz, S. and Levoy, M. (2000). *QSplat: A Multiresolution Point Rendering System for Large Meshes*. Proceedings of SIGGRAPH'2000.

[19] Stankovic, J., Lu, C, Son, S. and Tao, G. (1999). *The Case of Feedback Control Real-Time Scheduling*. Proceedings of the 11th Euromicro Conference on Real-Time Systems, pp. 11-20.

[20] Abeni, L., Palopoli, L. and Buttazzo, G. (2000). *On Adaptive Control Techniques in Real-Time Resource Allocation*. Proceedings of the 12th Euromicro Conference on Real-Time Systems, pp. 129-136.

[21] Chang, C., Bishop, G. and Lastra, A. (1999). *LDI Tree: a Hierarchical Representation for Image-Based Rendering*. Proceedings of SIGGRAPH'99.

[22] Oliveira, M. and Bishop, G. (1999). *Image-Based Objects*. Proceedings of 1999 ACM Symposium on Interactive 3D Graphics, pp. 191-198.

[23] Aliaga, G. and Lastra, A. (1997). *Architectural Walkthroughs Using Portal Textures*. Proceedings of IEEE Visualization 97, pp. 355-362.

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

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

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

[27] MPI Forum (1997). MPI-2: Extensions to the Message-Passing Interface. Technical Report MPI 7/18/97, Message-Passing Interface Forum.

[28] Bartz, D., Schneider, B. and Silva, C. (2000). *Rendering and Visualisation in Parallel Environments*. SIGRAPH 2000, course on Rendering and Visualisation in Parallel Environments.

[29] Crockett, T. (1997). *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.

[30] Schneider, B. (1998). *Parallel Rendering on PC Workstations*. Proceedings of 1998 International Conference on Parallel and Distributed Processing Techniques and Applications.

[31] Hubbell, J. (1996). *Network rendering*. Autodesk University Sourcebook Vol. 2, pp. 443–453. Miller Freeman.

[32] Alves, M, Tovar, E., Fohler, G. and Buttazzo, G. (2000). *CIDER - Envisaging a COTS Communication Infrastructure for Evolutionary Dependable Real-Time Systems*. Proceedings for the WIP and Industrial Experience Sessions of the 12th Euromicro Conference on Real-Time Systems, pp. 19-22.

[33] The Myrinet Homepage: http://www.myri.com/myrinet

[34] Yagel, R. and Ray, W. (1996). *Visibility Computation of Efficient Walkthrough of Complex Environments*. Presence, Vol. 5, No. 1, pp. 1-16.

[35] Pesce, M. (1995). *VRML: Browsing and Building Cyberspace*. New Riders.

[36] Schmalstieg, D. (1997). *The Remote Rendering Pipeline*. Dissertation, Institute of Computer Technology, Vienna University of Technology.

[37] Baiceanu, V., Cowan, C.,McNamee, D.,Pu, C. and Walpole, J. (1996). *Multimedia Applications Require Adaptive CPU Scheduling*. In Workshop on Resource Allocation Problems in Multimedia Systems.

[38] SMPEG Library by Loki Software: http://www.lokigames.com/

[39] Simple Direct Media Layer Homepage: http://www.libsdl.org