# High Performance Computing Rendering System supporting Electronic Commerce Applications in IP-Networks

Olaf Schmidt, Tomas Plachetka
University of Paderborn
Department of Computer Science
33102 Paderborn, Germany

**Abstract** *In this paper a remote parallel rendering system is described which provides a framework for advanced electronic commerce applications. The system enables the synthesis of photo-realistic images on demand by the use of advanced high performance computing (hpc) technology. The end-user has a cost-efficient access to hpc-systems, and so even small companies can profit from powerful systems which are too expensive for a local installation. In order to build an efficient and reliable remote rendering system, several research topics like resource allocation, job-scheduling in metacomputing environments and load balancing have to be addressed. An overview of these research problems is also given in this paper.*

*Keywords:* image synthesis, high performance computing, remote rendering, metacomputing, e-commerce

## 1 Introduction

Most people come across the term supercomputing in references to large, esoteric machines that have traditionally been used by universities, national laboratories and large companies for scientific and engineering "number crunching". Such machines are regarded by most IT managers as either too expensive or irrelevant to their business. However, there are many business problems that can be tackled using novel computing, simulation or optimization techniques which require powerful computers. There is a wide range of applications that don't need the ultimate performance levels provided by supercomputers, but are too demanding to run on a single microprocessor. One of these applications which becomes more and more important is the photorealistic visualization of complex environments. Leaps in graphics technology coupled with the power and capacity of high performance computing now allow engineers and architects to design and render, in real time, virtual walk-throughs of large buildings or even whole streets. In entertainment, too, computer graphics has become an increasingly important tool. Calls for more and more complex and realistic visualization will require more and more computing power of the kind offered by high performance computing.

The idea of the system described in this paper is to provide a framework for advanced electronic commerce applications which allow a user to produce complex realistic visualizations on demand by the use of hpc-technology. Our system integrates the hpc-technology with the technology for electronic commerce systems that focus on business to business scenarios. The production of high-quality visualization of 3D models is offered as an e-commerce service over the Internet. The end-users have a cost-efficient access to hpc-systems and so even small companies can use high performance computing systems which are too expensive for local installations.

A brief overview of existing parallelization

strategies for advanced rendering techniques is given in section 2. Different end-user scenarios and applications of the e-commerce service are discussed in section 3. In order to set up an efficient hpc-rendering service which is able to support different e-commerce scenarios, a number of serious research problems must be addressed. These research problems as well as the architecture of the IP-based remote rendering system are described in section 4. In section 5 some results are presented. In section 6 we draw our conclusions and outline our future research directions.

## 2 Parallel Image Synthesis

Computer graphics technology is quickly becoming part of everyday experience. Special effects for movie industry, as well as popular video games and advertising, rely more and more on sophisticated techniques for synthetic imaging. Specialized applications of computer graphics, such as computer-aided lighting design, visualization of architectural spaces and virtual show rooms are also blooming. The objective of realistic image synthesis is to generate pictures with a maximum degree of realism. The calculation of realistic images requires a precise treatment of lighting effects like color bleeding, indirect illumination, penumbras and soft shadows from light sources. This can be achieved by simulating the underlying physical phenomena of light emission, propagation and reflection. Two different classes of algorithms exist which try to simulate the global illumination; that is, they model the inter-reflection of light between all objects in a 3D-scene.
The first class contains ray tracing methods. These methods simulate the motion of light photons either by tracing the photon paths backwards from the position of an observer (eye ray tracing) or forwards from the light sources (light ray tracing). The basic ray tracing algorithm works well for scenes with many specular surfaces. Several extensions like Monte Carlo ray tracing have been proposed which take the indirect illumination of surfaces into account. Ray tracing methods are viewpoint dependent. This means that the complete simulation process must be repeated when the position of the observer changes.
The second class of global illumination methods contains radiosity methods [1]. The diffuse illumination of the scene is described by a system of linear equations. This system is independent of the position of an observer in the scene. Thus, after calculating a radiosity solution for a given scene (illumination of each surface), the viewpoint can be modified without a need of recalculating the radiosity solution.
Global illumination methods like ray tracing and radiosity algorithms currently implemented in image synthesis systems provide the necessary rendering quality, but suffer from their extensive computational costs and their enormous memory requirements. The use of scaleable massively parallel systems offers significant reductions in the time and cost requirements. A number of parallel global illumination methods have been proposed which are based on different parallelization strategies. These strategies can be grouped into two categories.
The algorithms of the first category allow each processor to access the entire database. All processing elements work independently of each other and communicate with one particular processing element which distributes work to the other processes. Even with moderately complex scenes, the inter-process communication overhead increases rapidly with the number of processors. The most important problem is that on distributed memory systems the database is replicated at each processor. A very desirable feature of parallel systems is violated, namely that larger problems should be treated simply by adding more processors to the system. If large-scale performance increases are supposed to be achieved without a limitation on the size of the problem, a data-parallel algorithms must be used. The environment is subdivided into sub-environments which are distributed among the processors. The processors solve sub-problems which are later combined to a valid solution to the input problem.

High performance computers allow simulation of the global illumination for very complex virtual environments. Unfortunately, massive parallel computers are much too expensive. A small company interested in producing high quality images and animations cannot afford to buy a high performance multiprocessor computing system to speed up their production process.

## 3 Supported Applications

The first e-commerce application which has been supported by the remote rendering-system was the interactive configuration of bathrooms [2]. Retailers of sanitary and bathroom equipment use the remote rendering system in order to support the selling process of their products. In a typical scenario the customer configures the bathroom equipment in an interactive session on a PC in the showroom of the retailer. The 3D-configuration-tool allows selection of objects from a database and their placement in a simple projection view. The local PC is capable of quickly displaying simply shaded pictures of the bathroom from different views. However, these pictures are still synthetic and abstract. Including physically correct shadows, reflections and refractions takes a much longer time than the customer is willing to invest. The high quality pictures must be computed within several minutes, while the customer is waiting. Using a simple user interface, the model of the bathroom is sent over the Internet to a rendering server. The server computes photo-realistic pictures on a parallel computer in a very short time and sends them to the PC in the showroom.

In recent years the architectural community has made a considerable investment in the generation of computer models of new buildings and existing build-up areas. Currently, performance and time considerations limit the usefulness of these models to the production of single frame ray traced images or expensively produced animations of single buildings. High performance computing systems must be employed in order to produce photorealistic walkthrough animation in complex virtual architectural environments in reasonable time periods. Another application of sophisticated rendering techniques is the production of films and commercial spots. Computer based image synthesis for creating visual effects became more and more important during the past 10 years. The rendering of computer animated sequences in image resolutions needed for the film production is almost impossible without parallel computing. The most time consuming part is the simulation of global illumination in computer generated environments (also referred to as virtual sets). The quality is critical but the production schedules and budgets are tightly controlled. Because of the lack of rendering systems which support efficient parallel simulation of global illumination, the lighting process is simplified in most of the cases. This results in images which are still impressive but far away from being physically correct. Only a few large companies can afford to invest in large workstation clusters and powerful parallel computing systems. Unfortunately, most of the hpc-systems are still much to expensive for local installations in office spaces of small companies which produce animations for architects and film industry. A remote rendering service offers a solution to this problem. This kind of an e-commerce application separates the creative process of animation design from the time consuming rendering process on high-performance computing systems.

## 4 Technical Scenario

The remote rendering-system has a hierarchical client/server-architecture (see figure 1).

In the current implementation the *client* is passive. This means that the user of the system specifies only references (URLs) to the scene files instead of sending the data to the server. It is up to the user to export the scene from the modeling software in a file format supported by the server. The files (geometry, textures, cam-

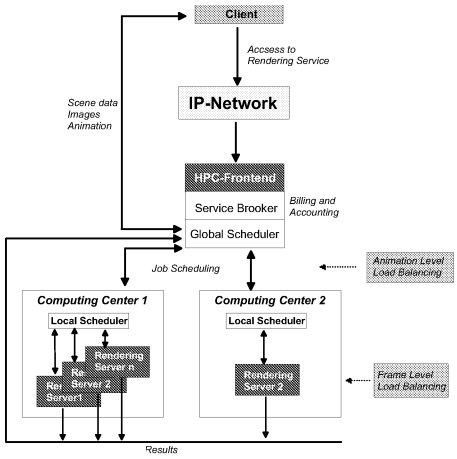


Figure 1: Hierarchical architecture of the WWW based remote rendering system

era path) are stored in the web area of a user. The user connects to the service broker, specifies job control parameters (rendering method, anti-aliasing level, resolution, the URL of the job data etc.) and submits the job. The result is sent to the user per e-mail as soon as the job is finished. An alternative to sending the results via e-mail is placing the results on WWW and send the user the referring URLs.
The *HPC-Frontend* is the interface between the client system and the remote rendering service. It is implemented as an Internet/Intranet-service and consists of two parts: *service broker* and *global scheduler*.
The service broker is an Apache Web server. It manages the administration of the users and commercial activities. After a job has been submitted to the service broker, a request for the state of the job-queue is sent to the global scheduler. The incoming job is assigned a unique ID and passed to the global scheduler. The global scheduler is responsible for the job administration. The tasks of the global scheduler are: Download of the input files; Conversion of the input; Queuing and scheduling of submitted jobs; Maintaining a database of available computing resources; Dynamical allocation and releasing of computing resources in cooperation with local access methods.
The rendering of animation sequences or the simulation of global illumination is performed on parallel computer systems using the parallel methods described in [3]. The hpc-resources may be distributed in several computing centers. Two software components must be installed in every computing center in order to provide the hardware resources for the remote rendering service scenario: a local scheduler and a (parallel) rendering server. Both are adapted to the available systems used in the computing centers.
The local scheduler is responsible for the management of the jobs sent by the global scheduler. One task of the local scheduler is to regularly report the status of all currently available hardware resources of its computing center to the global scheduler. Locally installed computing center management software is used by the local scheduler for requesting the status of the systems, allocating, modifying and releasing partitions of available parallel computers as well as for starting a rendering server on an allocated partition. For every rendering-job a partition of processors is allocated on an available parallel system. After a successful allocation, an appropriate rendering server (depends on the rendering method specified by the user) is started on the obtained partition. Results are sent to the global scheduling engine and the resources are released immediately after a job is finished. In case of an animation, the local schedulers cooperate with the global scheduler in order to perform a dynamic load balancing on animation level. The PVM message-passing library was used for the implementation of the parallel rendering server. Thus, a wide range of parallel computing systems can be employed in the scenario of the remote parallel rendering service.
The remote rendering service is offered over IP-networks to ensure a high availability of the service to many end-users. A disadvantage of a low-bandwidth connection are long communications times when sending large amounts of data over the network. Thus, results (pictures, animations, illuminated models) have to be compressed in order to keep communication overhead low.

An efficient utilization of the available heterogeneous parallel computers in a metacomputing environment can be achieved by applying sophisticated methods for resource allocation, job scheduling (see [4]) and runtime-approximation. The system described in this paper uses the administration software *CCS* (Computing Center Software) for the allocation of processor partitions and for starting jobs on the allocated partitions [5]. CCS allows to allocate a partition for a freely definable period of time on available parallel computing systems. In order to obtain a high utilization of the systems, the running time of global illumination algorithms like ray tracing or radiosity must be estimated as exactly as possible before the rendering process is started. This is an important (and partially unsolved) research problem of computer graphics (see e.g. [6]). The time needed for the computation of single pictures or animations of a virtual environment depends on the scene geometry and on the material properties of the objects in the scene as well as on the camera position. A slight modification of the camera position may cause an extreme variation in the execution time. In the rendering server scenario heuristics for the approximation of the remaining calculation time of an animation sequence as a function of the job history are used.

The integrated parallel global illumination algorithms of the rendering server (see section 2) must utilize the allocated processors very efficiently in order to process arriving jobs in shortest possible time. The goal is to achieve a linear speed-up even with a large number of processors. Thus, load balancing is one of the most important issues which has to be addressed. Efficient dynamic load balancing strategies for the integrated parallel algorithms were developed and implemented [3][7].

A new hierarchical dynamic load balancing strategy considers the special aspects of a metacomputing environment to achieve a very efficient calculation of ray traced animation sequences. This strategy also allows to guarantee the job completion times by dynamically adapting the number of processors during the processing of the current job. A cooperation of the global and local scheduler is necessary in order to perform a two level load balancing. An animation sequence is defined by a number of independent camera positions which describe the movement of the camera through a given 3D-scene. A picture (frame) is to be calculated for every camera. The basic idea of the load balancing strategy is to perform a dynamic load balancing on animation level as well as on frame level. In this scenario, additional rendering servers can be started on any available parallel system in order to speed-up the calculations when the current status of the animation production is behind the plan. Capacities of several parallel computing systems can be combined. Therefore, the global scheduler has to communicate with the local schedulers of different computing centers. The camera positions of an animation sequence are administered by the global scheduler which assigns sub-jobs to local schedulers. A sub-job contains only a fixed number of the camera positions. Once such a part of the whole animation sequence is finished, the idle local scheduler requests a new part. The global scheduler has always information of the current status of the calculation of the whole animation, and therefore can decide whether more processors are needed to meet the deadline. The advantage of this adaptive, demand-driven load-balancing on animation level is that several parallel computers can work on one animation sequence simultaneously. On the frame level, the dynamic load balancing is performed by the rendering method itself [3] obtaining an approximately linear speed-up for the calculation of single frames.

Another important aspect in the scenario of a remote rendering service is the communication overhead of the remote rendering-service. The substantial part of this overhead is caused by the transfer of the results produced during the parallel calculations (pictures, animations, illuminated models) to the client. In case of an animation, this problem can be solved by applying standard compression techniques (MPEG for animation sequences, JPEG for single im-

ages). The situation becomes much worse when meshes of polygons generated by a parallel radiosity-algorithm, are transmitted. During the radiosity calculation, the polygon-mesh is adaptively refined in order to account for high illumination gradients on large surfaces. The resulting mesh is substantially larger then the input-mesh sent by the user. Furthermore, additional lighting information are stored in the resulting mesh.
In the context of the rendering service described in this paper, two different strategies are employed in order to achieve a fast and qualitatively good compression of radiosity solutions: a mesh decimation method and a method generating radiosity textures.
A parallel mesh decimation was developed and integrated in the rendering service scenario. This method enables the reduction of the number of polygons (80 - 90 percent) without introducing visual. The parallel implementation of the mesh decimation performs a drastic simplification of extremely complex scenes in a few minutes.
Another method for the simplification of radiosity solutions is a loss-free radiosity texture generation for object surfaces. The idea of this second method is to store the illumination information into textures (bitmaps) which are mapped on the original objects. Only the textures and the mapping information have to be transferred over network instead of the large adaptively refined mesh. Both methods contribute to the efficiency increase of parallel remote rendering systems by reducing the communication overhead significantly due to the fact that much smaller files are transferred.

# 5 Results

A number of performance-measurements of the remote rendering service were performed during trial phases of industrial end-users. Parallel radiosity and parallel ray tracing were tested on the Fuijitsu/Siemens hpcLine-Sytem (192 Pentium II, 450 MHz processors) with different input-scenes of various complexity. The calculation times of the rendering methods on differing numbers of processors are presented in table 1 and in table 2. With both parallel methods a large reduction of execution times was achieved by applying more processors during the calculation process. Execution times were reduced from several hours down to some minutes in all tests.

| proc. | Office | Conference Room |
|---|---|---|
| 1 | 1h 52m 02s | 12h 51m 04s |
| 8 | 15m 50s | 1h 47m 00s |
| 16 | 9m 03s | 57m 45s |
| 24 | 7m 12s | 36m 49s |
| 32 | 6m 24s | 32m 06s |

Table 1: Calculation times for different scenes with radiosity method (99 % converged).

| proc. | 400x400 | 600x600 | 800x800 |
|---|---|---|---|
| 1 | 46m 43s | 2h 18m 02s | 3h 55m 56s |
| 12 | 5m 36s | 11m 36s | 20m 01s |
| 16 | 4m 16s | 8m 55s | 14m 56s |
| 24 | 3m 07s | 5m 58s | 10m 04s |
| 36 | 2m 16s | 4m 18s | 6m 54s |

Table 2: Calculation times for different image resulutions with ray tracing method.

The scalability properties of the parallel rendering system are demonstrated in figure 2 and figure 3. It is obvious that the methods work best, when the complexity of a scene is large and image resolution is high. Almost linear speed-up was achieved in this cases. The

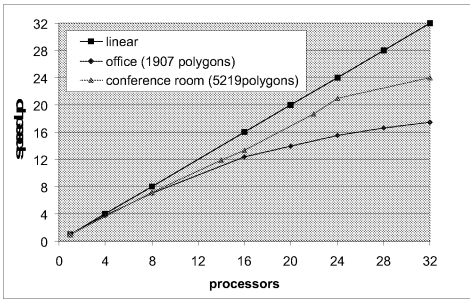


Figure 2: Speed-up for parallel radiosity

end-users of the remote rendering-system usually request for high image resolutions (up to 3000x2250 pixels) when ray tracing animations

are produced. The largest radiosity-model computed by the rendering server up to now, consisted of over 3.5 million polygons. Thus, a large job-complexity (and scalability) is guarantied by the demands of the end-users.
The turnaround times of different jobs in the

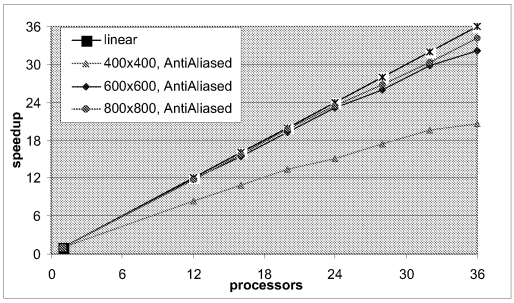


Figure 3: Speed-up for parallel ray tracing

remote rendering server scenario were also measured during our tests. The overhead (download + conversion + scheduling) is large for simple jobs but remains almost constant. For complex jobs the overhead is not significant in comparison to the rendering time. Thus, system overhead can be neglected in these cases.

# 6 Conclusions

The introduced system integrates advanced hpc-technology into electronic commerce applications that go beyond presenting products on the WWW. Even small companies can profit from the advantages of powerful hpc-systems which are to expensive for a local installation. The design of our system is kept as general as possible in order to allow the integration of other components to customize the system to the demands of additional electronic commerce applications. A prototype of the system has been integrated and tested by industrial end-users. A large performance gain has been achieved compared to the sequential rendering times of the rendering software installed locally in the offices of the end-users. The quality of the calculated pictures has been improved as well, because the remote rendering system is able to simulate complex lighting conditions in large environments more accurately. For complex jobs the system overhead is low compared to the rendering time. Future research will concentrate on the aspect of parallel running time estimation for the supported rendering methods and on an efficient scheduling of rendering tasks in a heterogeneous metacomputing environment.

# References

[1] C.M. Goral, Torrance Greenberg D.P., and Battaile G. Modeling the interaction of light between diffuse surfaces. In *Computer Graphics (SIGGRAPH 84 Proceedings)*, pages 213–223, 1984.

[2] Reinhard Lüling and Olaf Schmidt. Hipec: High performance computing visualization system supporting networked electronic commerce applications. In *EURO-PAR'98 Parallel Processing Proceedings*, number 1470 in Lecture Notes in Computer Science, pages 1149–1152. Springer, 1998.

[3] Olaf Schmidt, Jan Rathert, and Ludger Reeker. Parallel simulation of global illumination. In *Proc. of the 1998 Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'98)*, volume III, pages 1289–1296, 1998.

[4] Gehring J. and Preiss Th. Scheduling a metacomputer with un-cooperative subschedulers. In *Proc. of IPPS 99 Workshop on Job Scheduling Strategies for Parallel Processing*, Lecture Notes in Computer Science, 1999.

[5] Keller A. and Reinefeld A. Ccs resource management in networked hpc systems. In *Proc. Heterogeneous Computing Workshop HPCW 98 (Orlando)*, 1998.

[6] Reinhard E., Kok A.J., and Chalmers A.G. Cost distribution prediction for parallel ray tracing. In *2nd Eurographics Workshop on Parallel Graphics and Visualization*, pages 70–90, 1998.

[7] Olaf Schmidt and Ludger Reeker. New dynamic load balancing strategy for efficient data-parallel radiosity calculations. In *Proc. of the 1999 Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'99)*, volume I, pages 532–538, 1999.