Fifth International Conference on Computer Graphics, Imaging and Visualisation

Parallel and Distributed Visualization
The State of the Art

Ali Meligy
Middle East University for Graduate Studies, Faculty of Information Technology,
Amman, Jordan, e-mail: ameligy@ugs.edu.jo

Abstract

Visualization is one of the most important applications of computer graphics. To have a parallel infrastructure for visualization, some technologies would be needed. We identify the state-of-the-art technologies that have prepared for building such an infrastructure and examine a collection of applications that would benefit from it. We consider a broad range of scientific and technological advances in visualization, which are relevant to visual supercomputing. Mainly, we present the original abstracts from the cited papers.

Keywords: Parallel processing, distributed processing, cluster computers, parallel rendering algorithms, visual supercomputing, visualization, autonomic computing, mobile visualization.

1. Introduction

Developing parallel applications that are robust and that provide good parallel speedup across current and future multiprocessors is a challenging task [Cul99], [Bac03] and [Vok07]. In the field of parallel graphics, we can partition the computations among the available processors. For example, with n processors, we subdivide the line path into n partitions and simultaneously generate line segments in each of the subintervals. For modifications and extensions of this approach see [Mel97, Mel05, Mel06].

This article is devoted to the recent advances in *parallel and distributed visualization.* Visualization can be implemented in charts, graphs, and complex maps. If the large sets of numerical data are converted to a visual form, the trends and patterns are often immediately apparent. Mathematicians, physical scientists, and others use visual techniques to analyze mathematical functions and processes [Hua06]. We also present an overview of the state of the art of the impacts of the Internet, Grid and mobile technologies on visualization. We highlight those latest developments that are relevant, or potentially relevant, to visualization. *Autonomic computing* can play an integral role in the evolutionary development of an infrastructure for visualization. An introduction to parallel visualization is included in section 2. In Section 3, we give a more precise definition of the term *Visual Supercomputing* and outline its technical scope. In Section 4, we review major scientific and technological developments and identify the state-of-the-art technologies that have prepared us for building an infrastructure for visual supercomputing. In Section 5, we examine a collection of applications that would benefit from such an infrastructure, and discuss their technical requirements. In section 6 we discuss the mobile visualization. A conclusion is given in section 7.

2. Parallel and Distributed Visualization

The field of visualization has undergone considerable changes since its founding in the late 1980s. From its origins in scientific visualization, new areas have arisen. These include information visualization and, more recently, mobile visualization (including location-aware computing) and visual analytics. Several new trends are emerging. The most important is the fusion of visualization techniques with other areas such as machine vision, data mining and databases. Another trend, is for algorithms to be combined with usability studies to assure that techniques and systems are well designed and that their value is quantified.

The computer graphics and visualization community has been seeking for high-performance computing, and has accumulated large volumes of research outputs in parallel, distributed, and web-based techniques for visualization. The community has shown equally great enthusiasm to embrace the cluster, Grid and mobile technologies [Bro05].

Databases are ideal for data visualization. The tasks of generating displays are broken up in two different methods. The database itself is located at the **server** side, querying bridges the gap between the application program and the database. The data visualization itself can be calculated and displayed on the **client**-side taking advantage of the built in java capabilities of most browsers [Boe03], [Mah03].

The version of SQL being used should not affect the ability of visualization tools to perform their operations because it is the output data that is important to the performance of the tools [Pos03]. While the standards of

978-0-7695-3359-9/08 $25.00 © 2008 IEEE
DOI 10.1109/CGIV.2008.31

databases and other software affect data visualization, there are also standards that standardize the depiction of data [Shr04].

3. Visualization Using Supercomputers

Visual supercomputing encompasses a large collection of hardware technologies and software systems for supporting the computation and management of visualization tasks. Such an infrastructure must take *web computing*, *Grid computing* and *mobile computing* into account. Hence, it has to provide comprehensive support for visualization tasks in complex networked computing environments. The demands for visualization multiply in every direction with an increasing number of new applications, which result in new, and often conflicting, requirements. For example, in some applications (e.g. bioinformatics), the size of datasets to be processed continues to grow, while in others (e.g. mobile visualization), a careful control of data size is absolutely necessary.

A visualization process often requires a high degree of domain knowledge about the application concerned. Developments in business computing, such as electronic customer relationship management (e-CRM) [Pan03], have shown that it is possible to provide users with better quality of services with appropriate technologies that are capable of collecting and processing users' experience. The emergence of *autonomic computing* [Kep03] is gathering further momentum in developing self-managed services in a complex infrastructure. Therefore, a visual supercomputing infrastructure should have the responsibility for managing: visualization resources, visualization processes, source data and resultant data, users' interaction and communication, users' experience in accomplishing a visualization task. Remote visualization is an enabling technology aiming to resolve the barrier of physical distance. Although many researchers have developed innovative algorithms for remote visualization, previous work has focused little on systematically investigating optimal configurations of remote visualization architectures. In [Sis07], **Sisneros** et. al. study caching and prefetching, an important aspect of such architecture design, in order to optimize the fetch time in a remote visualization system. Unlike a processor cache or Web cache, caching for remote visualization is unique and complex. Through actual experimentation and numerical simulation, they have discovered ways to systematically evaluate and search for optimal configurations of remote visualization caches under various scenarios, such as different network speeds, sizes of data for user requests, prefetch schemes, cache depletion schemes, etc.

To minimize the latency in interactive visualizations across wide-area networks, Wu, et. al. [Wu08], propose an approach that adaptively decomposes and maps the visualization pipeline onto a set of strategically selected network nodes. This scheme is realized by grouping the modules that implement visualization and networking subtasks and mapping them onto computing nodes with possibly disparate computing capabilities and network connections. Using estimates for communication and processing times of subtasks, they present a polynomial-time algorithm to compute a decomposition and mapping to achieve minimum end-to-end delay of the visualization pipeline. They also present experimental results using geographically distributed deployments to demonstrate the effectiveness of this method in visualizing data sets from three application domains [Wu08]. The cost models for computation and transfer times, as well as the dynamic programming method, can be easily extended to other remote visualization systems such as Vis5D+, ParaView, ASPECT, and EnSight ([Vis07] , [Par07], [Asp07], and [Com07]) to optimize their wide-area network deployments.

4. Technologies of Visual Supercomputing

In this section, we examine how the advances in computing and communication technologies have shaped the visual supercomputing. Several technologies were developed for virtual reality [Bur03, She02]. Ray tracing for Large volume visualization has been ported to run on a cluster of PCs [DeM03]. Excellent surveys [Bar01, Bro03, Eng02, Kei02,Vit01, Whi96] and some major publications [Kou03, Sla02] include further details.

Parallel and distributed computation in visualization is broadly divided into two fundamental categories: *object space* and *image space* [Wit94]. 'Object space parallel' refers to the decomposition of a visualization task by dividing input data into a collection of smaller components, each being processed by a computation node. Algorithms in this category are also known as *sort-last* [Mol94], reflecting the need for sorting graphics primitives generated by different computation nodes at the image composition stage of a graphics pipeline. 'Image space parallel' refers to the decomposition of a visualization task into a collection of sub-tasks, each responsible for a small portion of pixels in the visualization image to be synthesized. Algorithms in this category are also known as *sort-first*, reflecting the need for organizing (or 'sorting') data according to the target sub-images prior to their entering into the graphics pipeline.

Data partitioning is important for any visualization task to be computed on parallel and distributed architectures. Further consideration includes image and frame coherence , and overlapping and exchange of boundary data [Mur03].

Data partitioning and distribution schemes may be classified according to organization of data replication, which may be in one of the following three forms:

Structured or Hierarchical Partitioning ,in which one or more higher level structures are superimposed upon the raw dataset, facilitating data decomposition based on the 'logical' organization of the data. An *occupancy map* is a simple form of such structures, which employs a binary flag to indicate whether or not a block of data is of any interest to the rendering algorithm [Mei01a, Mei01b]. A relatively more complex approach is the *Kd-tree Partitioning* [Bro04], which is used for partitioning k-dimensional space into sub-volumes along planes through the dataset. Another commonly used approach is *Octree subdivision*, which recursively divides the object-space (or an octant) into eight octants. While most structured partitioning takes places in the object-space, many of these methods can also serve image-space parallelization as they can facilitate efficient view-dependent data fetch [Lin02], and combined image and data coherence. Scene graphs were used as a hierarchical structure for managing sort-first, distributed memory parallel visualization [Bet03a,Bet03b], and facilitating real-time virtual reality applications [Nae03].

Load balancing is normally addressed by appropriate task assignment methods, which are typically classified by its run-time behaviour. *Static task assignment* [Wit94] pre-determines the workload of each processor according to the predicated workload of each sub-task and processing power of each computation node.

Image composition, which transforms parallel streams into a useful output (usually a single image), is often a bottleneck in algorithms, especially sort last algorithms. Many classical implementations use the *direct send* method, in which each processor sends its rendered pixels directly to the processor responsible for image composition. However, this simple method suffers from the problem of link contention with a large communication overhead. [Wit94] and [Ma01] proposed to organize message paths in the form of a binary-tree, together with a binary swap algorithm for improving processor utilization. Stompel *et al.* [Sto03] presented a scheduled linear image-compositing algorithm, as a highly optimized direct send method, offering better scaling on larger numbers of processors.

A future visual supercomputing infrastructure should be based on all personal computers, either loosely or tightly connected [Bro05]. The latest generations of commodity graphics cards, such as the NVidia GeForce and ATI Radeon families, are allowing more and more applications to take advantage of graphics hardware. Demanding visualization techniques such as volume rendering and ray casting have already been successfully implemented [Mur03, Rez00, Roe03]. However, there are some limitations. For example, the size of the volume that can be manipulated is limited by the amount of dedicated graphics memory available on the card, and this can easily become a bottleneck when dealing with large datasets. Texture data must be fetched via the *accelerated graphics port* (AGP) from the main memory of the PC, and this prevents interactive performance from being achieved. Sophisticated partitioning of the data can be applied as a pre-processing stage to help overcome this limitation [Cor03]. However, it will be the replacement of the AGP with technology based on the new PCI-express standard that will eventually overcome this bandwidth bottleneck [Wil03].

The TeraRecon VolumePro delivers high-quality and real-time volume rendering capability [Pfi99] Built upon the results of earlier research [Pfi96], the commercial VolumePro card currently available for PCs can deliver up to 30 frames per second for a 512^3 voxel dataset.

Several recent developments have demonstrated how graphics hardware of a PC cluster can accelerate a graphics and visualization task [Mur03,Wyl01], implementing either image-space (sort-first) or object-space (sort-last) parallelism. WireGL [Hum01] was the first of a new breed of graphics software specifically designed to make use of such cluster systems.

In addition to stereoscopic displays, one growing trend is building very large high-resolution displays, involving, for instance, 63 million pixels [Mor03]. Such a display can create an unusual sensation of *presence*, and involvement, enabling a team of users to interrogate a high fidelity model in its totality. Techniques are available for users to interact with a virtual world with 3D input devices, some of which facilitate users' experience of physical immersion [Sto00]. These include:

3D mouse, *Interactive glove* [Sla02], and *Force feedback device.* A special-purpose software is also needed to manage the virtual environment, such as the open source DIVERSE [Kel02].

The Grid is becoming more and more important in visualization, particularly when computational resources required for real time interaction in a virtual environment are not locally available. Also, the popular component-based programming paradigm, which has been adopted by many visualization systems such as VTK, AVS and OpenDX, can make use of Grid resources. This allows different computation steps of a visualization pipeline to be distributed around the globe [Sha03]. In particular, the gViz project [Bro04] has extended IRIS Explorer to work in a Grid computing framework, with authentication to allow remote execution of modules being handled by the Globus toolkit.

Augmented Reality (AR) is an extension of the traditional virtual environment technology. Most AR technologies have been based upon the use of some form of transparent display, which is positioned between the real world and the eyes of the user [Pin01]. Several AR techniques have now been shown to add value to the information available to doctors in the medical world. 3D medical datasets of a patient can be rendered in real time and overlaid onto the patient, allowing the doctor virtually to see inside the patient. This technique can also be used for medical training. Some examples of deploying this technology can be found in a recent survey [Vid04]. One approach to facilitating interaction in an AR environment is to use *Tiles* as a reference between the virtual object and the real world [Pou02].

Web-based visualization and collaborative visualization will continue to challenge the underlying technologies of a visual supercomputing infrastructure [Bro05]. Several other middleware developments took place in the same time frame as Globus but based on different principles. UNICORE [Erw01] facilitated seamless access to computing resources and integration of legacy applications. In 2002, an alliance was formed between the Globus Project and industrial partners to promote an Open Grid Services Architecture (OGSA) [Fos02].

The main problem with applying the Grid methodology, and any of the above implementations or proposed standards, to visual supercomputing is the need for interactivity with components running on the Grid. While users' interactive intervention is an integral part of many visualization tasks, it does not always fit naturally with the idea of virtual 'visualization' resources. Some sophisticated middleware components are therefore required. One interesting attempt is the development of an *Interactive Access* plug-in to the UNICORE client [Sne01], which allows end-users to interact, via the UNICORE middleware, with simulation processes running at multiple locations.

Autonomic computing [Kep03] refers to computing systems that possess the capability of self-knowing and self-management. Such a system may feature one or more of the following attributes:

A noticeable amount of research effort in autonomic computing has been placed on the self-management of system infrastructure and business services. Examples of this include self-configuration in patching management [Dun04] and Grid service composition [Aga03], self-optimization in power management [Kan04], business objectives management [Aib04], and network resource management [Nor04], and self-healing in online service management [Che04] and distributed software systems [Min03].

Efforts have also been made to broaden the scope of autonomic computing, addressing a wide range of related research issues, such as economic models [Eym03], physiological models [Lee03], interaction law [Min00], preference specification [Wal04b], ontology [Lin03, Tzi03], human-computer interaction [And03], and so forth.

Though the development of generic software environments for autonomic applications is still in its infancy, several attempts were made, which include projects such as QADPZ [Con03], AUTONOMIA [Don03] and Almaden Optimal-Grid [Dee03].

QADPZ [Con03] provides an open source framework for managing heterogeneous distributed computation in a network of desktop computers using autonomic principles. In QADPZ, the system complexity is hidden in the middleware layer, facilitating self-knowledge, self-configuration, self-optimization and self-healing.

AUTONOMIA [Don03] is a prototype software development environment that provides application developers with tools for specifying and implementing autonomic requirements in network applications and services. The use of different autonomic features permits to deliver a grid system more robust and easier to use. Future plans include integrating support for the Open Grid Services Architecture (OGSA) [Fos02].

5. Applications of Parallel and Distributed Visualization

If we were to have a Grid for visualization, what kind of applications would benefit from it, and perhaps more importantly, how would these applications necessitate specific requirements for such an infrastructure? Shalf and Bethel outlined a futuristic scenario depicting how a geophysics researcher and her international collaborators may benefit from grid-based computation and visualization. They concluded that the current state of visualization is not grid ready [Sha03]. In this section, we examine several traditional and newly emerged application areas, and discuss their requirements, especially those difficult to be met by the state-of-the-art visualization environments.

Visual data mining and large-scale data visualization

Data mining should be closely coupled with visualization [Won99]. Interactive visualization of large datasets not only demands sufficient computational resources, but also requires effective interactive techniques for data exploration, view navigation, data segmentation, data filtering, data fusion and direct manipulation [Kei02].

Data repositories at terabyte level are becoming commonplace in many applications, including bioinformatics, medicine, remote sensing and nanotechnology. In some applications, such as network traffic visualization [Kou99] and video visualization [Dan03], we are encountering the scenario that dynamic data streams are almost temporally unbounded. Many visualization tasks are evolving into visual data mining processes [Kei02].

Out-of-core algorithms (also known as *external memory algorithms*) [Vit01] are designed to solve a variety of batch and interactive computational problems by minimizing disk I/O overhead. Various out-of-core visualization algorithms have been proposed to handle large structured and unstructured 3D datasets, for instance, in the context of :

(i) isosurface extraction [Sut00], (ii) terrain rendering [Lin02], (iii) mesh simplification [Lin00], (iv) rendering time-varying volume data, and (v) rendering unstructured volumetric grids [Far01]. While some algorithms rely little on internal memory , others utilize preprocessed data structures, such as octree [and indexing to optimize disk I/O operations. Kurc *et al.* [Kur01] reported their experience in visualizing large volume datasets using *Active Data Repository*, which is composed of a set of modular services and a unified interface for supporting the management of, and mapping between, in-core and out-core data.

Other view-dependent works include visible set estimation [Klo00], visibility-based prefetching [Cor03], and view-dependent progressive rendering [Nor03].

In computer graphics and computer aided design, *scene graphs*,built upon the concept of constructive solid geometry, have played an indispensable role in combining simple objects into a complex object and bringing many objects together into a scene. It is common for graphics systems to support scene graphs, for instance, in RenderMan, OpenGL, OpenRM, VRML, Java3D, POV-ray and Open Scene Graph. However, support for combinational modelling in visualization systems [Bet03a, Nae03] is largely based on surface-based scene graphs, relying on image-space composition. Chen and Tucker [Che00] outlined the concept of *constructive volume geometry* for combining volumetric datasets and procedurally defined scalar fields. vlib [Win01], an open source volume graphics API, offers volumetric scene graphs as its fundamental data structure, and provides a discrete ray tracer for direct rendering *volumetric scene graphs*.

In large-scale data visualization, high-performance rendering techniques, such as massively parallel rendering [Ma01], and stream-based rendering [Hum02], are essential to the process of *making displayable by a computer*. With very large datasets, 'meaningful information' is often featured in a visualization at a sub-pixel level, in a large amount or in four or higher dimensions. This challenges us to develop visualization techniques into tools for visual data mining [Kei02].

A popular approach to the handling of a huge amount of visual information is the use of *focus and context* techniques, which highlight a 'focus' in detail and depict its 'context' with less details to provide an overview. This approach has also been employed in non-photorealistic rendering [Tre00], magnification lens [Lam01], two-level rendering [Had03], and digital dissection [Che03].

One of the main challenges is *computer-assisted design of visual representations*. Many techniques in information visualization enable automated placement of information in a visualization, for instance, *Sunburst* [Sta00].

Problem Solving Environments (PSEs) are 'computer systems that provide all of the computational facilities necessary to solve a target class of problems'. For example, Cactus, is an open source PSE, which was originally designed to provide a framework for solving Einstein's Equations, and gradually evolved into a 'unified modular and parallel computational framework for physicists and engineers' [All01]. While PSEs have been successfully deployed to model many problems in science, engineering and finance, new problems, including a number of grand challenge problems, continue to be formulated.

Zhou *et al.* [Zho02] proposed an approach towards automatic steering based on comparative visualization involving both experimental and computational results.

The RealityGrid project have built some impressive demonstrations of steering Lattice-Boltzmann simulations, which are massive Grid applications, involving collections of machines across the world, and are state of art in what can be achieved on a global scale [Bro03].

On a smaller scale, the gViz e-science project [Woo03] has studied two approaches to computational steering. One extends IRIS Explorer to run in secure distributed fashion across Grid machines, so an IRIS Explorer session spans the internet. The simulation runs inside IRIS Explorer. An example of mission critical systems are training simulators such as flight simulators, which have used custom built hardware to train pilots for many years both in routine flying and critical incident handling [Sog02].

Medical simulators are expected to be the next major application to benefit from simulator technology, but based on commodity graphics hardware The military is another large market for mission critical visualization. For example, the US Fleet Numerical Meteorology and Oceanography Centre was tasked with supplying military forces deployed in the Persian Gulf with highly accurate meteorological information critical to conducting land, sea and air operations [Bro05].

For example in a system for delivering interactive volume interrogation of patient data in the operating theatre [McC03], visualization tasks were carried out on a server over a mile away from the hospital and then delivered across the data network. Applications such as this raise many issues including: how to guarantee a minimum bandwidth required for both data communication and data processing; the use of redundancy for both communication and computation to ensure a reliable delivery of visualization; and the handling of secure information.

6. Mobile visualization

Related to mobile visualization, there are some specific problem areas [Kir95]:

How can multimedia data structures be exchanged efficiently across heterogeneous (especially wireless) networks? How can complex interactive graphics applications for accessing distributed data be efficiently executed on a mobile system infrastructure?

The first point discusses the problem of communication between application components, while the second addresses the question of identifying and allocating the application components themselves.

From the communication level point-of-view, there are two fundamental problems that have to be tackled :

1. Network resources may be heterogeneous and scarce, and may vary dynamically.
2. The Infoverse may contain arbitrary structures with widely varying, application-dependent access behavior (i.e., breadth-first vs. depth-first traversal).

After being able to exchange the multimedia information of the Info-verse, it is now of interest how complex interactive applications for accessing and visualizing the Infoverse can be efficiently mapped to the distributed mobile infrastructure.

Partition of the visualization process

Izadi *et al.* [Iza02] proposed the FUSE system as a development tool for collaborative systems across multiple platforms. Lamberti *et al.* [Lam03] demonstrated a mobile graphical interactive rendering task running on a PDA, which is provided by a remote graphics workstation. Wolf *et al.* [Wol02] proposed the Smart Pointer as a role for PDA devices, where it either presents a subset of the visualization when part of a larger visualization environment (such as a CAVE) or it aims to provide the same overall image as other (desktop) clients, both approaches using a remote visualization server. Hartling *et al.* [Har02] presented a middleware system, Tweek, which displays a 2D GUI to a virtual environment using a PDA. The user may interact with the virtual environment via the PDA. D'Amora and Bernardini [Dam03] developed a PDA 3D viewer that can access a remote database of CAD models. Apart from the technical aspect, human factor issues in using PDAs for visualization need to be addressed [Pas00].

7. Conclusion

In this article we have identified the state-of-the-art technologies that have prepared for building a parallel infrastructure for visualization. We considered a broad range of scientific and technological advances in visualization, which are relevant to visual supercomputing.. We have examined a collection of applications that would benefit from such an infrastructure, and discussed their technical requirements.

References

[Aga03] M. Agarwal and M. Parashar. Enabling autonomic compositions in Grid environments. *In Proc. 4th Int. Workshop on Grid Computing, pp. 34–41, 2003.*

[Aib04] S. Aiber, D. Gilat, A. Landau, N. Razinkov, A. Sela and S. Wasserkrug. Autonomic self-optimization according to business objectives. *In Proc. 1st Int. Conf. on Autonomic Comp. pp 206–213, 2004.*

[All01] G.Allen,W.Benger,T.Dramlitsch,T.Goodale, H.-C.Hege, G. Lanfermann, A. Merzky, T. Radke, E. Seidel and J. Shalf. Cactus tools for Grid applications. *Cluster Computing 4(3): 179–188, 2001.*

[And03] S. Anderson, M. Hartswood, R. Procter, M. Rounce field, R. Slack, J. Soutter and A. Voss. Making autonomic computing systems accountable: the problem of human computer interaction. *Proc.14th Int Workshop on Database and Expert Systems Applications, pp. 718–724, 2003.*

[Asp07] "Aspect," http://www.aspect-sdm.org/, 2007.

[Bac03] Bacon, J. *Concurrent Systems.* Addison Wesley, 3rd ed., London, 2003.

[Bar01] D. Bartz and C. Silva, Rendering and Visualization in Parallel Environments. *In Eurographics Tutorials, 2001.*

[Bet03a] E. W. Bethel, G. Humphreys, B. Paul and J. D. Brederson. Sort-first, distributed memory parallel visualization and rendering. *Proc. IEEE Symp. Par. and Large Data Vis. and Graphics, pp. 41– 50, 2003.*

[Bet03b] W. Bethel and J. Shalf. Consuming network bandwidth with visapult. *The Visualization Handbook,C. Hansen, C. Johnson, (Eds.)* Academic Press, 2003.

[Boe03] Boerner, Katy, Chen, Chaomei & Boyack, Kevin. *Visualizing knowledge domains*. Annual Review of Information Science & Technology, Volume 37. Medford, NJ:Information Today, Inc. (2003)

[Bro03] K. Brodlie, J. Wood, D. Duce, J. Gallop, J. Walton. Distributed and collaborative visualization. *In Eurographics 2003 State of the Art Report, 2003.*

[Bro04] K. Brodlie, D. Duce, J. Gallop, M. Sagar, J. Walton and J. Wood. Visualization in grid computing environments. *In Proc. IEEE Visualization, pp. 155–162, 2004.*

[Bro05] Ken Brodlie, et. al. Visual Supercomputing: Technologies, Applications and Challenges. *COMPUTER GRAPHICS forum, Vol 24 (2005) number 2 pp. 217-245*

[Bur03] G. Burdea and P. Coiffet. *Virtual Reality Technology*, 2nd ed. Wiley, 2003.

[Che00] M. Chen and J. V. Tucker. Constructive volume geometry. *Computer Graphics Forum* 19(4): 281–293, 2000.

[Che03] M.Chen, J.V.Tucker, R.H. Clayton and A. V. Holden. Constructive volume geometry applied to visualization of cardiac anatomy and electrophysiology. *Int. J. Bifurcation and Chaos 13(12): 3591–3604, 2003.*

[Che04] M. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan and E. Brewer. Failure diagnosis using decision trees. *In Proc. 1st Int. Conf. on Autonomic Computing, pp. 36–43, 2004.*

[Com07] "Computational Engineering International," http://www.ceintl. com/products/ensight.html, 2007.

[**Con03**] Z. Constantinescu. Towards an autonomic distributed computing environment. *Proc. 14th Int. Workshop on DB and Expert Systems Applications, pp. 699– 703, 2003.*

[**Cor03**] W. T. Corr ˆea, J. T. Klosowski and C. T. Silva. Visibility-based prefetching for interactive out-of-core rendering. *In Proc. IEEE Symp. Parallel and Large-Data Visualization and Graphics, pp. 2–8, 2003.*

[**Cul99**] David E. Culler and J. Pak Singh *Parallel Computer Architecture.* Morgan Kaufmann Publ., 1999.

[**Dam03**] B.D'Amora and F.Bernardini. Pervasive 3D viewing for product data management. *IEEE CG&A 23(2): 14– 19, 2003.*

[**Dan03**] G. W. Daniel and M. Chen. Video visualization. *In Proc. IEEE Visualization, pp. 409–416, 2003.*

[**Dee03**] G. Deen, T. Lehman and J. Jaufman. The almaden optimalgrid project. *In Autonomic Computing Workshop Proc. 5th Int. Workshop on Active Middleware Services, 2003.*

[**DeM03**] D.E. DeMarle et al. Distributed Interactive Ray Tracing for Large Volume Visualization. *IEEE Symposium on Parallel Visualization and Graphics, pages 87-94, October 2003.*

[**Don03**] X. Dong, S. Hariri, L. Xue, H. Chen, M. Zhang, S. Pavuluri and S. Rao. Autonomia: an autonomic computing environment. *Proc. IEEE Int. Conf. on Performance, Computing, and Communications, pp. 61–68, 2003.*

[**Dun04**] J. Dunagan, R. Roussev, B. Daniels, A. J. C. Verbowski and Y. Wang. Towards self-managing software patching process using black-box persistent-state manifestes. *Proc. 1st Int. Conf. on Autonomic Computing, pp. 106– 113, 2004.*

[**Erw01**] D.W. Erwin and D.F.Snelling. UNICORE: a Grid computing environment. *Lecture Notes in Computer Science 2150, 825–834, 2001.*

[**Eym03**] T.Eymann, M.Reinicke, O.Ardaiz, P.Artigas, F.Freitag and L. Navarro. Self-organizing resource allocation for autonomic network. In Proc. 14th Int. Workshop on Database and Expert Systems Applications, pp. 656– 660, 2003.

[**Far01**] R. Farias and C. T. Silva. Out-of-core rendering of large, unstructured grids. *IEEE CG&A* 21(4): 42–50, 2001.

[**Fos02**] I. Foster, C. J. N. Kesselman and S. Tuecke. The physiology of the Grid: an open Grid services architecture for distributed system integration. *In Open Grid Services Infrastructure WG, 2002.*

[**Had03**] M.Hadwiger, C.Berger and H.Hauser. High-quality two level volume rendering of segmented data sets on consumer graphics hardware. *In Proc. IEEE Visualization, pp. 301–308, 2003.*

[**Har02**] Mark J. Harris et al. Physically-based visual simulation on graphics hardware. *In Proceedings of the conference on Graphics hardware 2002, pages 109-118. Eurographics Association, 2002.*

[**Hua06**] Huang, J. et. al.
A distributed execution environment for large data visualization.,
Journal of Physics: Conference Series 46 (2006) 570–576

[**Hum01**] G. Humphreys, M. Eldridge, I. Buck, G. Stoll, M. Everett and P. Hanrahan. WireGL: a scalable graphics system for clusters. *In Proc. ACM SIGGRAPH , 2001.*

[**Hum02**] G. Humphreys, M. Houston, R. Ng, R. Frank, S. Ahern, P. D. Kirchner and J. T. Klosowski. Chormium: a stream-processing framework for interactive rendering on clusters.
In Proc. ACM SIGGRAPH , 2002.

[**Iza02**] S.Izadi, P.**Coutinho**, T.Rodden and G. Smith. The FUSE platform: supporting ubiquitous collaboration within diverse mobile environments. *Automated Software Engineering 9(2): 167–186, 2002.*

[**Kan04**] N. Kandasamy, S. Abdelwahed and J. Hayes.
Self-optimization in computer systems via on-line control: applications to power management.
In Proc. 1st Int. Conf. on Autonomic Computing, pp. 54–61, 2004.

[**Kei02**] D. A. Keim, W. Muller and H. Schumann. Visual data mining. *In Eurographics State of the Art Reports, 2002.*

[**Kel02**] J. Kelso, L. E. Arsenault, S. G. Satterfield and R. D. Kriz. DIVERSE: a framework for building extensible and reconfigurable device independent virtual environments. *In Proc. IEEE Virtual Reality, pp. 183–190. 2002.*

[**Kep03**] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *IEEE Computer, pp. 41–50, 2003.*

[**Kir95**] Kirste, T.: An infrastructure for mobile information systems based on a fragmented object model. *Distrib. Syst. Engng. 2 (1995), pp. 161-170.*

[**Klo00**] J. T. Klosowski and C. T. Silva.
The prioritized-layered projection algorithm for visible set estimation. *IEEE Trans. Visualization and Computer Graphics 6(2): 108– 123, 2000.*

[**Kou03**] M. Koutek. *Scientific Visualisation in Virtual Reality: Interaction Techniques and Application Development*.
Delft University of Technology, 2003.

[**Kou99**] E. E. Koutsofio. Visualizing large-scale telecommunication networks and services. *In Proc. IEEE Visualization, 1999.*

[**Kur01**] T. Kurc, C. Chang, A. Sussman and J.Saltz.
Visualization of large data sets with the active data repository. *IEEE CG&A* 21(4): 24–33, 2001.

[**Lam01**] E. LaMar, B. Hamann and K. I. Joy. A magnification lens for interactive volume visualization. *Proc. IEEE Pacific Conf. Comp. Graphics and Appl., pp. 223–231, 2001.*

[**Lam03**] F.Lamberti, C.Zunino, A.Sanna, A.Fiume and M.Maniezzo. An accelerated remote graphics architecture for PDAS. *Proc. 8th Int Conf on 3D Web Techn., pp. 55–ff, 2003.*

[**Lee03**] A. Lee and M. Ibrahim. Emotional attributes in autonomic computing systems. *In Proc. 14th Int. Workshop on Database and Expert Systems Applications, pp. 681– 685, 2003.*

[**Lin00**] P. Lindstrom. Out-of-core simplification of large polygonal models. *Proc. ACM SIGGRAPH* , pp. 259–262, 2000.

[**Lin02**] p. Lindstrom and V. Pascucci.
Terrain simplification simplified:A general framework for view-dependent out-of-core visualization. *IEEE Trans. Visualization and Computer Graphics* 8(3): 239–254, 2002.

[**Lin03**] C. Linn. Semantic reliability in distributed systems: ontology issues and system engineering. *Proc. IEEE/WIC Int. Conf. on Web Intelligence, pp. 292–300, 2003.*

[**Ma01**] K.-L. Ma and S. Parker.
Massively parallel software rendering for visualizing large-scale data sets. *IEEE CG&A 21(4): 72–83, 2001.*

[**Mah03**] Mahovsky, J. and Bendicenti, L
An Architecture for Java-Based Real-Time Distributed Visualization. *IEEE Transactions on Visualization and Computer Graphics. Vol. 9 No. 4 (2003)*

[**McC03**] R. F. McCloy and N. W. John
Remote Visualization of Patient Data in the Operating Theatre during helpatopancreatic surgery
Computer Assisted Radiology and Surgery, pp. 53-58, 2003.

[**Mei01a**] M. Meissner, S.Grimm, W. Straβ er, J. Packer and D.Latimer Parallel Volume Rendering on a Single-Chip SIMD Architecture. *IEEE 2001 Symposium on Parallel and Large-Data Visuallization and Graphics, San Diego, Ca, October, 2001.*

[**Mei01b**] M. Meissner, et. al. Accelerating Volume Rendering on a Single-Chip SIMD Architecture. *Proc. IEEE Int. Symposium on Circuits and Systems, Vol. 2, pp. 757-760, 2001.*

[**Mel97**] A. M. Meligy and H. Nassar A Parallel Algorithm for Computer Graphics on Systems with Shared Memory. *Annual Conf. Stat. Comp. Sci. OR, Cairo University, 1997.*

[**Mel05**] Meligy, Aly M. Reuse-Based Software Development for Parallel Graphics on Hybrid Architectures. *European J. Sci. Res. Vol. 11, Nr. 2, Nov. 2005*

[**Mel06**] Meligy, Aly M. *Parallel Computer Graphics*. Dar ElNahda, Cairo, 2006.

[**Min00**] N. Minsky and V. Ungureanu. Law-governed interac tion: a coordination and control mechanism for hetero geneous distributed systems. *In ACMTrans. Software Engineering and Methodology, 2000.*

[**Min03**] N. H. Minsky. On conditions for self-healing in distributed software systems. *Autonomic Computing Workshop Proc. 5thInt. Workshop on Active Middle-ware Services, 2003.*

[**Mol94**] S. Molnar, C. M, D. Ellsworth and H. Fuchs.
A sorting classification of parallel rendering. *IEEE CG&A* 14(4): 23–32, 1994.

[**Mor03**] K. Moreland and D. Thompson. From cluster to wall with VTK. *Proc. IEEE Symp. Parallel and Large-Data Visualization and Graphics, pp. 25–32,* 2003.

[**Mur03**] S. Muraki, E. B. Lum, K.-L. Ma, M. Ogata and X. Liu.
APC cluster system for simultaneous interactive volumetric modeling and visualization. *In Proc. IEEE Symp. Parallel and Large-Data Visualization and Graphics, pp. 95–102,* 2003.

[**Nae03**] M. Naef, E. Lamboray, O. Staadt and M. Gross.
The blue-c distributed scene graph. *In Proc. IEEE Virtual Reality, pp. 275–276, 2003.*

[**Nor03**] A. Norton and A. Rockwood.
Enabling view-dependant progressive volume visualization on the grid. *IEEE CG&A 23(2): 22-31, 2003.*

[**Nor04**] J. Norris, K.Coleman, A.Fox and G.Candea. OnCall: defeating spikes with a free-market application cluster. Proc. 1st Int. Conf. on Autonomic Computing, pp. 1198–205, 2004.

[**Pan03**] S. L. Pan and J.-N. Lee.
Using e-CRM for a unified view of the customer. *Communications*

of the ACM 46(4): 95–99, 2003.

[**Par07**] “Paraview,” http://www.paraview.org/HTML/Index.html, 2007.

[**Pas00**] J.Pascoe, N.Ryan and D.Morse. Using while moving: HCI issues in fieldwork environments. *ACMTrans. Computer-Human Interaction 7(3): 417–437, 2000.*

[**Pfi96**] H. Pfister and A. Kaufman. Cube-4—a scalable architecture for real-time volume rendering. *In Proc. ACM/IEEE Symp. Volume Rendering, pp. 47–54, 1996.*

[**Pfi99**] H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer and L. Seiler. The VolumePro real-time ray-casting system. *In Proc. ACM SIGGRAPH , pp. 251–260, 1999.*

[**Pfi00**] H. Pfister and A. Kaufman Surfels: surface elements as rendering primitives. *In Proc. ACM SIGGRAPH , pp. 335-342,2000.*

[**Pin01**] C. Pinhanez. Augmenting reality with projected interactive displays. *Proc.Int. Symp. Virtual and Augmented Architecture, 2001.*

[**Pos03**] Post, F.; Nilson, G.; and Bonneau, G. *Data Visualization, the State of the Art.* Kluer Academic Publisher, Boston.(2003)

[**Pou02**] I. Poupyrev, D. S. Tan, M. Billinghurst, H. Kato, H. Regenbrecht and N. Tetsutani. A survey of augmented reality. *IEEE Computer, 2–9, 2002.*

[**Rez00**] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner and T. Ertl. Interactive volume rendering on standard PC graphics hardware using multi-textures and multi-stage-rasterization. *Proc. Eurographics/SIGGRAPH Workshop on Graphics Hardware, pp. 109–118, 147, 2000.*

[**Roe03**] S. Roettger, S. Guthe, D. Weiskopf, T. Ertl and W. Strasser. Smart hardware accelerated volume rendering. In Proc. EUROGRAPHICS/IEEE-TCVG Symp. Visualization, 2003.

[**Sha03**] J. Shalf and E. W. Bethel. The Grid and future visualization systems architectures. *IEEE CG&A 23(2): 6–9, 2003.*

[**Shr04**] Shriffin, Richard & Börner, Katy. Mapping knowledge domains. *PNAS*, vol. 101, suppl. 1, pp.5183-5 (2004).

[**Sis07**] R. Sisneros, C. Jones, J. Huang, J. Gao, B.Park, and N. F. Samatova A Multilevel Cache Model for Runtime Optimization of Remote Visualization *IEEE Trans. Vis. Computer Graphics, VOL. 13, NO. 5, 2007*

[**Sla02**] M. Slater, A. Steed and Y. Chrysanthou. *Computer Graphics and Virtual Environments.* Addison Wesley, 2002.

[**Sne01**] D. Snelling. UNICORE: A grid computing environment. *In Proc. EUROPAR August, 2001.*

[**Sog02**] F.Sogandares. Stone axes and warhammers: a decade of distributed simulation in aviation research. *Proc. 16th Workshop on Parallel and Distributed Simulation, pp. 125–132, 2002.*

[**Sta00**] J. Stasko and E. Zhang. Focus+context display and navigation techniques for enhancing radial space-filling hierarchy visualization. *In Proc. IEEE Information Visualization, pp. 57–65, 2000.*

[**Sto00**] R. J. Stone. Haptic feedback: A potted history from telepresence to virtual reality. *Proc. 1st Int. Workshop on Haptic Human-Computer Interaction,vol. LNCS 2058, pp. 1–7, 2000.*

[**Sto03**] A. Stompel, K.-L. Ma, E. B. Lum, J. Ahrens and J. Patchett. SLIC: Scheduled linear image compositing for parallel volume rendering. *In Proc. IEEE Symp. Parallel and Large-Data Visualization and Graphics, 2003.*

[**Sut00**] P. M. Sutton and C. D. Hansen. Accelerated isosurface extraction in time-varying fields. *IEEE Trans. Visual ization and Computer Graphics 6(2): 98–107, 2000.*

[**Tel03**] A. Telea and J. J. van Wijk. 3D IBFV: hardware-accelerated 3D flow visualization. *Proc. IEEE Visualization, pp. 233–240, 2003.*

[**Tre00**] S. M. F. Treavett and M. Chen. Pen-and-ink rendering in volume visualization. *Proc. IEEE Visualization 2000, pp. 203–210, 2000.*

[**Tzi03**] G.Tziallas and B. Theodoulidis. Building autonomic computing systems based on ontological component models. *Proc. 14th Int. Workshop on Database and Expert Systems Applications, pp. 674–680, 2003.*

[**Vid04**] F. P. Vidal, F. Bello, K. Brodlie, N. W. John, D. Gould, R. Phillips and N. Avis. Medical visualization and virtual environments. *In Eurographics State of the Art Reports, 2004.*

[**Vis07**] “Vis5d+,” http://vis5d.sourceforge.net/, 2007.

[**Vit01**] J. S. Vitter. External memory algorithms and data structures: dealing with massive data. *ACM Computer Survey 33(2): 209–271, 2001.*

[**Vok07**] Vokorokos, L.; Perhač, J.; and Madoš, B. , Data Visualization in Parallel Environment Based on the OpenGL Standard. 5th Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence and Informatics January 25-26, 2007, Poprad, Slovakia

[**Whi96**] S. R. Whitman. A survey of parallel algorithms for graphics and visualization. *In Proc. Int. Workshop on High Performance Computing for Computer Graphics and Visualisation, pp. 3–22, 1996.*

[**Wil03**] A. Wilen, J. P. Schade and R. Thornburg. *Introduction to PCI Express: A Hardware and Software Developer’s Guide*. Intel Press, 2003.

[**Win01**] A. S. Winter and M. Chen. vlib: a volume graphics API. *In Proc. Volume Graphics, pp. 133–147, 2001.*

[**Wit94**] C. M. Wittenbrink and M. Harrington. A scalable MIMD volume rendering algorithm. *In Proc. 8th Int. Parallel Processing Symp., pp. 916–922, 1994.*

[**Wol02**] M.Wolf, Z.Cai, W.Huang and K.Schwan. Smartpointers: personalized scientific data portals in your hand. In *Proc. ACM/IEEE Conf. on Supercomputing, pp. 1–16, 2002.*

[**Won99**] P. Wong Visual Data Mining. *IEEE CG&A 19(5): 20-21, 1999.*

[**Woo03**] J.Wood, K. Brodlie and J.Walton. gViz—visualization and steering for the Grid. *In Proc. e-Science All Hands Meeting, 2003.*

[**Wu08**] Q. Wu, J. Gao, M. Zhu, N. S.V. Rao, J. Huang, and S. S. Iyengar, Self-Adaptive Configuration of Visualization Pipeline over Wide-Area Networks. *IEEE Trans. Computers, VOL. 57, NO. 1, 2008*

[**Wyl01**] B. Wylie, C. Pavlakos, V. Lewis and K. Moreland. Scalable rendering on PC clusters. *IEEE CG&A 10(5): 60– 67, 1990.*

[**Zho02**] H. Zhou, M. Chen and M. Webster. Comparative evaluation of visualization and experimental results using image comparison metrics. *In Proc. IEEE Visualization, pp. 315–322, 2002.*