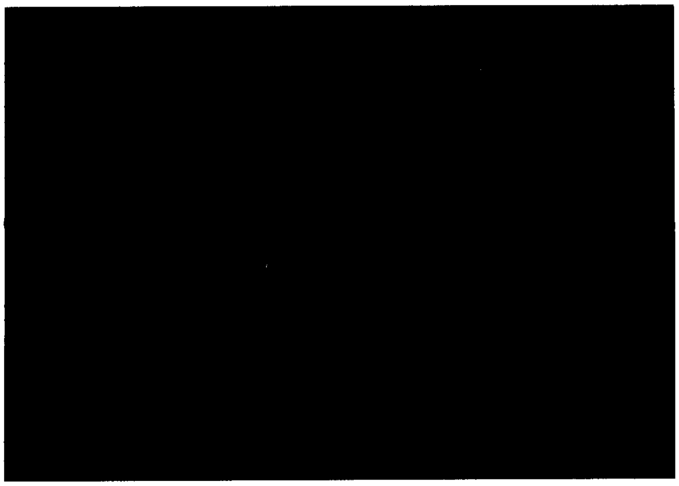
Recent Developments in Parallel Rendering



Scott Whitman
David Sarnoff Research Center
Charles D. Hansen
Los Alamos National Laboratory
Thomas W. Crockett
Institute for Computer Applications in Science and Engineering

Using parallel computers for computer graphics rendering dates back to the late 1970s. Several papers published then focused on image space decompositions for theoretical parallel machines. Early research concentrated on algorithmic studies and special-purpose hardware, but the growing availability of commercial parallel systems added a new dimension to parallel rendering.

Shortly after parallel machines became available in the mid-1980s, computer graphics algorithms (principally ray tracing and fractal geometry rendering) were implemented on these platforms. The initial problems were considered embarrassingly parallel, primarily because the data sets were replicated on every processor, and thus very little communication was required. Later, polygon scan-conversion algorithms were implemented, and more recently, direct volume-rendering algorithms.

The problems encountered with these more demanding algorithms typify all problems run on parallel computers: How do you partition a large data set among the computer's memories to minimize communication and network contention, maximize processor utilization, and achieve good speedup relative to a serial approach?

Much of this special issue focuses on multiple instruction, multiple data stream (MIMD) computers. Current machines typically incorporate tens to hundreds of microprocessors that communicate via message passing or remote memory references. Commercial machines in this category include the Thinking Machines CM-5, Intel Paragon, Fujitsu VPP500, IBM SP2, Kendall Square Research KSR-2, Cray T3D, and Meiko CS-2. Some researchers have used single instruction, multiple data stream (SIMD) architectures, but this class of system is not represented in the articles published here (see instead the *Parallel Rendering Symposium Proceedings*, ACM Press, New York, October 1993). The MasPar MP-2 is a current example of a SIMD machine.

While parallel rendering is certainly not new, prior work on the topic was scattered across various journals and conference proceedings, many of them obscure. (The articles printed here contain references to some of this work.) Moreover, researchers in this emerging area had trouble finding an appropriate forum for their work, which often contained too much parallel computing content for the graphics community and too much graphics content for the parallel computing community. Earlier venues for parallel rendering included 1989 and 1990 Siggraph courses, the 1990 conference Parallel Processing for Computer Vision and Display in Leeds, UK, and the 1993 Eurographics Rendering Workshop in Bristol, UK.

In response to growing research interest in parallel rendering and the limited opportunities for presenting results, we organized the 1993 Parallel Rendering Symposium. This meeting, sponsored by the IEEE Computer Society Technical Committee on Computer Graphics in cooperation with ACM Siggraph, took place in conjunction with the Visualization 93 conference. The call for papers attracted 39 submissions, of which we selected 15 for inclusion in the proceedings. Four of the six articles in this special issue are updated and revised versions of papers presented at the Symposium. The fifth was invited, evolving from a draft submitted for the symposium. The sixth was submitted directly to *IEEE CG&A* and is included here because of the common subject matter. Together, these articles provide a snapshot of current research in parallel rendering.

Given the expanding use of parallel computing, we believe computer graphics specialists need to understand the issues involved in effectively producing visual output from these systems. As the articles in this issue demonstrate, parallel rendering offers the potential for high performance. Realizing this potential requires careful analysis of the algorithms in light of the system's architectural parameters. Significant obstacles remain, principally in the areas of scalability, load balancing, and image composition.

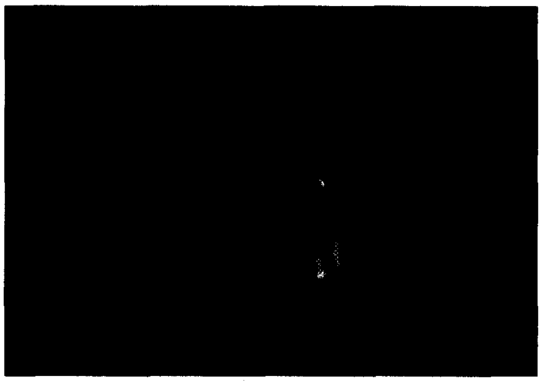
While one could use general-purpose multicomputers for graphics rendering, most of us are more familiar with high-end graphics architectures such as those commercially offered by Silicon Graphics, Evans & Sutherland, Kubota Pacific, and Hewlett-Packard, or research architectures such as the University of North Carolina's Pixel-Planes and PixelFlow, and the David Sarnoff Research Center's Princeton Engine. These systems also exploit parallelism to accelerate rendering, although it is usually incorporated in specialized hardware.

0272-17-16/94$4.00 © 1994 IEEE

About the images

The head art for this introduction comes from David Banks and Thomas Crockett of ICASE, Bart Singer of High Technology Corp., and Ron Joslin of NASA Langley Research Center. The image depicts a hairpin vortex, a structure that often arises in the transition from laminar to turbulent fluid flow. It contains 245,616 triangles and was rendered on an Intel Paragon XP/S using a parallel graphics library developed at ICASE. The vortex tube was reconstructed from an early time-step of a fluid dynamics simulation that accumulated 2,000 Cray 2 hours over a period of two calendar years. The green color indicates regions of large angular velocity on the surface of the vortex tube, while red areas have lower angular velocity. The dark grooves show streamlines on the tube's surface. They have been displaced inward to reinforce the visual perception of a rotating vortex.

The figure in this sidebar comes from Andrei State, Senior Research Associate, University of North Carolina at Chapel Hill. The image shows a simulated radiation treatment plan for cancer patients, developed within the Vistanet Gigabit Network Project. The image was generated on the Pixel-Planes 5 engine, which features general-purpose MIMD nodes and massively parallel SIMD pixel processors. The renderer displays low-resolution frames at interactive rates and progressively refines the image during user pauses. Patient anatomy and dose distribution (a multimodal 128 × 128 × 44 voxel data set) are shown via translucent and opaque isosurfaces. A cutplane displays dose isocurves. The wireframe elements are treatment beams.



About the articles

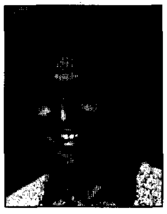
The article by Molnar et al. presents a taxonomy of parallel rendering algorithms applicable to both hardware- and software-based approaches. The authors classify algorithms according to where in the graphics pipeline communication occurs and show that different approaches work better in different cases.

Ellsworth presents a polygon-rendering algorithm designed for a large-scale message-passing system. His approach reduces communication overheads using a store-and-forward scheme and incorporates a dynamic load-balancing mechanism that exploits frame-to-frame coherence. Whitman's article also presents a dynamic load-balancing scheme, but in the context of a shared-memory architecture. His approach achieves high performance with minimal load-balancing overhead. Whitman's algorithm performs fine-grain load balancing during the rendering of a single frame, while Ellsworth uses the results from previous frames to load-balance the next frame in an animation.

On the topic of parallel volume rendering, Neumann provides a theoretical analysis of communication costs for several approaches to ray-cast volume rendering on mesh architectures. His results indicate that object-space decompositions scale better than image-space methods on this class of machine. Ma and his coauthors also use an object-space approach, but focus on the problem of efficiently compositing the partial images that result. They present results for a large MIMD message-passing system as well as for a network of workstations.

While current research in parallel rendering focuses on volume rendering and polygon scan conversion, much of the early work in this area concentrated on ray-tracing algorithms. The final article in this issue, by Badouel et al., summarizes several of the strategies used for parallel ray tracing on MIMD machines. The authors present experimental results for two different approaches, one based on explicit message passing and the other using a software implementation of shared memory.

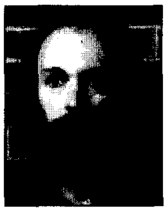
Parallel rendering as a whole is by no means a mature field. As you will see in this issue, efficient parallel rendering is an elusive goal, with more questions raised than answered. As parallel computers become more powerful and more affordable, integrating graphics with parallel applications will be a central issue for the visualization community. Efficient parallel rendering algorithms are a prerequisite to this process. ❑



Scott Whitman is a member of the technical staff at David Sarnoff Research Center. His research interests include polygon and volume-rendering algorithms, parallel computing, and computer architecture. In the field of parallel rendering, Whitman has published a book and numerous articles, and chaired several Siggraph courses. He received his PhD and MS in computer science from Ohio State University and his BS in applied mathematics from Carnegie Mellon University. He is amember of ACM and Siggraph, and the IEEE Computer Society.

Whitman can be reached at David Sarnoff Research Center, CN 5300, Princeton, NJ 08543, e-mail slim@sarnoff.com.

Charles D. Hansen's photo and biography appear at the end of the article he coauthored, on page 68.



Thomas W. Crockett is a staff scientist at the Institute for Computer Applications in Science and Engineering (ICASE). His research interests include parallel methods for graphics and visualization, system software for parallel computers, and performance measurement and analysis. He received his BS in mathematics from the College of William and Mary in 1977. He is amember of ACM and Siggraph, and an affiliate of the IEEE Computer Society.

Crockett can be reached at ICASE, MS 132C, NASA Langley Research Center, Hampton, VA 23681, e-mail tom@icase.edu.