Distributed Volume Modelling and Collaborative Visualization*

Chandrajit Bajaj Daniel Schikore Guoliang Xu

Department of Computer Science,
Purdue University,
West Lafayette, IN 47907

Telephone: 317-494-6531
FAX: 317-494-0739
Email: bajaj@cs.purdue.edu

Abstract

In this brief paper we describe a conferenced environment that we have built in a client-server networked environment called SHASTRA to support distributed volume modelling and collaborative visualization. The SHASTRA environment allows multiple users to share and interact over extremely large volume data sets while viewing multiple isosurfaces and renderings with independent viewing directions and cutaways. The distributed modelling and visualization algorithms uses the computational power of multiple networked workstations to speedily produce piecewise trivariate polynomial finite elements (modelling) and translucent shaded images of isosurfaces as well as volume rayshaded renderings (visualization) of extremely large data sets. The color graphics in SHASTRA are built on top of XS, a machine independent 3D-graphics and windows library, that runs on multiple platforms in a heterogeneous environment.

1 Introduction

Measurement-based volumetric data sets arise for example from medical imaging (Computed Tomography – CT, Magnetic Resonance Imaging – MRI), atmospheric, geological, geophysical measurements. Synthetic volume data sets are generated for example by computer based simulation such as meteorological, thermodynamic simulations, finite element stress analyses and computational fluid dynamics. Modelling the information contained in these, typically huge, data sets via trivariate polynomial finite element approximations provides mechanisms to allow querying, interaction and manipulation. Volume visualization provides mechanisms to express information contained in these, typically huge, data sets via translucent displays of isosurfaces or volume renderings – the challenge, of course, lies in making these images easy to understand. In [3] we present details of distributed volume rendering computations in SHASTRA.

Our goal is to depart from traditional single user systems and build computer-enhanced multi-user collaborative scientific visualization and analysis environments [2]. Such CSCW (Computer Supported Cooperative Work) environments provide support for collaboration in the problem-solving phase, as well as in the review phase [9]. A conferenced collaborative volume visualization environment lets multiple users on a network share a volume data set, simultaneously view shaded volume renderings of the data, and interact with multiple views, cutaways and iso-surfaces. It provides facilities for interactive control and specification of the visualization process. We have adopted a hybrid strategy to benefit from distributed systems. Distributing the output of a high computation

*Supported in part by NSF grants CCR 92-22467, DMS 91-01424, AFOSR grants F49620-93-10138, F49620-94-1-0080, ONR grant N00014-94-1-0370 and NASA grant NAG-93-1-1473.

Figure 1: A Stack of C^1 Smooth Cubic Spline Contours which are Iso-Curves from a Stack of MRI Data Slices

task emphasizes sharing of resources among applications. In addition, partitioning a high computation task, and distributing it, accords us the benefit of parallelism of distribution [8]. The distributed system, thus, serves as a high-performance multi-user virtual machine for large finite element and isosurface calculations as well as volume rendering computations and collaborative visualization.

In §2 of this paper we present details of distributed trivariate polynomial finite element computations for both dense cuberille data (CT/MRI) as well as volumetric scattered data (flow simulations on unstructured grids or meshes). In §3 we elaborate on the collaborative visualizations of multiple isosurfaces and volume renderings within the SHASTRA environment. Timing details of an implementation of distributed volume rendering computations in SHASTRA are given in [3].

2 Modelling for Visualization

We construct smooth trivariate polynomial finite element approximations to model and visualize scalar fields in three dimensions. These finite elements are higher dimensional analogues of traditional piecewise smooth interpolatory and approximatory surface patches [5, 7]. For CT/MRI cuberille data the trivariate finite elements are C^1-smooth and either tri-quadratic or tri-cubic degree and generalizations of the C^0 continuous piecewise linear or trilinear approximations [10, 11]. After constructing these piecewise trivariate polynomial approximants one can efficiently compute stationary points or intrinsic curves on any iso-surface (iso-contour), perform inter-surface calculations as well as simultaneously view several iso-surfaces or arbitrary cross-sections.

One of our distributed algorithms to construct these piecewise approximants utilizes the fact that three dimensional CT/MRI cuberille image data comes in a stack of parallel slices. In this approach we consider each slice data as lying in three dimensional space with the spatial coordinates x, y being two dimensions and the CT/MRI density values yielding the third dimension w. We then fit a C^1 smooth piecewise bi-cubic polynomial function $W(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} w_{ij} B_i^3(u(x)) B_j^3(v(y))$ to simultaneously interpolate and least-squares approximate this dense data, with a small number of polynomial pieces [6]. Here $B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$ is the degree n Bernstein-Bezier polynomial, and $u(x) = \frac{x-a_1}{a_2-a_1}$ and $v(y) = \frac{y-b_1}{b_2-b_1}$ and is defined locally for rectangles given by $[a_1 \le x \le a_2,\, b_1 \le x \le b_2]$. The desired contours in each slice of anatomical data are now C^1 smooth iso-curves $w = constant$ of this computed density function. In one data partitioning scheme the different data slices are handled by different workstations (servers) and the results of the iso-curves are communicated and viewed by

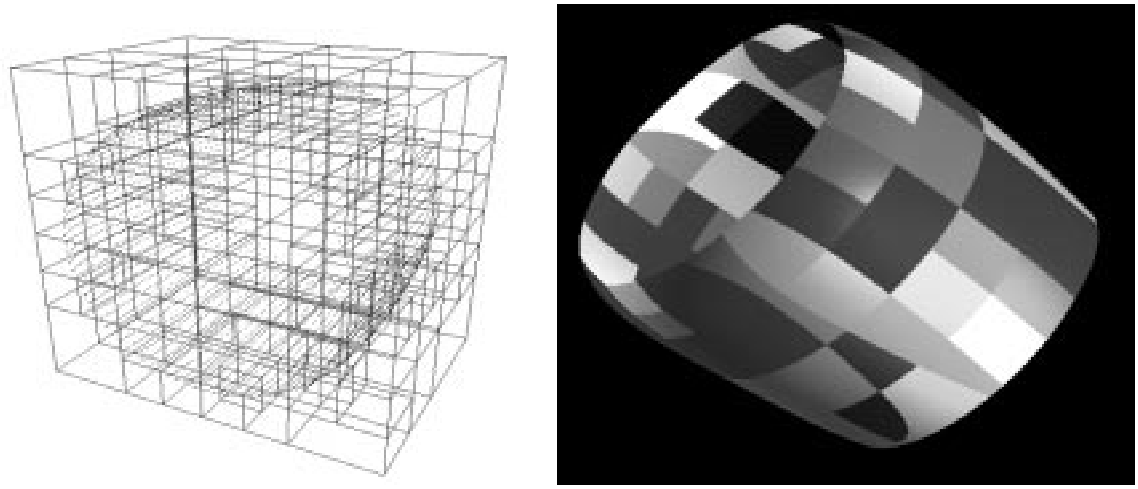
Figure 2: A Shared Visualization in SHASTRA of the Models of the Head and the Brain which are Iso-Surfaces from the Volume Cuberille MRI Data

designated display workstations (clients). Alternatively, since the piecewise trivariate interpolants are locally computable, each data slice may be further subdivided into rectangles and processed on separate workstations. See Figure 1 which shows an example of this approach used to construct a contour model of a human head from MRI data.

Another of our distributed algorithms to construct piecewise approximants works directly in object space. Here the given cuberille volume data is considered as lying in four dimensional space with the spatial coordinates x, y, z being three dimensions and the CT/MRI density values yielding the fourth dimension w. We then fit a C^1 smooth piecewise tri-cubic polynomial function $W(x,y,z) = \sum_{i=0}^{3} \sum_{j=0}^{3} \sum_{k=0}^{3} w_{ijk} B_i^3(u(x)) B_j^3(v(y)) B_k^3(w(z))$ to simultaneously interpolate and least-squares approximate this dense volume data, with a small number of polynomial pieces [6]. Here $B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$ is the degree n Bernstein-Bezier polynomial, and $u(x) = \frac{x-a_1}{a_2-a_1}$, $v(y) = \frac{y-b_1}{b_2-b_1}$ and $w(z) = \frac{z-c_1}{c_2-c_1}$ and defined locally for cuboids given by $[a_1 \le x \le a_2,\ b_1 \le x \le b_2,\ c_1 \le x \le c_2]$. The desired surface models of subparts of the anatomical data are now C^1 smooth iso-surfaces $w = \mathit{constant}$ of this computed volume density function. See Figure 2 which shows an example of this approach used to construct a volume model of a human head from MRI data.

In the first type of algorithm we construct C^1 smooth bi-cubic bivariate interpolants and approximants on networked workstations, where each server system computes the solution of a linear system with 16 variables. In the second type of algorithm we construct C^1 smooth tri-cubic trivariate interpolants and approximants on networked workstations, where each server system computes the solution of a linear system with 64 variables. Details of the derivation of the individual linear systems are given in [6].

We also model and visualize, using trivariate polynomial finite element approximations, scattered data (x, y, z, w) defined over a domain surface S in three dimensions. Note that the (x, y, z) points are all lying on a surface S in three dimensions (rather than in all of three space) and the scalar data w at each of these points arises variously, for example as temperature on the surface of a human; or a pressure distribution over the surface of a jet engine. The distributed modelling algorithms reconstruct both the domain surface S using C^1 smooth piecewise implicit tricubic polynomial surface $S(x,y,z) = \sum_{i=0}^{3} \sum_{j=0}^{3} \sum_{k=0}^{3} w_{ijk} B_i^3(u(x)) B_j^3(v(y)) B_k^3(w(z)) = 0$ patches defined over an adaptive cubical grid and approximating the (x, y, z) data. See Figure 3 where such a



(a) (b)

Figure 3: Sampled points of a jet engine outer cowl and its C^1 smooth piecewise tricubic polynomial surface approximation

piecewise smooth approximation of teh domain surface is shown together with the defining adaptive cubical grid encompassing the scattered data. The scalar function is additionally modelled by a C^1 smooth piecewise tricubic polynomial function $W(x, y, z) = \sum_{i=0}^{3} \sum_{j=0}^{3} \sum_{k=0}^{3} w_{ijk} B_i^3(u(x)) B_j^3(v(y)) B_k^3(w(z))$ approximating the scalar values w defined for the (x, y, z) points. The modelled scalar function $W(x, y, z)$ can be visualized by displaying it around the modelled domain surface and in a direction normal to the surface (at every point), or as iso-value contours of the function, surrounding or projected onto the domain surface. See Figure 4 where such visualizations are computed for sampled pressure values surrounding a jet engine. Details of this algorithms are given in [7]. See also [4] where this model recosntruction problem is solved using finite element approximations defined over a tetrahedral mesh (rather than a cubical grid).

3 Shared Workspaces

SHASTRA is a collaborative multimedia scientific manipulation environment in which experts in a cooperating group communicate and interact to solve problems. The SHASTRA environment consists of a group of interacting applications. Some applications are responsible for managing the distributed environment (the Kernel applications), others are responsible for maintaining collaborative sessions (the Session Managers), yet others provide specific communication services (the Service Applications), while yet others provide scientific design and manipulation functionality (the SHASTRA Toolkits). Service applications are special purpose tools for multimedia support – providing mechanisms of textual, graphical, audio and video rendition and communication. Different tools register with the environment at startup providing information about what kind of services they offer (Directory), and how and where they can be contacted for those services (Location). The environment provides mechanisms to create remote instances of applications and connect to them in client-server mode (Distribution). In addition, the environment provides support for a variety of multi-user interactions (Collaboration). It provides mechanisms for starting and terminating collaborative sessions, and joining or leaving them. The infrastructure is described in detail in [1].

3.1 A Collaborative Visualization Tool

POLY is a 3-D rendering and visualization tool in the SHASTRA environment. New SHASTRA toolkits use POLY as their 3D graphics interface, since it isolates 3D graphics object manipulation, rendering and visual-

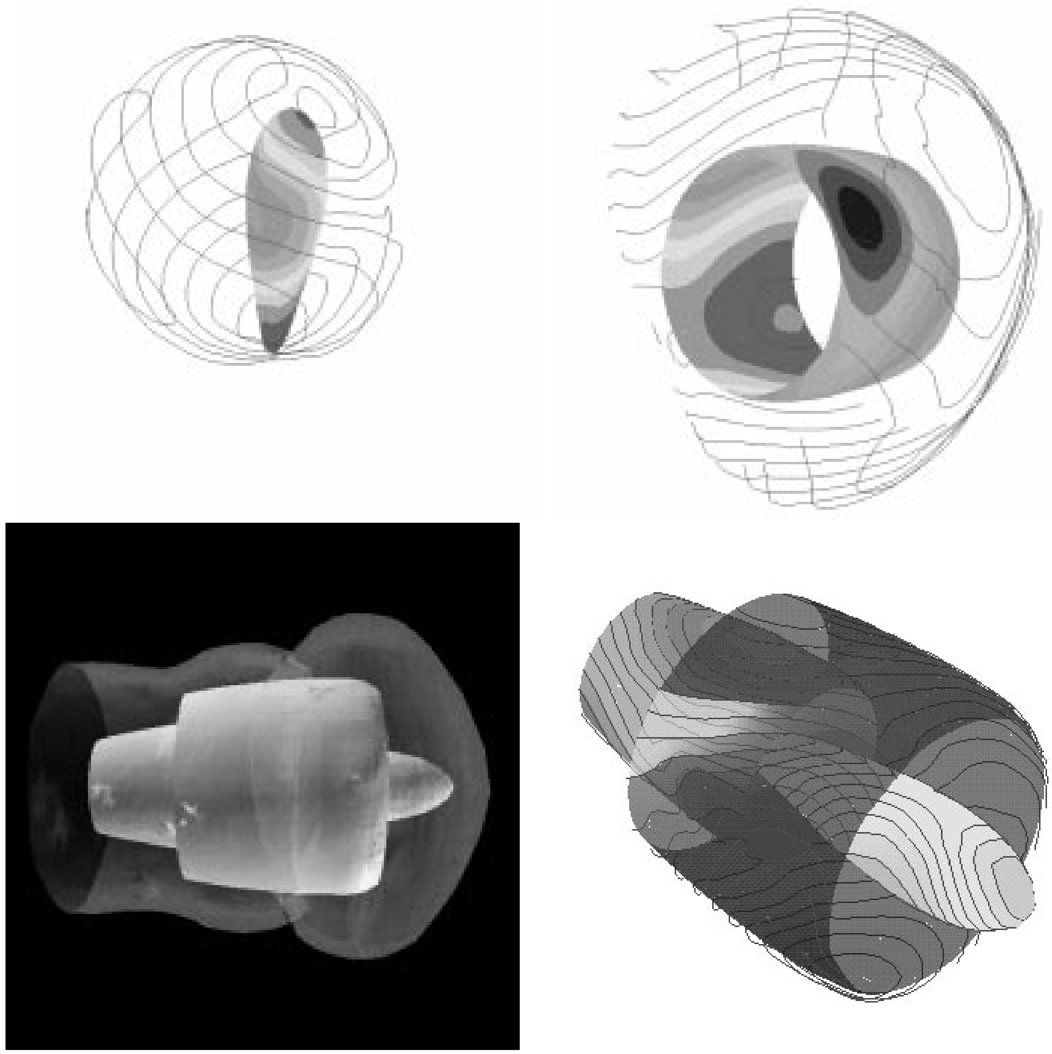


Figure 4: A Pressure Surface over a Jet Engine surface displayed around the Engine as well as Iso-Pressure Contours of the Pressure Surface projected onto the Engine surface

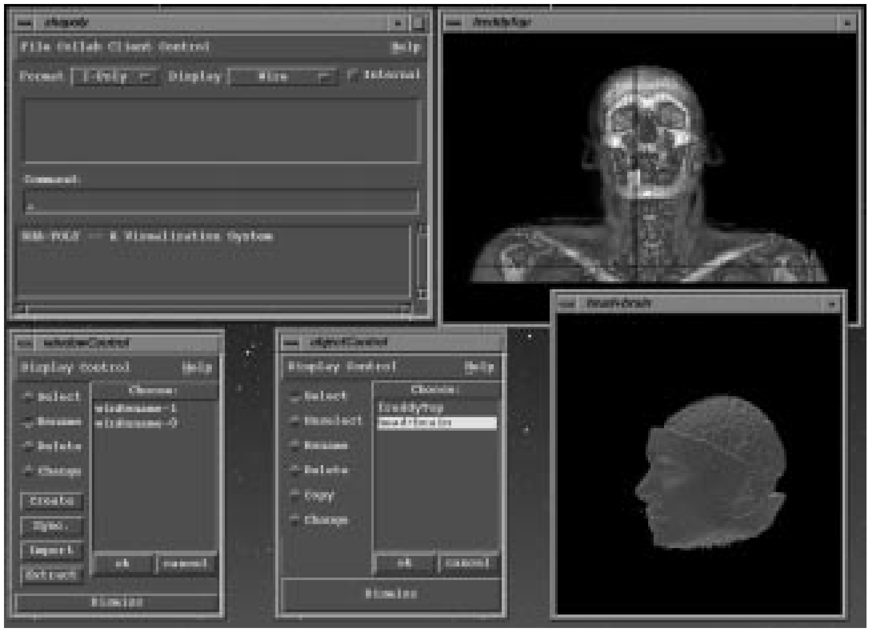


Figure 5: Using the SHASTRA Application called POLY for Collaborative Visualization

ization functionality. POLY provides a variety of mechanisms for visualization of multi-dimensional data. It understands a number of graphical object formats, which it converts to an internal form for efficient display and transport. It has a user interface that supports manipulation of graphical objects. At its network interfaces, POLY interoperates with other SHASTRA toolkits, and provides a very high level abstraction for manipulation of such data. The Motif based GUI is used to manipulate visualized objects in multiple XS graphics windows.

The user interface of the visualization system is shown in Figure 5. The top image is a volume rendering of the upper torso of Freddy. The skeletal structures are opaque and shaded, while the rest of the structures have been assigned different levels of transparency. The bottom image shows a surface rendering of a human head with a cutaway of the skull to show part of the brain surface.

The SHASTRA environment for collaborative visualization consists of a collection of instances of POLY. A collaborative session is initiated by one of the POLY users in the environment. This user becomes the group leader and specifies to the local Kernel the list of POLY users that will be invited to participate in the session, and becomes the group leader. The Kernel instantiates a Session Manager, which starts a session with the group leader as its sole participant, and then invites the specified users of concurrently executing remote POLY sessions to participate. Users that accept are incorporated into the session. Any POLY instance not in the conference can request admittance, and join. A participant can leave an ongoing session at any time. Users can be dynamically invited to join or removed from conferences by the group leader or his designees.

The hybrid computation model for conferences in SHASTRA consists of a centralized Session Manager for each session, which regulates the activity of multiple instances of POLY. Though this model suffers from problems of scale due to the centralized Session Manager, it performs well for typical group sizes. An important benefit derived from the replication is in the realm of platform heterogeneity – the application instances are responsible for dealing with particular platform idiosyncracies. In addition, since the conference consists of cooperating applications, the notion of private and shared workspace and private and shared interaction is easily supported. The centralization of the Session Manager for a collaborative session accords us the benefit of centralized state. The Session Manager serves as a repository of shared objects. This makes it easy to accommodate late joiners of sessions to come up to date quickly. It also eases the task of serialization of input actions for multi-point synchronous interaction, and constraint management for mutual consistency.

A permissions based regulatory subsystem permits control of data flow at runtime, providing a variety of

Figure 6: One Site in a Collaborative Visualization

interaction modes. Collaboration in SHASTRA can occur in the REGULATED (Turn-taking or Master-Slave) mode or in the UNREGULATED (Free Interaction) mode. In the REGULATED mode, users take turns by passing a baton. The collaboration infrastructure of SHASTRA has a two tiered permissions based regulatory subsystem used to control interaction primarily in the UNREGULATED mode. SHASTRA permissions control 'Access' to a view of the conference, local viewing controls to 'Browse' a view, rights to 'Modify' conference state, and rights to 'Copy' shared objects.

The session manager allows only one user to manipulate "hot spots" in the shared space – where there is a possibility of contention – at any particular instant. It uses the first-come-first-served paradigm to decide which user gets temporary exclusive control. The baton passing facility of the system can be used to take turns to adjust visualization parameters. Alternately, designers can use the auxiliary communication channels – like audio, video, and text by initiating SHASTRA service applications PHONE, VIDEO or TALK – to regulate access, and for arbitration[1]. All operations are performed via the (central) session manager which is responsible for keeping all sites up-to-date, so that the users have a dynamically changing and continuously updated view of the action in the shared windows.

3.2 Collaborative Volume Visualization

Every participating POLY instance creates a shared window in which all the cooperative interaction occurs. Users introduce graphics objects into the session by selecting them into the Collaboration Window. The Session Manager is responsible for providing access to the objects at all participating sites which have the Access permission, and for permitting interaction relevant to the operation at sites which have Modify permission for the collaboration. Collaborating users can twiddle visualization modes and parameters, and adjust viewing modes and direction. The system provides telepointers in the shared windows. It also provides indications of remote presence which describe the viewing location of remote users in the collaborative session. Figure 6 and 7 depict two sites in a three way collaborative visualization. The entire volume rendering of Freddy is shared by all collaborating sites – they share the data set, the viewing location, as well as visualization control parameters. The collaborators share data sets and viewing location in the other two renderings of MRI data sets of the human head. However, they use different cutaways to examine different parts of the data.

Figure 7: Another Site in a Collaborative Visualization

At one extreme, the SHASTRA implementation for Collaborative Visualization can be used by a single user to perform scientific visualizations, just like in a non-collaborative setting. Allowing other users to join the session with only Access and Browse permissions sets up the environment like an electronic blackboard to teach novice users the basics of the process. An appropriate setting of collaboration permissions and turn-taking can be used to allow hands on experience with the task. In conjunction with the audio and video communication services of SHASTRA, this becomes a powerful instructional environment. Collaborative sessions using POLY are a valuable tool for review and analysis of problem solutions. Multimedia communication facilities permit a rapid exchange of rationales for choices, interpretations of analyses and iterative improvement.

4 Conclusions

We have used the SHASTRA distributed and collaborative environment to both model and volume render large data sets efficiently. Timings for distributed ray casting and shading for volume visualization are discussed in detail in paper [3]. For example, the skull data set (512 x 512 x 113 short integers, 56.5 Mb) was volume rendered on an Indigo in 255 seconds. Using only two remote servers, the volume rendering took 195 seconds, and using only four remote IPX servers it took 165 seconds. (All numbers are whole seconds of real time the user has to wait before the final image is available. Note that this includes network latency, process swapping, and NFS access time.) Our computing environment for the distributed modelling and rendering tasks consisted of an IRIS Indigo R4000 with Elan graphics, and Sun 4/50 (Sparc IPX) server workstation, each with 32Mb RAM, linked by a 10Mbps ethernet. The data sets used are stored on remote file systems, and are accessed through NFS. Preliminary measurements of total time taken to model and visualize different volumetric data sets (made during conditions of normal network traffic) are very encouraging, and indicate that there is close to a linear speedup achieved by using multiple workstations to render very large volume data sets.

Acknowledgements: We are grateful to the of research center of Johns Hopkins University (E. Fishman, D. Hauser)

for the human cadaver data set, SUNY Stony Brook (A. Kaufman) for one of the human head data sets, and the University of North Carolina for anonymous ftp access to the skull and the other human head data set.

References

[1] V. Anupam and C. Bajaj. Collaborative Multimedia Scientific Design in SHASTRA. In *Proc. of the First ACM International Conference on Multimedia, ACM MULTIMEDIA 93*, pages 447–456. ACM Press, 1993.

[2] V. Anupam and C. Bajaj. SHASTRA - An Architecture for Development of Collaborative Applications. In *Proc. of the Second Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 155–166. IEEE Computer Society Press, 1993.

[3] V. Anupam, C. Bajaj, D. Schikore, and M. Schikore. *Distributed and Collaborative Volume Visualization. IEEE Computer*, x(x):x–y, 1994.

[4] C. Bajaj, F. Bernardini, and G. Xu. *Reconstruction of Surfaces and Surfaces-on-Surfaces from Unorganized Weighted Points.* Computer Science Technical Report, CS-94-001, Purdue University, 1994.

[5] C. Bajaj, J. Chen, and G. Xu. Free form surface design with a-patches. In *Proceedings of Graphics Interface'94*, pages x–y, Banff, Canada., 1994.

[6] C. Bajaj and G. Xu. *Trivariate Interpolants and Scientific Visulization.* Computer Science Technical Report, CSD-TR-93-18, Purdue University, 1993.

[7] C. Bajaj and G. Xu. *Reconstructing CAD models from Scans.* Computer Science Technical Report, Purdue University, 1994.

[8] D. Choi and C. Levit. Implemenation of a Distributed Graphics System. *International Journal of Supercomputing Applications*, 2:82–95, 1987.

[9] M. Gerald-Yamasaki. Cooperative Visualization of Computational Fluid Dynamics. *Computer Graphics Forum*, 12(3):497– 508, 1993.

[10] Lorensen, W., and Cline, H. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, 21:163–169, 1987.

[11] J. Wilhelms and A. Van Gelder. Octrees for Faster Isosurface Generation. *Computer Graphics*, 24:57–62, 1990.