Visualization and Tracking of Parallel CFD Simulations

Arsi Vaziri and Mark Kremenetsky[1]

Report NAS-95-004 February 1995

Numerical Aerodynamic Systems Division
NASA Ames Research Center
Moffett Field, CA 94035-1000
e-mail: vaziri@nas.nasa.gov
e-mail: Kremenet@nas.nasa.gov

KEYWORDS distributed visualization, parallel CFD, simulation steering, co-processing, AVS.

ABSTRACT

We describe a system for interactive visualization and tracking of a 3-D unsteady computational fluid dynamics (CFD) simulation on a parallel computer. CM/AVS, a distributed, parallel implementation of a visualization environment (AVS) runs on the CM-5 parallel supercomputer. A CFD solver is run as a CM/AVS module on the CM-5. Data communication between the solver, other parallel visualization modules, and a graphics workstation, which is running AVS, are handled by CM/AVS. Partitioning of the visualization task, between CM-5 and the workstation, can be done interactively in the visual programming environment provided by AVS. Flow solver parameters can also be altered by programmable interactive widgets.

This system partially removes the requirement of storing large solution files at frequent time steps, a characteristic of the traditional "simulate-> store->visualize" post-processing approach.

---

1. This author is an employee of IBM Corporation.

1.0 INTRODUCTION

The ultimate goal of computational aerosciences is the optimal design of aerodynamic bodies; it is accomplished through state of the art numerical modelling and use of modern software development techniques. We can outline three requirements of any design procedure related to aeronautics:

1) The capability to predict the flow past aerodynamic bodies of complex geometrical shapes in various flight regimes.

2) The capability to provide an interactive design control and calculations to allow immediate improvement.

3) An automatic design technology based on evaluation of aerodynamic performance.

The direction for efficient improvement for item 1) seems to be quite clear. Recent advances in computer technology and numerical algorithms have encouraged the use of sophisticated and accurate fluid dynamics models. These are based on Euler and Navier Stokes equations coupled with complicated nonlinear phenomena such as thermal, turbulent, and chemical effects within aerodynamic design optimization.

A new family of Massively Parallel Processors (MPP; e. g., TMC CM-5, Intel Paragon, IBM SP-2, etc.) provide enough computational resources to resolve large, unsteady, 3-D CFD problems containing more than one million discrete elements in several minutes.

By contrast, the trend for development of control components of an optimization technology (particularly graphics methods) is still unclear.

In a post-computation processing mode, data are stored in external file storage devices and are analyzed or visualized on graphics systems at a later time. For time-dependent flows, results are seldom stored at every time step due to capacity limitations imposed by the data storage facility.

Efficient functionality of the design optimization cycle requires rapid interaction with time-varying and fixed data, including those from the flow predictions. A batch processing environment does not seem to provide this level of interaction and feedback. A steady increase in performance of modern supercomputers and workstations has substantially improved the prospect of run-time visualization of simulation results.

In order to exploit this newly emerged paradigm of interactive computing and visualization, without an overwhelming reliance on special-purpose software, we have used CM/AVS (Thinking Machines Corp., 1993), a parallel implementation of a commercially available visualization software AVS (Advanced Visual Systems, 1992) on a parallel supercomputer.

1.1 Related Work

An early high-performance, interactive computation/visualization demonstration system, dubbed a "numerical laboratory", was built from highly specialized hardware and software components at the Los Alamos National Laboratory (Winkler, et. al., 1987). Due to its specialized nature, this system was not suitable for day to day, routine computations; its evolution with advancing technology also became difficult.

Vendors of some emerging high-performance super workstations also implemented demonstrations of concurrent computations and visualization in order to showcase their new products. Depending on the implementation, model computations could be distributed on a supercomputer or performed locally on a single or multi-processor shared memory workstation. Model computations were usually small, 2-D or trivially 3-D, demonstration problems that were chosen in order to produce a rapid compute/visualize demonstration cycle.

The introduction of Massively Parallel Processors (MPP) and networks of workstations has provided enough memory to distribute large simulations over the computational nodes. Collection and visualization of data from the MPP nodes has thus become a serious and challenging problem.

There are two tasks at hand when it comes to visualizing data that are generated on MPP nodes. One is collection and communication of data from the computational nodes; the second is the ability to partition the visualization processing between the MPP and the workstation. The latter is a more complex problem since it requires a means of redistributing the data to parallel visualization programs that are running on the same or a different set of MPP nodes. Several systems have addressed some aspects of these two tasks in recent years. A complete, general-purpose system, has yet to emerge.

A system for simulation steering on an array of high performance workstations and an MPP have been reported (Woodward, 1993). It performs concurrent computation and visualization of 2-D and axi-symmetric fluid flow problems. A locally developed, custom visualization software was utilized to allow geometry modifications while the simulation was running.

In another approach to visualization of MPP data in real-time, PORTAL (Rowlan and Wightman, 1994), a library of data gathering and communication has been developed at Argonne National Laboratory. The focus of PORTAL is visualization of distributed data during run-time. It was originally developed to transfer data in a global climate simulation model,

running on IBM SP-1, to an AVS module on a workstation. Data is sent directly to an AVS process over Unix sockets. Some level of control over the simulation and simulation steering may be possible within an application using PORTAL. However, PORTAL does not provide any means of redistributing the data on the MPP for visualization processing.

pV3 (Haimes, 1994), is a distributed visualization software for unsteady CFD data developed at MIT. The current version of pV3 uses PVM and can run on several platforms. A version of pV3 runs on IBM SP-2 at NASA Ames. In this implementation, PVM is used as the transport mechanism for data collection and communication between the simulation, a master process, and the workstation. A unique feature of pV3 is an implementation of extracts (Globus, 1992) to optimize data transfer between the visualization server and computation clients while preserving rendering flexibility. It does not allow redistributing simulation data over the nodes.

CM/AVS is a parallel modular visualization environment (MVE) based on AVS and runs on the CM-5 parallel supercomputer. CM/AVS interactively communicates with AVS running on the workstation. CM/AVS is a first generation parallel MVE; it addresses the two tasks of data communication and parallel visualization quite nicely. Parallel visualization tasks can be written in an AVS-like module development environment and added to the library of standard AVS, modules. A module can then be dragged into the workspace as a remote module and visually connected to a network of other workstation-based modules. When processing begins, the remote modules run on the remote parallel computer (CM-5) and send their data through Unix sockets to the AVS modules running on the workstation. Since a compute-intensive visualization module can be written in parallel to operate on the remote computer, it is possible to have parallel and serial versions of the same visualization task on the respective machines. This provides a flexible means of partitioning the visualizing task between the workstation and the MPP.

2.0 THE CURRENT ENVIRONMENT

The environment we have used to track parallel CFD simulations is based on CM/AVS running on the CM-5 and connected over a network to a serial version AVS, running on an SGI workstation. The system is composed of general purpose off-the-shelf hardware and software components provided by vendors. Except for writing the CFD solver into a CM/AVS module (not a very difficult task), existing facilities provide a quick visualization capability that can be brought up in a couple of days.

It is desirable to have a system of this kind run on several platforms. However, the current state of the technology is such that these off-the-

shelf components are only available for specific platforms; e. g., CM/AVS only runs on a CM-5. As a result, cross-platform utilization is curtailed; a condition that needs to be alleviated by development of next generation platform-independent distributed/parallel visualization systems.

2.1 Hardware Environment

The CM-5 Connection Machine system installed at NAS System Division, NASA Ames Research Center consists of 128 SPARC-based computational nodes with custom vector units, a Scalable Disk Array of 48 disk drives, 4 Control Processors, an Input/Output Control Processor, and HiPPI channels.

The CM-5 architecture uses three independent networks to connect all components of the system, (Thinking Machines Corp., 1992). It has a scalable I/O system which simultaneously supports Unix-compatible operations and scalable high-performance, parallel I/O operations. HiPPI is supported on the system through a directly connected HiPPI subsystem.

2.2 Software Environment

The CM-5 operating system is based on SunSoft's Solaris operating system with extensions to support parallel, timeshared interactive processes, reconfiguration of nodes into various-sized partitions, and parallel I/O devices.

There are two principal ways to create a scalable parallel application on the CM-5. Using a data parallel language such as CM Fortran, or writing a message passing program.

The visualization library CM/AVS is designed to integrate very large parallel data sets produced by CM-5 computations into a standard serial visualization environment. CM/AVS extends the Application Visualization System (AVS) to operate with parallel data on the CM-5 supercomputer. Based on the data parallel languages C* and CM Fortran, it introduces a parallel field which maps naturally into these languages.

CM/AVS is implemented as a set of libraries which are linked in before the standard AVS libraries. These libraries provide access to the parallel communication system, replace some AVS functions, and a add handful of new CM/AVS functions. The system also includes a library of parallel AVS modules which can be used to process fields on the CM-5.

In a typical use of CM/AVS, an unmodified AVS kernel is run on a workstation. Some modules are also run on this workstation, while others are compiled with CM/AVS and run remotely on the CM-5. These remote

CM-5 modules behave identically to any other remote AVS module; the user is therefor free to connect serial and parallel modules. In the current NAS applications, AVS is run on an SGI IRIS Crimson workstation, and communication with the CM-5 uses an Ethernet connection.

2.2.1Parallel Fields

A data parallel language such as C* or CM Fortran exploits parallelism by applying the same operation to many data elements on many processors simultaneously. Data elements are aggregated into parallel variables in C* and parallel arrays in CM Fortran (for convenience, we will use "parallel variables" to refer to both of these aggregates). Parallel variables are multi-dimensional Cartesian grids, so there is a very natural mapping between these data structures and AVS fields.

One bit in the AVSfield structure is set to indicate that the field is parallel. The functions in the CM/AVS library examine this bit and dispatch to the appropriate serial or parallel routines. The data member of a parallel field is simply a pointer to a parallel variable on the CM-5. If the parallel field is curvilinear or irregular, another parallel variable will hold its coordinate mapping array. To access pointers to these parallel variables, there are several new CM/AVS functions. For example, `CMAVSfield_data_get` takes an AVSfield structure as input and returns a pointer to the parallel variable on the CM-5.

In CM Fortran, pointers are not available. So we construct a serial array which contains all the information needed to point to the parallel array. This serial array is then passed to a second Fortran function, which can operate as if it had been passed a parallel array.

2.2.2Communication

To transfer fields, CM/AVS depends on the direct module-to-module data transfer mechanism of AVS. It contains a new parallel communication layer that manages the conversion between parallel fields and those expected by standard AVS modules. This layer also uses the most efficient transport available to send data from one module to another. As in any other AVS module, the choice of transport is completely hidden from the module writer.

When CM/AVS modules are running in the same process a parallel field can then be transferred from one module to another by a simple pointer copy. If two modules are running in different processes on the same CM-5 partition, they can be connected by a socket that uses the CM-5 Data Network. Since the source and destination processors are the same for each piece of data, this amounts to copying data through the kernel and

is quite fast. All other communication methods are accessed through parallel sockets.

CM/AVS can also take advantage of HiPPI networks. Finally, if another network connection, such as an Ethernet, is available between the workstation and the CM-5, CM/AVS will use parallel sockets to transfer data over this connection.

3.0 RESULTS: PARALLEL CFD SIMULATION

We are interested in solving a class of 3-D unsteady flow problems for which simulations are fast enough for interactive visualization. As long as the solution continues to change at a rate of one frame every few seconds, visualization tracking has proven to be quite useful. We have been able to observe the evolving flow field and interact with the solver for further examination of a desired flow feature.

Since computation and visualization is performed on the fly, there is no need to store the solution data at a large number of time steps. There may however be cases in which future access to the entire time history of the flow field is desired. To accommodate such cases, snapshots of the solution data may be stored occasionally. For problem sizes we have experimented with, the cost of computations is small enough so that simulation results need only be stored on the order of once every 500 iteration time steps. Interactive simulation is restarted from the saved data in order to compute the solution at subsequent time steps.

Clearly, there is a class of problems that are too large to be dealt with effectively in the environment we have described here. The problem arises from the current limitations in computer and networking technologies. It is also difficult at times to adequately process transient data in a data flow visualization environment (Mayer., and Tabatabai, 1993).

3.1 Simulation Tracking

The software used for simulation tracking are user CFD applications that have been written and run on the CM-5. Simulation and visualization are distributed between the CM-5 and a high-performance graphics workstation from Silicon Graphics. In this interactive simulation tracking system the user can stop the computations at any time and examine the evolving numerical flow visualization in more detail and then continue or restart the process.

An operational CFD solver which normally runs in batch is the starting point to a CM/AVS CFD simulation tracking module. The process of writing a CM/AVS module is easy and similar to writing AVS modules.

The difference is in defining AVS fields as parallel and loading appropriate parallel data into these fields. AVS provides a template for a semi-automatic module generation.

We have chosen two parallel CFD solvers which were originally written in the data parallel syntax of CM Fortran. It was convenient to also write the additional code in the same language as the solver. Parallel visualization modules, other than the solver, available under CM/AVS are written in either parallel C (C*) or CM Fortran. More detail about the process of writing CM/AVS modules can be found in Vaziri, et. al, 1994 and the CM/AVS Users Manual.

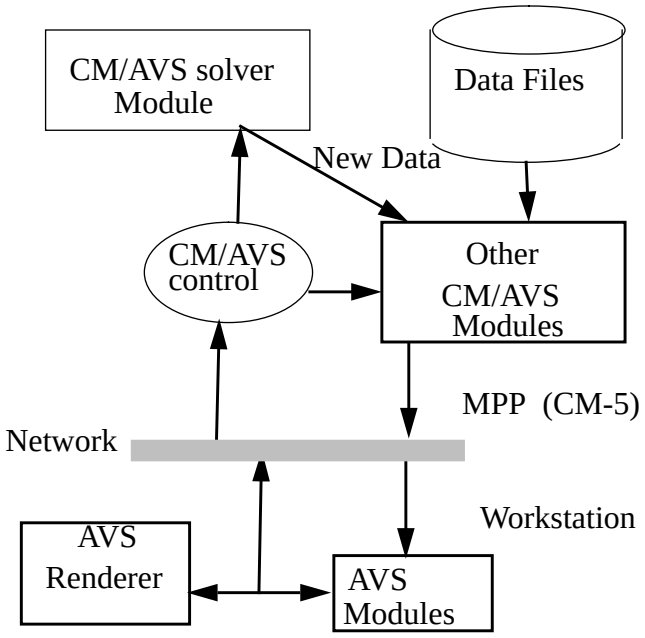


FIGURE 1. Schematic of the data flow in the system.

Figure 1 is a schematic of how data flows between the main components of the system. Starting from the top left, the solver module internally advances the computational time step, applies the boundary conditions and computes a new solution corresponding to the new time. This solution is loaded into CM/AVS parallel fields and becomes available at the module output port, represented by "New Data" in Figure 1. Parallel visualization modules on the CM-5 or traditional serial AVS modules on the workstation, connected to the output port of the solver module, then receive the new data for processing. Conversion from parallel to serial data, for going over the network, is handled by CM/AVS.

Interactive dials and widgets to control parameters within the solver are easily programmed and become visible on the display. Parameter values

are changed by mouse input or typein and are sent back to the solver. Changes take effect with the next time step of the solution.

So far we have experimented only with time step changes for an interactive control. Even such a narrow control thread helped us to find the right balance between the convergence rate, stability, and the specific time scale resolution. More complicated applications, such as an optimal design procedure, should offer much wider range of control parameters for interactive and automatic variations coupled with visual real time steering.

In addition to solver modules, we also have modules for post-processing simulation data stored on the Scalable Disk Array as data files, also indicated in Figure 1. These modules read parallel CFD data stored in PLOT3D format (Walatka, et. al, 1993), or a Saturn Ring simulation data (Cuzzi, et. al, 1993).

3.2 The cm3d solver

One of the two CFD solvers we have used for our computations is "cm3d". Cm3d is a compressible Navier-Stokes solver for use on multiple overlapping three-dimensional structured curvilinear grids. It uses centered differences and non-linear artificial dissipation on the right-hand-side and a factored implicit scheme for the left-hand-side. The factored implicit scheme can solve either scalar tridiagonal, scalar pentadiagonal, or block-tridiagonal systems. This code was used to compute unsteady three-dimensional low Reynolds number flow past a tapered cylinder.

The spanwise variation in natural shedding frequency results in interesting three-dimensional flow phenomena. The computed data are very similar to experimental results. The computations highlight the capability of CM-5 for numerical simulation of three-dimensional unsteady flow fields.

The cm3d solver has been made into a CM/AVS module and is used in the simulation tracking experiments (Vaziri, et. al, 1994). In our 3-D, unsteady flow simulation we are able to start, stop and restart the simulation. Through an AVS file browser, we can chose the number of time-steps it is to run and the input data file to be used. Other simulation parameters, such as the time interval to send data to CM/AVS, or the simulation time step can also be changed on the fly through an interactive dial. Figure 2 shows the complicated structure of a density isosurface from a simulation of the unsteady flow past a tapered cylinder.

Our test problem uses a single grid zone consisting of 131,072 points (64x64x32). We have also experimented with a 4 zone 909,312 point jet problem. At this stage, we are only able to visualize a single grid of a

multi-zone calculation in progress due inherent limitations imposed by the data structure within AVS. We have tried wiring multiple geometries to the renderer (the geometry viewer) but thetapered.sgi work around is not of sufficient generality to be useful. This problem has been rectified in the next release, AVS-6.

Effective, run-time visualization processing requires fast data transfer times. We have found that if the size of the data to be transferred is subsampled at its source (on the CM-5) the data update rate for the visualization can be reduced to make the interactive processing effective. In order to get a visualization update rate of about one frame every 2 seconds, we have had to hard-wire a down-sizing routine inside cm3d solver module.

The subsampling requirement should not be necessary if we were able to use faster communications speed through a HiPPI channel. Optimized data transfer scenarios, such as use of extracts or data compression techniques, also address this problem but have not been implemented in this system.

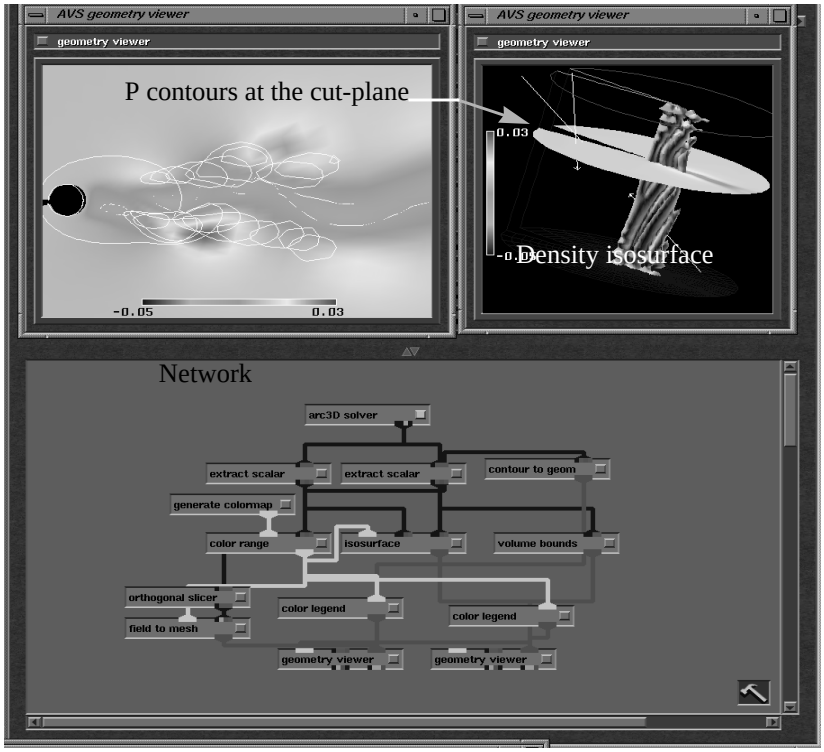


FIGURE 2. Two views of an unsteady flow past a tapered cylinder in a simulation tracking experiment.

3.3 The Blast Wave Simulation

The second application of interest was the numerical simulations of blast waves in a transonic transient laminar flow. The kernel of this application is an unfactored iterative parallel Navier-Stokes solver (Jordan, Kremenetsky and Richardson 1992). The discretization model was based on Beam-Warming implicit numerical scheme with the first order accuracy in time and the second order accuracy in space. The resulting system of linear algebraic equations uses the parallel version of Bi-CGstab algorithm that provides a reasonable convergence for antisymmetric systems.

The initial configuration of blast waves problem consists of a high pressure and density region at the center of a cubic cell of a periodic lattice with low pressure and density elsewhere.

Initially, the high pressure volume begins to expand in the radial direction as the classical shock waves. At the same time, the expansion waves move to fill the void at the center of the cubic cell. When the expanding flow reaches the boundaries, it collides with its periodic images from other cells; thus creating a complex structure of interfering nonlinear waves. These processes create a nonlinear damped periodic system with energy being dissipated in time. Finally, the system will come to an equilibrium and steady state.

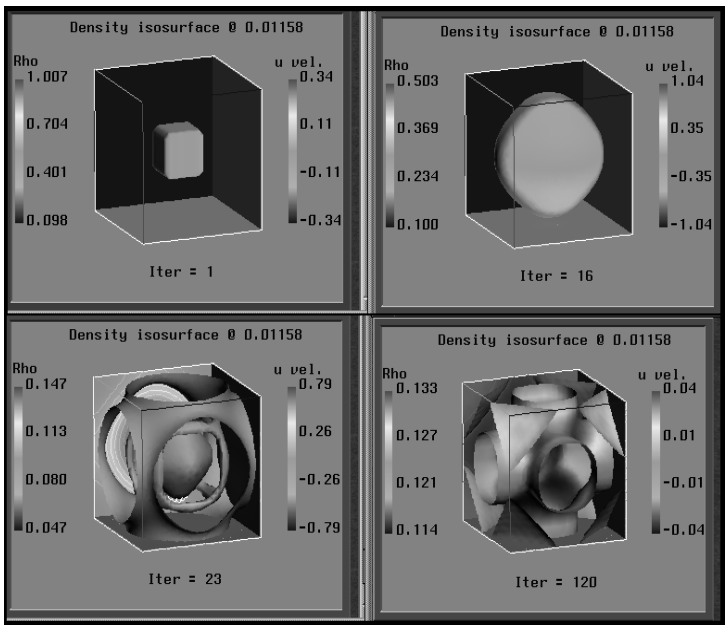


FIGURE 3. Time history of Density isosurface @ 0.1158 in the blast wave simulation.

For a domain of dimensions 32 x 32 x 32 (32,768 grid points) computations for this problem are fast and the visualization can be viewed in real time. In addition, isosurface visualization of scalar variables is an effective means of visualization for this problem. An isosurface of variables such as pressure, density, and individual components of the velocity could be viewed as they evolve with time.

4.0 SUMMARY

We have implemented an interactive parallel visualization tool for simultaneous visualization of computations in progress. Case studies are presented in CFD for which our simulation tracking approach is practical.

These simulations are generally characterized by 8 independent and dependent flow variables over structured grids of fewer than one million grid nodes. In addition to the problem domain size, other factors such as processing power and network speeds influence the performance and applicability of our system. Examples of tracking flow past a tapered cylinder and the simulation of blast waves in a transonic transient laminar flow in which these factors are examined were given.

A class of non-trivial, 3-D, parallel CFD computations, with significant grid sizes, have been identified which can be interactively run with this tool.

3-D parallel CFD simulations that can be effectively run under this system are limited perhaps to about one million grid points. Faster networks, such as HiPPI, are needed to effectively transmit the data between heterogenous computing platforms and test effectiveness of larger size problems. We must currently use data subsampling techniques to achieve data transfer speeds that are sufficiently fast for interactive processing.

5.0 CONCLUSIONS

CM/AVS provides user-transparent data collection and communication tasks from the MPP nodes. It is also possible to develop new parallel visualization techniques and test scenarios for tracking and steering of simulations under a modular interactive visualization environment, such as CM/AVS.

For a limited problem size domain, in time-dependent CFD simulations, simultaneous recomputing/visualization is a viable alternative to storage of large data sets and the associated cost of post processing.

This system provides the testbed for future "simulation steering" experiments in which one can interactively and visually modify the internal parameters of the simulation, including geometry or the boundary conditions, and then continue (or restart) the computations.

6.0 ACKNOWLEDGMENTS

We have benefited from many discussions with Dennis Jesperson and Matt Fitzgibbon during the conduct of this research. Review comments from Al Globus and Michael Gerald-Yamasaki have contributed to clarity of the final version. This work was partly supported through NASA contract NAS2-13537.

7.0 REFERENCES

[1] Advanced Visual Systems. 1992. AVS User's Guide, Version 4.

[2] Cuzzi, J., C. Levit, et al. 1993. "Simulation of a Moonlet Belt Under the Influence of Multiple Perturbers". Proc. 1993 Annual Meeting of the American Astronomical Society Division of Planetary Sciences, Boulder Colorado.

[3] Globus, A. 1992. "A Software Model for Visualization of Time Dependent 3-D CFD Results." Report RNR-92-031, NAS Applied Research Branch, Moffett Field, CA, Nov. 1992. Available at URL (http://www.nas.nasa.gov/NAS/reports/RNRreports/aglobus/RNR-92-031/alUnsteady.html).

[4] Haimes, R., 1994. "pV3: A Distributed System for Large-Scale Unsteady CFD Visualization." AIAA Paper 94-0321. 32nd AIAA Aerospace Sciences Meeting. Reno, Nevada.

[5] Jesperson, D. C., and C. Levit. 1991. "Numerical Simulation of Flow Past a Tapered Cylinder." AIAA paper 91-0751. AIAA 29th Aerospace Sciences Meeting.Reno, Nevada.

[6] Jordan, K. E., M. D. Kremenetsky, and J. L. Richardson. 1992. "Visualizing Navier-Stokes Solutions Using the Connection Machine." Proc. Seventh IMACS International Conference on Computer Methods for PDE, New Jersey.

[7] Mayer, H. F., and B. Tabatabai. 1993. "Visualizing Results of Transient Flow Simulations." Visualization '93 Proceedings, San Jose, CA.

[8] Rowlan, J. S. and B. T. Wightman. 1994."PORTAL: A Communication library for run-time visualization of distributed, asynchronous data." Proceedings of the 1994 Scalable High-Performance ComputingConference,Knoxville.May23-25.

[9] Thinking Machines Corporation. 1992. CM-5 Technical Summary

[10]Thinking Machines Corporation. 1993.CM/AVS Users Guide.

[11]Vaziri, A., M. Kremenetsky, M. Fitzgibbon, and C. Levit. 1994. "Experiences with CM/AVS to Visualize and compute simulation data on the CM-5." 1994 AVS Users Conference Proceedings, May 2-4, Boston, MA.Also as Report RNR-94-005 available at URL: (http://www.nas.nasa.gov/NAS/reports/RNRreports/avaziri/RNR-94-005/RNR-94-005.html)

[12]Walatka, P. P., P. G. Buning, L. Pierce, and P. A. Elson. 1993. PLOT3D User's Manual, NASA Ames Research Center, Moffett Field, CA, 94035.

[13]Winkler, R. B., J.W. Chalmers, S. W. Hodson, P. R. Woodward and N. J. Zabusky. 1987. "A numerical laboratory." Phys. Today 40(10).

[14]Woodward, P. R. 1993. "Interactive Scientific Visualization of Fluid Flow."Computer, Vol. 26, No. 10.