EXPERIENCES WITH CM/AVS TO VISUALIZE AND COMPUTE SIMULATION DATA ON THE CM-5(1)

Arsi Vaziri(2), Mark Kremenetsky(3), Matt Fitzgibbon(4), and Creon Levit(2)

Report RNR-94-005 March 1994

NASA Ames Research Center
Mail Stop T27A-1
Moffett Field, CA 94035-1000
{vaziri, kremenet, fitzgibb, creon}@nas.nasa.gov

ABSTRACT

We have used AVS and its parallel implementation, CM/AVS, to create a distributed environment between a Silicon Graphics IRIS Crimson workstation and a CM-5 Connection Machine parallel supercomputer from Thinking Machines Corporation. This heterogeneous distributed system is used to visualize data generated from simulations on the CM-5. CM/AVS modules have been developed to read, process, and visualize parallel array data stored on the CM-5 Scalable Disk Array (SDA). The data are cast into CM/AVS fields, sent through the local area network, and fed to local AVS modules running on the workstation. In addition, we are developing a CM/AVS CFD module based on a 3-D time dependent flow solver. A distributed network of CM/AVS and AVS modules is used to concurrently visualize the time varying flow field being computed on the CM-5. We will show early results of using this distributed parallel computation/visualization environment for some current application codes. Examples of post processing CFD data and concurrent computation/visualization of time-varying 3-D flow fields are given.

---

(1). To appear in Proceedings AVS'94, Boston, MA, May 1994.

(2). NAS Applied Research Branch, NASA Ames Research Center, Moffett Field, CA 94035-1000
(3). Thinking Machines Corp., NASA Ames Research Center, Moffett Field, CA 94035-1000
(4). Thinking Machines Corp., 245 First Street, Cambridge, MA 02142-1264

EXPERIENCES WITH CM/AVS TO VISUALIZE AND COMPUTE SIMULATION DATA ON THE CM-5

Arsi Vaziri(1), Mark Kremenetsky(2), Matt Fitzgibbon(3), and Creon Levit(1)

NASA Ames Research Center
Mail Stop T27A-1
Moffett Field, CA 94035-1000
{vaziri, kremenet, fitzgibb, creon}@nas.nasa.gov

ABSTRACT

We have used AVS and its parallel implementation, CM/AVS, to create a distributed environment between a Silicon Graphics IRIS Crimson workstation and a CM-5 Connection Machine parallel supercomputer from Thinking Machines Corporation. This heterogeneous distributed system is used to visualize data generated from simulations on the CM-5. CM/AVS modules have been developed to read, process, and visualize parallel array data stored on the CM-5 Scalable Disk Array (SDA). The data are cast into CM/AVS fields, sent through the local area network, and fed to local AVS modules running on the workstation. In addition, we are developing a CM/AVS CFD module based on a 3-D time dependent flow solver. A distributed network of CM/AVS and AVS modules is used to concurrently visualize the time varying flow field being computed on the CM-5. We will show early results of using this distributed parallel computation/visualization environment for some current application codes. Examples of post processing CFD data and concurrent computation/visualization of time-varying 3-D flow fields are given.

1. INTRODUCTION

An unprecedented increase in computational capabilities has enabled scientists to compute detailed numerical flow fields. Simulations have progressed from 2-D, low-resolution, steady-state to 3-D, multi-zone, high resolution, unsteady flow fields. Today, the large, unsteady CFD simulations at NASA's Numerical Aerodynamic Simulation (NAS)(4) supercomputer facility contain millions of grid points and are run for thousands of time steps. These simulations produce many gigabytes of data[8][16]. Interactive visualization of such data in their entirety has been a formidable task for visualization systems to date. Current CFD visualization systems generally deal with large datasets in a post-processing mode[10]. Systems with user interaction are usually required to make a trade-off between the number of time slices or the spatial resolution of the data in their representations.

Various models for post-computation visualization of data have been investigated at NAS, and this is still our dominant mode of CFD data analysis and visualization. The extensive visualization facilities and systems for analysis of computational fluid flow data range from virtual reality based systems[2], to cooperative visualization between multiple workstations and a supercomputer[6]. Flow analysis toolkits FAST[17] and UFAT[11] are used extensively. PLOT3D[18] has been in development and extensive use at NASA Ames for a number of years. It has contributed to and influenced the development of other visualization tools mentioned above.

---

(1). NAS Applied Research Branch, NASA Ames Research Center, Moffett Field, CA 94035-1000

(2). Thinking Machines Corp., NASA Ames Research Center, Moffett Field, CA 94035-1000

(3). Thinking Machines Corp., 245 First Street, Cambridge, MA 02142-1264

(4). NASA's Numerical Aerodynamic Simulation (NAS) facility is operated by the NAS System Division at Ames Research Center, Moffett Field, California. One of the major objectives of NAS is to act as a pathfinder in the use of advanced computer systems to solve computational aerosciences problems. To this end, NAS supercomputer center currently includes a 128 processor Thinking Machines CM-5, a128 node Intel iPSC/860, a 208 node Intel Paragon, two 16 processor Cray C-90s, a host of mid-range CONVEX systems, and several hundred Silicon Graphics workstations.The NAS supercomputer system provides about fifty GigaFLOPS of computational power and links approximately 1400 industry, university, and government users in a high-speed network.

While this enormous volume of data may be daunting, the levels of detail achieved in the CFD simulations have provided intriguing opportunities. These highly detailed simulations have generated interest in integrated, interactive processing of CFD results with concurrent data from laboratory experiments. Using interactive scientific visualization techniques, scientists want to examine experimental and computational datasets side by side. This may reveal hidden complexities in the two flow fields which may not be apparent with traditional post-processing techniques. Additionally, recent performance efficiencies achieved by parallel CFD codes[14] could make parallel computers a viable alternative for this class of interactive fluid dynamics experiment/computation simulations.

The general motivation for the work presented in this paper is to experiment with an interactive visualization programming environment for parallel computing. The specific motivation is to provide a means of visualizing CM-5 simulation data. The primary focus is on data from computational fluid dynamic (CFD) simulations. Visualization of these data is significant due to a serious emergence of parallel computing as a promising computational resource for CFD simulations at NASA Ames Research Center.

We discuss processing and visualization of 3-D, time-varying CFD data on a distributed system under the control of AVS and its parallel extension CM/AVS. Parallel modules written with CM/AVS are executed on the CM-5 Connection Machine from Thinking Machines. These CM-5 modules are invoked as remote modules from the module tools panel of AVS, which is run on a Silicon Graphics Iris Crimson workstation. Currently, the CM-5 and the SGI are connected by Ethernet, and this will be upgraded to HiPPI (High Performance Parallel Interface) when it becomes available during the next few months.

As a visualization environment, CM/AVS supports interactive post-processing capabilities. However, it is also a distributed parallel computing environment where various scenarios for partitioning of the computations and visualization on the CM-5 and the workstation could be considered. In this distributed system we have experimented with the following two modes of processing:

1. Post-processing of parallel arrays stored on CM-5.
2. Tracking a 3-D, unsteady CFD simulation running on CM-5.

Simulation steering, where one would interact with an on-going simulation, can be built on top of our system as it evolves in the future. Recently, visualization steering for a 2-D compressible flow over obstacles has been reported by Woodward[19].

In our tracking application, partitioning decisions were usually dictated by the speed of the network connection between the CM-5 and local workstations, forcing us to choose the least amount of data transfer across the network. We are also restricted to performing rendering on the workstation since there is currently no supported graphics rendering software available on the CM-5.

Although we used only two platforms in this distributed system, there are no difficulties in including additional AVS platforms. At times, for example, a CONVEX platform running CONVEX/AVS was also included in our distributed system. This configuration was used to test and debug remote module execution.

A brief description of the massively parallel CM-5 supercomputer is presented in Section 2. Section 3 provides details of the current version CM/AVS, as well as a description of parallel arrays and their processing as CM/AVS fields. In Section 4 we describe the modules that have been developed, followed by conclusions and discussion of future directions in Section 5.

2. THE ENVIRONMENT

The CM-5 Connection Machine system was installed at NAS System Division, NASA Ames Research Center in 1993. Detailed descriptions of NAS computing environment and networks have been given elsewhere[13]. However, a brief description of the hardware and software characteristics of the CM-5 at NAS will be presented below as background.

2.1 The Hardware Environment

The CM-5 system at NAS consists of 128 computational nodes with vector units (VU), Scalable Disk Array (SDA) of 48 disk drives, 4 Control Processors (CP), an Input/Output Control Processor (IOCP), and HiPPI channels. The CM-5 contains three kinds of processing nodes:

- Computational processor nodes (PN)
- Control processor nodes (CP)
- I/O control processor nodes (IOCP)

All three are based on 32MHz SPARC microprocessors. In the case of the PNs, each SPARC is supplemented by four Vector Units (VU), which execute vector-based arithmetic operations and provide a high-bandwidth path to the memory on each node.

The CM-5 architecture uses three independent networks to connect all components of the system[3]:

- Data Network that is the primary data communication network for point-to-point communication.
- Control Network that supports cooperative global operations including broadcast, reductions, scans, and synchronization.
- Diagnostic Network for testing for hardware failures.

Table 1 summarizes some performance characteristics of the CM-5 node.

TABLE 1. Performance characteristics of the CM-5

Peak 64 bit floating point arithmetic	128 arith.instr. Mops/node
Memory range/node	32 Mb
Minimum latency	0.275 micro sec
Hardware Bandwidth	20 MB/s
Application Latency	< 3.0 micro sec

The CM-5 has a scalable I/O system which simultaneously supports Unix-compatible operations and scalable high-performance, parallel I/O operations. There is also a complete family of I/O devices such as: LAN Interface, Ethernet, FDDI Interface, Scalable Disk Array (SDA), and HiPPI. An important member of this family, SDA, is a very flexible RAID system. The SDA is extensively scalable in both capacity and bandwidth (e.g. the SDA system at NASA Ames has 25 GB capacity and provides about 75 MB/s bandwidth). HiPPI is supported on the system through a directly connected HiPPI subsystem.

2.2 The Software Environment

The CM-5 operating system (CMOST) is based on SunSoft's Solaris (SunOS) operating system and is fully UNIX compatible. Thinking Machines extensions support parallel, timeshared interactive processes, reconfiguration of nodes into various-sized partitions, and parallel I/O devices.

There are three ways to create a scalable parallel application on the CM-5:

1. Write the application in a “data parallel language”. Available languages are: C*, CM Fortran, and *Lisp.
2. Write a message-passing program where all communication for transferring data between nodes is under explicit control of the programmer. The native message-passing library on the CM-5 is CMMD.
3. Use a translator (e.g., CMAX) to convert a scalable FORTRAN77 program into a CM Fortran program.

Program development is also aided by an integrated graphical development environment called PRISM which includes a multi-language debugger with performance statistics gathering and reporting capabilities. In addition, the CM Scientific System Library (CMSSL) is a parallel library of high- performance mathematical and communication software, optimized for the CM-5 architecture.

Two visualization libraries are available for the CM-5: CMX11 and CM/AVS. Both are designed to integrate very large parallel data sets produced by CM-5 computations into standard serial visualization environments. In the next section we describe the CM/AVS.

3. CM/AVS: A PARALLEL AVS LIBRARY

CM/AVS, from Thinking Machines Corporation[4], extends the Application Visualization System (AVS)[1] 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 some local 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 modules; the user is free to mix 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.

3.1 Parallel 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.

A single 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. This allows functions such as `CMAVS-field_reset_minmax` to operate correctly for serial or parallel fields.

This bit can be set either by calling `CMAVSdata_alloc` to allocate a parallel field or by using the new `PARALLEL` flag in `AVScreate_input_port`. For example:

```
iport = AVScreate_input_port("input field","field 2D
            4-vector byte",

REQUIRED | PARALLEL);
```

will allocate a parallel field to hold an image that arrives on the module's input port.

The data member of a parallel field is simply a pointer to a parallel variable on the CM-5. If the parallel field is recti-linear or irregular, another parallel variable will hold its coordinate mapping array. To access pointers to these parallel variables, we add 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 C*, such a parallel variable must also have a shape that describes how the data are spread across the processor nodes and provides context flags which can be used to restrict operations to an arbitrary subset of the data; for C* we add `CMAVSfield_alloc_data_shape` to allocate this shape.

In CM Fortran, pointers are not available. So, using a mechanism similar to that of `AVSfield_data_offset`, 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.

To give an idea of how CM/AVS modules are written, Figure presents a simple module which computes the luminance of an image on the CM-5.

3.2 Communication

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.

The most direct and efficient method of communication is available 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. While this is obviously very fast, it is restricted to subroutine modules and requires that the modules be written so that they can cooperate.

All other communication methods are accessed through parallel sockets. Parallel sockets are optimized for transferring large amounts of data to and from the CM-5 using standard protocols such as TCP/IP. They are accessed with special system calls, such as CM_read, that accept a standard UNIX file descriptor (opened in the conventional manner), a pointer to parallel memory, and the number of bytes to be read or written.

Parallel sockets provide a layer of abstraction which allows CM/AVS to operate efficiently in many different heterogeneous computing environments. We discuss three example transport mechanisms, in order of decreasing bandwidth.

```
int
  luminance_compute(AVSfield *in, AVSfield **out)
{
  shape Image;
  Byte:void *in_data, *out_data;
  int result, dims[2];

  /*  Set up the weights for NTSC luminance */
 float red_weight= .299, green_weight= .587, blue_weight= .114;

  /* Get pointers to the arrays containing AVS field data */
  result = AVSfield_get_dimensions(in, dims);
  Image   = CMAVSfield_alloc_data_shape(in);
  in_data = CMAVSfield_data_get(in,Image);

  /* If there is already output data, deallocate it. */
  if (*out != 0)
    AVSdata_free("field", *out);

  *out = CMAVSdata_alloc("field 2D scalar byte", dims);

  with (Image) {
    /* Get a pointer to the output data */
    out_data = CMAVSfield_data_get(*out, Image);

    /* Copy the points from input to output */
    result= CMAVSfield_copy_points(in,*out);

    /* Compute the luminance */
    *out_data = in_data[1] * red_weight   +
 in_data[2] * green_weight +
 in_data[3] * blue_weight;
  }
  deallocate_shape(&Image);

  /* Return 1 to indicate success */
  return 1;
}
```

FIGURE 1. A C* CM/AVS module to compute the luminance of an image

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.

CM/AVS can also take advantage of HiPPI networks. A user need only make the appropriate entry in his AVS hosts file, so that the remote connection is made over this HiPPI connection.

Finally, if an Ethernet connection is available between the workstation and the CM-5, CM/AVS will use parallel sockets to transfer data over this connection.

3.3 CM/AVS Modules Under DJM

The Distributed Job Manager (DJM) provides job scheduling, resource management, and accounting services. It is available on many platforms, including the CM-5[15].

Users submit jobs to DJM queues by using the jsub or jrun commands. These commands ask the user to specify the resources (memory, time, etc.) that the job requires. When the resources become available, DJM schedules the job for execution.

Recall that remote AVS modules are executed by rsh[1]. The particular rsh command to be used is specified in the second field of the user's AVS hosts file, and this command is prefixed to each module invocation. To submit a CM/AVS job to a DJM queue, we use a level of indirection. We modify the rsh command to run a script instead of executing the module directly. This script executes the appropriate jsub or jrun command to submit the module for execution. The script may even take conditional action (for example, it is not usually necessary to submit the AVS list_dir process to a DJM queue).

Because DJM requires specification of resources when submitting a job, it is not immediately clear how one should submit an interactive job. If a user is interactively exploring a data set and using a broad palette of modules which are combined into a single process, the amount of time required for a CM/AVS job will not be known ahead of time. To address this issue, some sites create a special interactive queue where jobs are allowed to execute for many hours before being removed.

4. VISUALIZATION OF SIMULATION DATA

In our approach to visualization of data in a parallel computing environment such as CM-5, we have included post-processing of data and tracking of simulations. Both of these have been accomplished through CM/AVS in a distributed environment.

4.1 Saturn Ring Data

Our first application to test our system was a module to read and visualize a dataset consisting of the orbital positions and velocities of an n-body simulation problem of the Saturn rings. “Hill's problem” is the motion of a test particle in the gravity field of a central body (Saturn, in our case) and one or more perturbers (Saturn's moons). The computations were performed on the CM-5 at NAS[5]. The test particles represent “ring particles” or “moonlets” embedded in the F-ring region of Saturn. The F-ring is the next-to-outermost ring of saturn - the one with the interesting “kinked” and “braided” structures that were observed by Voyager 1 (and gone by the time Voyager 2 arrived). Saturn is simulated as an oblate planet, and the perturbers as point masses. The test particles (8000 of them) are “massless” and do not effect each other. The simulation is three-dimensional and is done in double precision. A parallelized Bulirch-Stoer integration scheme is utilized. At each timestep, it integrates using several different step sizes and then extrapolates to an infinite number of infinitely small steps.”Shadow” trajectories are computed, one for each test particle, to generate Liapunov exponents. Each massive body is taken sequentially. Its effect on all other bodies (massive and massless) is computed in parallel.

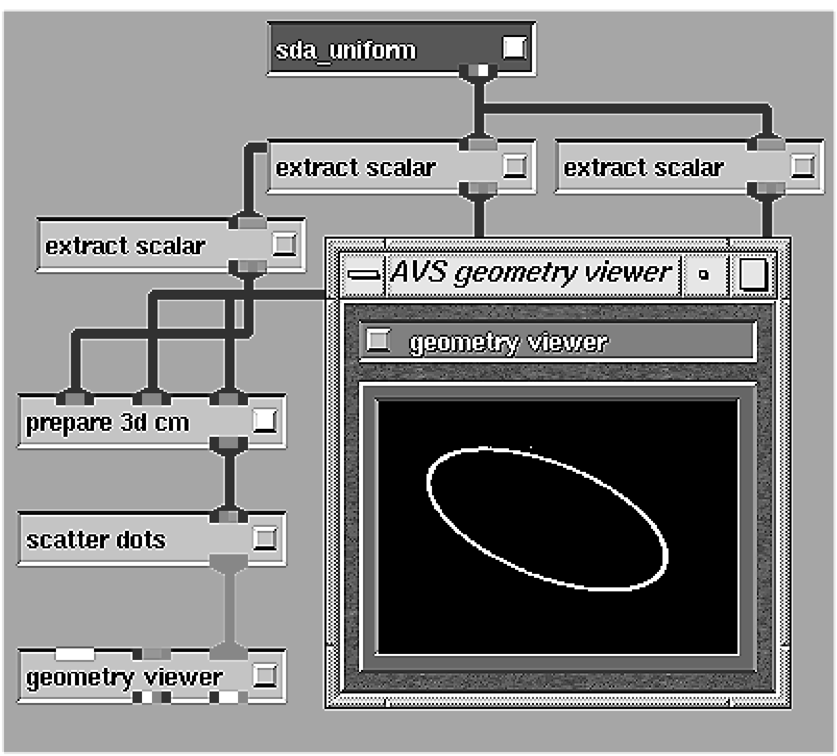

FIGURE 2. Saturn Data: an AVS network and a scatter diagram of particle positions.

Data (x,y,z,u,v and w) representing position and velocity components for each particle are output to the SDA approximately every 100 orbits. Runs of over 10^5 orbits have been computed.

Visualizations consist of 2-D plots and 3-D scatter plots of orbital elements (semi-major axis, eccentricity, inclinations, etc.), changing with time. Figure 2 shows the network and a scatter diagram of the particle positions.

4.2 Post-Processing PLOT3D Data

CFD data at NASA Ames are usually written in a PLOT3D format[18]. For parallel arrays on the CM-5 an appropriate PLOT3D format had to be defined. Efficient parallel I/O operations were considered in the design of the CM/AVS module which would read the pre-computed CFD data stored on the CM-5 SDA. Our initial implementation of the parallel PLOT3D reader can only handle single block data at this time. We plan to expand this module to read multiblock datasets. The module is useful for static interrogation of precomputed data. Many AVS modules for CFD analysis are available once the parallel CM/AVS field containing values of density, velocity components, and pressure have been read from the SDA. A density isosurface from a parallel PLOT3D dataset is shown in Figure 3.

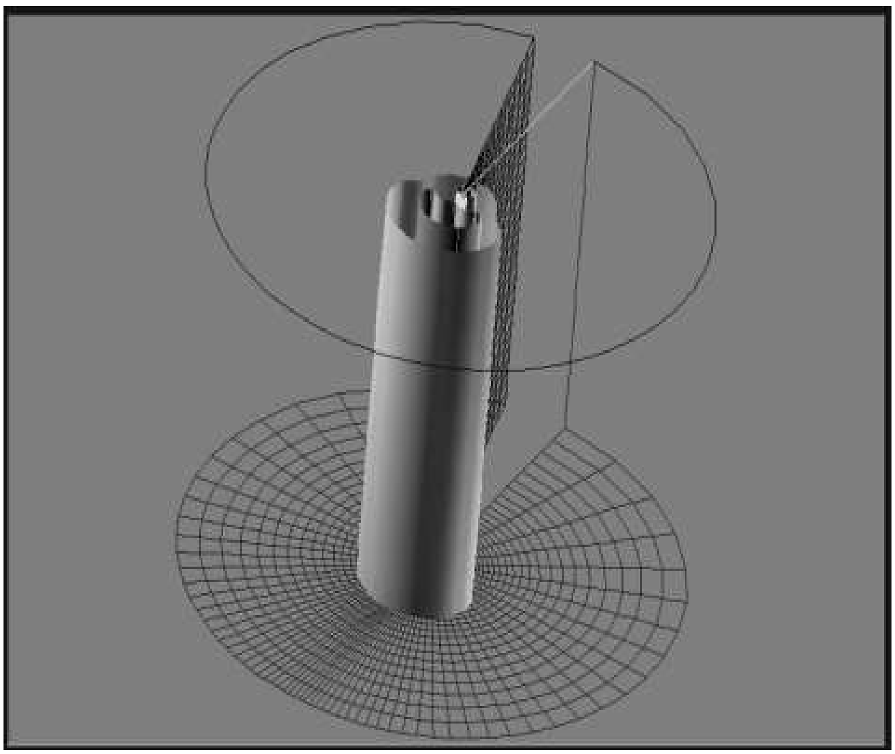

FIGURE 3. PLOT3D data: Density isosurface for flow around cylinder.

4.3 Simulation Tracking

We are interested in solving a class of 3-D unsteady flow problems for which the simulations are fast enough that the data transfer and visualization can be performed at interactive rates. We have found that as long as the solution continues to change at a rate of one frame every few seconds, visualization tracking can be quite useful. 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 both the current limitations in computer and networking technologies, and the inability of a data flow visualization environments to adequately and efficiently process transient flow data[7][12].

We have used the parallel CFD solver “cm3d”, developed by Jesperson & Levit at NASA Ames[9], for our computations. 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 has been 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 hot-wire and spectral data are very similar to experimental results[9]. 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. In our 3-D, unsteady flow simulation we cannot modify the simulation as it progresses, but 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. In Figure 4, we show the complicated structure of a density isosurface from a simulation of the unsteady flow past a tapered cylinder.

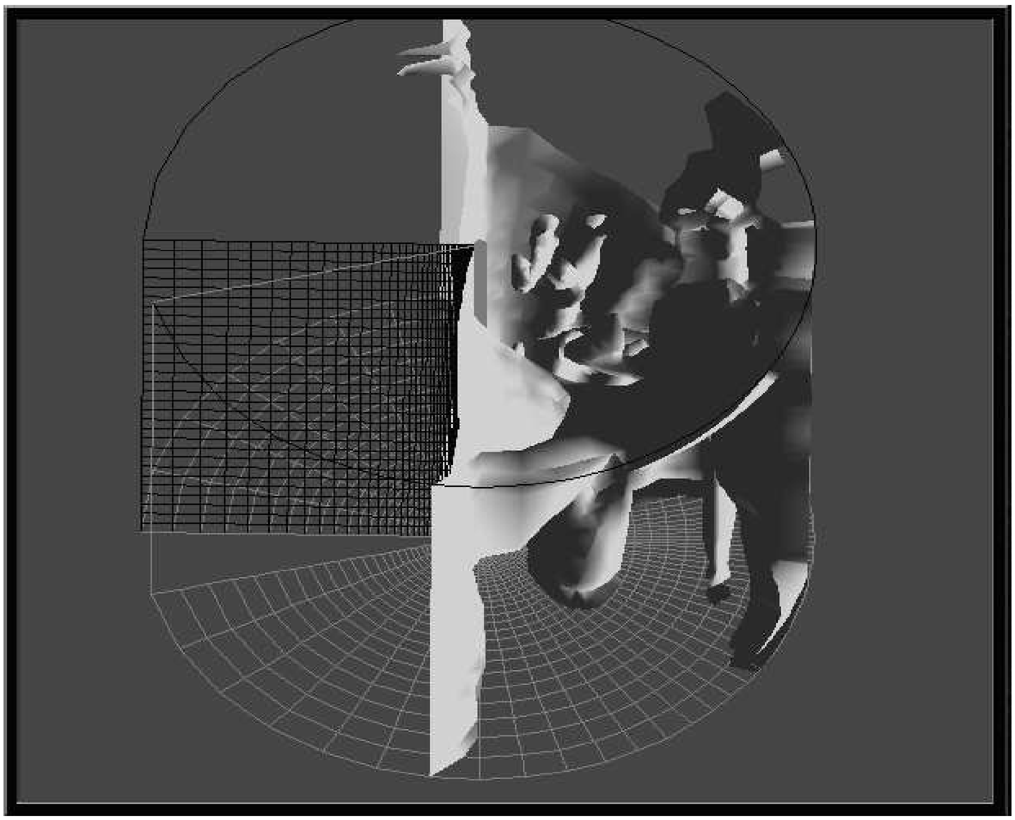

FIGURE 4. Density isosurface for an unsteady flow past a tapered cylinder in a simulation tracking experiment.

Our initial test problems are a single zone, 131,072 point (64x64x32) flow past a tapered cylinder and 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. In order to get a reasonable transfer time, the data for visualization are subsampled at the source. We have had to hard-wire a down-sizing routine inside the CM/AVS solver module for efficient subsampling. The subsampling requirement should not be necessary once we begin using a HiPPI channel for communication.

We have begun to define a class of computation/visualization problems which are of significant size (in the number of grid points or the number of dimensions) but yet are small enough that we can deal with them effectively in a distributed environment.

5. CONCLUSIONS

This paper has described the successful initial implementation of an interactive distributed data-flow visual parallel tool. Our experience during this initial phase of working with CM/AVS has indicated the following:

- A data-flow visual programming interface, such as AVS, can be an effective visualization/computation tool on a parallel computer such as the CM-5. It provides a workable platform to develop new visualization techniques for unsteady flow simulations.
- Simulation tracking and simultaneous computation/visualization is achievable for non-trivial problem sizes of computational interest.

- Improvements in network speeds are needed to effectively transmit the data between heterogenous computing platforms. We must currently use data subsampling techniques to achieve desired interactive speeds.
- For a limited problem size domain, in time-dependent CFD simulations, the cost of recomputing is reasonably low. Simultaneous recomputing/visualization my be a viable alternative to storage of large data sets and their post processing. We will continue exploring the effectiveness of computation/visualization and use fast parallel computation rates to leverage development of new visualization tools. For example, a parallel particle tracer is under development.

6. ACKNOWLEDGMENTS

We have greatly benefited from discussions with Dennis Jesperson, Tom Pulliam, and Horst Simon during the conduct of this research. This work was partly supported through NASA contract NAS2-13537.

7. REFERENCES

[1] AVS User's Guide, Advanced Visual Systems, Version 4, May 1992.
[2] Bryson, S. and C. Levit: The Virtual Windtunnel: An Environment for Exploration of Three-Dimensional Unsteady Flows, Proc. IEEE/ACM SIGGRAPH Visualization '91, San Diego, CA, October 1991.
[3] CM-5 Technical Summary, Thinking Machines Corporation, 1992
[4] CM/AVS Users Guide, Thinking Machines Corporation, 1993.
[5] Cuzzi, J., C. Levit, et al. "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, 1993.
[6] Gerald-Yamasaki, M. J.: Cooperative Visualization of Computational Fluid Dynamics, Computer Graphics Forum, 12(3), 1993.
[7] Globus, A.: Perspectives on the IRIS Explorer Visualization Environment.Report RNR-91-021, NAS Applied Research Branch, Moffett Field, CA, May 1992.
[8] Globus, A.: 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.
[9] Jesperson, D. C., and C. Levit: Numerical Simulation of Flow Past a Tapered Cylinder, AIAA paper 91-0751. AIAA 29th Aerospace Sciences Meeting. January 7-10, 1991, Reno, Nevada.
[10] Jordan, K. E., M. D. Kremenetsky, and J. L. Richardson: Visualizing Navier-Stokes Solutions Using the Connection Machine. Proc. Seventh IMACS International Conference on Computer Methods for PDE, New Jersey, 1992.
[11] Lane, D. A.: Visualization of Time-Dependent Flow Fields, Visualization '93 Proceedings, 1993.
[12] Mayer, H. F., and B. Tabatabai: Visualizing Results of Transient Flow Simulations, Visualization '93 Proceedings, 1993.
[13] NAS User Guide, NAS System Div., NASA Ames Research Center, Moffett Field, CA, 1993.
[14] Simon, H. D., W. R. Van Dalsem, and L. Dagum: Parallel CFD: Current Status and Future Requirements. In Parallel CFD, Implementations and Results Using Parallel Computers. H. D. Simon (Ed.), The MIT Press, 1992.
[15] Using DJM on CM-5. Thinking Machines Corporation, version2.0, January 1994.
[16] Vaziri, A.: Scientific Visualization in high-speed network environments. Computer Networks and ISDN Systems, Vol. 22, 1991.
[17] Walatka, P. P., J. Claus, R. K. McCabe, T. Plessel, and R. Potter: FAST User Guide, NASA Ames Research Center, Moffett Field, CA, July 1993.
[18] Walatka, P. P., P. G. Buning, L. Pierce, and P. A. Elson: PLOT3D User's Manual, NASA Ames Research Center, Moffett Field, CA, July 1992.
[19] Woodward, P. R.: Interactive Scientific Visualization of Fluid Flow, Computer, Vol. 26, No. 10, October 1993.