J6.2 VisAD FOR COLLABORATIVE VISUALIZATION OF DATA DISTRIBUTED ON CLUSTERS

Bill Hibbard*
University of Wisconsin - Madison and the NCSA Alliance

1. INTRODUCTION

Our Vis5D system is widely used to visualize numerical environmental simulations. It typically runs on workstations to visualize output from models that run on workstations or supercomputers. Very large model runs from supercomputers have typically required that Vis5D run on the largest workstations. Whereas most scientists have desktop workstations, scientists at the largest institutions have been able to purchase very large workstations from SGI and other vendors in order to visualize their very large model runs.

Now, however, vendors have ceased research and development for very large workstations and new desktop systems are overtaking the performance of the existing lines of large workstations. Note that this parallels the situation in supercomputers, where research emphasis is changing to large clusters of ordinary processors in place of very large processors. Some large institutional Vis5D users are wondering how we will address their visualization needs as specialized large workstations cease to be available.

2. VisAD FOR MODEL VISUALIZATION

Vis5D development has virtually ceased at SSEC, in favor of VisAD development. We are grateful that Vis5D development continues in the form of D3D from NOAA Forecast Systems Lab, Vis5d+ on SourceForge from Steven Johnson of MIT, the NCAR version, Cave5D from Argonne National Lab, and a number of other projects. However, our efforts now focus on VisAD in collaboration with the Unidata Program Office, the Australian Bureau of Meteorology, NCAR, NCSA, the University of Jena, and other institutions. VisAD is designed to be much more general than Vis5D, applying its same 3 D displays and interactivity to all types of environmental data including data from models, satellites, radars, point observations, and other sources. In fact, by virtue of its generality, VisAD is finding wide application in biology, astronomy, engineering and finance.

The generality of VisAD is based on its unified data model that reduces any data to expression via basic mathematical elements. This can be seen in VisAD's text shorthand for data schemas. For example, a single band image would have the schema:

$$((\text{line}, \text{element}) \rightarrow \text{radiance})$$

A more complex schema would express model output as:

$$(\text{time} \rightarrow ((\text{row}, \text{column}, \text{level}) \rightarrow (\text{temp}, \text{pres}, \text{water}, \text{wind\_u}, \text{wind\_v}, \text{wind\_w})))$$

with an associated invertible coordinate transform:

$$(\text{row}, \text{column}, \text{level}) \leftrightarrow (\text{latitude}, \text{longitude}, \text{altitude})$$

for expressing a map projection (note there may be a different map projection for each time step). Any of the primitive real types may have associated units, for example degrees for latitude and longitude, seconds since 1 January 1970 for time, and Kelvin for temperature. Any of the dependent values in functional relations, such as temperature and pressure, may be marked as missing or carry error estimates. Functional relations are typically represented by finite samplings, for example a finite sampling of time values and a 3 D grid in (row, column, level) coordinates. Note however, that unlike Vis5D, VisAD is not limited to regular grid samplings but can manage any regular or irregular sampling. Also unlike Vis5D, VisAD is not limited to a set of system-defined map projections, but can manage any user-supplied invertible transform between grid and earth coordinates. In fact, because of Java platform independence, user-supplied code for coordinate transforms become part of the data and can be moved between machines as part of data objects.

VisAD is a programmer's library rather than an end-user system. Thus in order to replace Vis5D it needs to be extended with model-specific user interfaces and applications. Efforts to do this have begun, notably the 3 D gridded data viewer that is part of Unidata's Metapps package, and Tom Whittaker's JMET.

3. VISUALIZING LARGE MODEL DATASETS

Models produce datasets too large to fit in workstation memory. In Vis5D we solved this problem by making the system manage caching of data between disk and memory in a way that was invisible to the user, including strategies for minimizing response times to user interactions.

Guided by this experience with Vis5D, VisAD also includes a general mechanism for transparently managing data caching between disk and memory. The system includes interfaces to a variety of file formats, and it is up to each interface whether to implement the caching mechanism. Among the twenty or so file

* Corresponding author address: Bill Hibbard, SSEC, University of Wisconsin, 1225 W. Dayton St., Madison, WI 53706; email: hibbard@facstaff.wisc.edu

formats currently supported by VisAD, the interfaces for Vis5D files, netCDF files, HDF EOS files, DODS servers, and VisAD binary files (i.e., the VisADForm interface) implement data caching.

Transparency of caching means that it is invisible to the user interface and to the API. This means that programmers writing model visualizations applications, such as Metapps and JMET, can ignore it. However, they also have the ability to control it if they like, by defining custom extensions to VisAD's CachingStrategy class.

In both Vis5D and VisAD, transparent data caching can provide visualization access to data sets too large to fit in memory. But in both systems, individual 3 D spatial grids are required to fit in memory. This was an effective approach as long as SGI and other workstation vendors were developing super large workstations. But now another approach is required for the largest model data sets.

4. PROCESSOR CLUSTERS FOR VISUALIZATION

The VisAD approach for visualizing the largest datasets parallels the way these datasets are produced on cluster of processors. Models running on clusters typically partition the horizontal earth domain (i.e., row and column) into blocks assigned to each processor. VisAD's object-oriented data model implementation makes it relatively easy for a single logical data object to be similarly partitioned across processors.

This approach has the advantage that it enables users to leave model output distributed across cluster processors where it is computed by models, rather than collecting it into one place. For large clusters, collecting data into one place can take much longer than the model computation that produces the data. Any time data has to all flow through or to a single processor, it becomes a bottle neck relative to the parallel performance of the cluster.

Of course, the user's workstation screen is a such a single point bottleneck. There are efforts to partition visualization screens across many processors. But the partition of model data does not correspond to the partition of data depictions, and in fact the way data must be sorted and routed between these two partitions is constantly changing in response to user rotation, panning and zooming of visualizations. Special and expensive cluster network architectures are required for this data re-sort to not be a single-point bottleneck. Furthermore, large partitioned screens require scientists to go to special rooms to do their work, whereas they need access to visualizations in their own offices.

In the VisAD approach, visualizations are produced on scientists' workstations. In order to avoid the workstation becoming a bottleneck, data from most cluster processors are visualized at reduced resolution. But the user can interactively select full resolution visualization from individual processors. This approach is reasonable, given the basic assumption that the motive for partitioning data is that a single spatial grid is too large for visualization by a single processor. Given the performance of current workstations, this implies that the spatial grid is so large that a full resolution visualization of an entire spatial grid will be too complex for users to understand all details at once. A low resolution visualization of the entire grid is equivalent to currently feasible Vis5D visualizations. In such a visualization it is not unreasonable for users to select high resolution in only the regions of interest, selected on the basis of low resolution visualizations. This basic approach can be the basis of experiments with numerous alternate user interfaces for selecting which processors produce full resolution visualizations.

5. COLLABORATIVE AND REMOTE VISUALIZATION

VisAD provides several mechanisms for supporting collaborative and slave displays. These are displays generated on other workstations that mimic the contents of a primary display. These allow scientists at geographically distributed workstations to share the same visualization, including interactive control of the visualization.

In order to reduce network bandwidth requirements to scientists at workstations remote from the cluster where model data are stored, slave displays can be generated by transferring only the final rendered images. This can be done using Java RMI distributed object technology, or using TCP/IP sockets when images are sent to Java applets running in web browsers.

6. ACKNOWLEDGMENTS

I wish to thank Don Middlton of NCAR for helping to formulate these ideas, and Curtis Rueden and Dave Glowacki for their development of VisAD slave and collaborative displays. I also wish to thank NCSA for supporting this work, and in particular thank Bob Wilhelmson of NCSA.