Applications of Vis5D in the Cray T3E MPP Environment

Sergei Maurtis and Jeff McAllister
Arctic Region Supercomputing Center, University of Alaska
Fairbanks, P.O. Box 756020, Fairbanks, Alaska 99709, USA
maurits@arsc.edu mcallister@arsc.edu

ABSTRACT: *The volumetric visualization package Vis5D has rapidly gained popularity during the last few years. Designed primarily as a tool for a single-processor environment, the package is not straightforwardly applicable to MPP situations, even after porting challenges have been overcome. A suite of MPP C-routines, Vis5DR (Vis5D-Repository), was developed at the ARSC to bridge this gap and to facilitate convenient run-time output directly into a Vis5D-compatible format. Performing direct output from each PE to an individual file, the package effectively eliminates interprocessor data transfer for I/O, therefore boosting computational performance. The Vis5DR down-sampling and sub-ranging options at the post-processing stage provide a practical solution for visualization of gigabytes-sized outputs at midrange workstations.*

Introduction (Challenge of Large Dataset Visualization via Remote Access)

The output of supercomputer models cannot be understood without the use of visualization software. Visualization itself is often a task pushing hardware to its limits, although it frequently has to be delegated to mid-range workstation-class platforms. This contradiction arises from the widespread accessibility of remote means of data generation combined with the essential requirement that visualization resources must be local. A remote user is capable of generating —and storing— gigabytes and gigabytes of data on a state-of-the-art supercomputer, but typically has much more modest ways to visualize this data on locally available platforms. The large output size from MPP supercomputing runs, where data is generated from many processors, only exacerbates this contradiction.

These are the types of problems which we attempted to address specifically for users of the popular Vis5D visualization package. The set of MPP extensions developed took the form of a suite of utilities reading and writing a file format quite close to the native Vis5D format. Since at the same time the package makes cross-platform data storage more convenient, it is named Vis5DR (Vis5D-Repository). This developing approach targets the problems originating from porting the essentially "scalar" Vis5D to a CrayT3E environment and, more generally, to an environment which does not meet a number of assumptions implicit in the Vis5D code. The method eliminates the runtime time/memory overhead of assembling results for I/O from a large number of processors, addressing the design limitations of the original "scalar" Vis5D.

Finally, for initial visualization and review of simulated data, the package offers a set of downsizing options. These allow for reduction of the down-sampled dataset size by orders of magnitude. After the boundaries of regions of special interest are established in space and time, a return to full resolution is facilitated by re-scanning the initial dataset inside the established limits. Preparations of both reduced resolution databases and full-resolution sub-domains can be done either on a remote supercomputer or on a local workstation. Although adding this preparatory previewing stage may seem to complicate the visualization process, it can frequently make a difference between the ability to look into the data immediately and just storing it for an indefinite wait.

The process of creating a Vis5D port for the T3E led to a greater understanding of the differences between the demands of parallel and scalar environments and how these intersect with what off-the-shelf visualization packages can offer. This paper presents an approach which started as a custom solution for visualization of the University of Alaska Fairbanks Eulerian Parallel Polar Ionosphere Model (UAF EPPIM) which has been generalized to support rather arbitrary domain decomposition problems. Vis5DR is still not a full port of all Vis5D-supported features, particularly various geographic projections. Part of the purpose of this publication is to discern how much interest there would be in adding more functionality and releasing the code as an open-source project for the community. It seemed that we must not be alone in finding the visualization of supercomputer model results more awkward than it needs to be.

Overview of Vis5D Visualization Package

Vis5D is a visualization system for gridded data produced by the University of Wisconsin Space Science and

Engineering Center under a NASA grant [1]. It is freely distributed under the GNU license. Its ports exist for many platforms (see for more information the Vis5D home page, www.ssec.wisc.edu/~billh/vis5d.html). Vis5D renders output of regularly gridded models as a 5-dimensional grid: three spatial dimensions, model variables, and timesteps. For each timestep Vis5D supplies various rendering options for 3-D variables, such as isosurfaces (including the nested isosurfaces of variable opacity for 3-D volumetric representation), slices, vector fields, and volume rendering by "colored fog" of variable opacity. Once the graphics are rendered, Vis5D supports real-time rotation, slicing, vertical profiling, and node-probing—all with full animation. Designed primarily as a tool for visualizing meteorological model outputs, Vis5D supports a variety of geographical projections and provides the possibility of superimposing data upon maps, geographic reliefs, and images. Vis5D's sophisticated control of a variety of visualization modes and representations has promoted this cross-platform format into one of the de-facto Internet standards for 3-D time-dependent data. Vis5D plug-ins for WWW-browsers are in widespread use in meteorology, space weather, and other applications. Vis5D users maintain an active mail-list at vis5d-list@ssec.wisc.edu. The latest 5.2 version of Vis5D was released in 1999 and several independent modifications of Vis5D are reported in the literature.

Vis5D allows three accuracy options for compressing data. The most aggressive compression dedicates just one byte per node per timestep per variable, distinguishing only 256 gradations in the variable's full range. Thus, this compression introduces a possibility for error estimated as half of +/- (min/max range)/256. This allocation can be increased to 2 bytes (32768 gradations) or to 4 bytes per node (no compresson), lowering the corresponding error on expense of proportional increase of the overall dataset size.

It is instructive to estimate the resulting dataset sizes, using dimensions of 256^3 as a frequently mentioned practical limit of current mid-to-high-range visualization platforms. To depict 256 steps of time-development with the highest compression of just one scalar variable combined with a 3-D vector field (i. e. density in a 3-D velocity field), it is necessary to allocate 16 GB. It is reported in the literature that datasets on the order of 10 GB in size were successfully rendered with Vis5D, but it took a top-of-the-line visualization platform — an SGI Onyx2 in multi-processor configuration with large RAM. Rendering Vis5D datasets of several hundreds of MB size is a more realistic upper limit range for most visualization platforms. Here is the gap between the output size modern models can produce and what can be visualized.

The process of Vis5D database preparation requires the use of a set of Fortran and C subroutines. It can be integrated into the numerical simulation (in this case the content of active arrays is stored into the database on every timestep), or postponed to the post-processing stage, when the data is read from the output file. Due to the possibility of aggressive data compression, the integrated output is especially useful. Although it requires a port of Vis5D to the computational platform of choice, these ports are readily available for all major brands of UNIX, Windows NT, and Linux. A remarkable exception from the list of available platforms and operating systems is Cray under UNICOS.

Porting Vis5D into CrayT3E Environment

Rationale (On Example of Parallelization of the UAF Eulerian Polar Ionosphere Model)

We found Vis5D provided excellent results for our project of 1997-1999, a computational optimization and parallelization of the University of Alaska Fairbanks Eulerian Polar Ionosphere Model (UAF EPPIM). Vis5D-based visualization was an integral part of the development (and debugging) process, though at first this was mostly done in a workstation environment. After porting to the MPP Cray T3E environment we established that porting the Vis5D-related part would constitute an additional challenge.

The UAF EPPIM [2] is a first principles ionospheric model, which solves the equations of continuity, motion, and energy balance for seven ion species, electrons, and a few minor neutral atmosphere components important for the ionization balance. Its geographic domain covers a significant portion of the Northern Hemisphere (9,200x11,000-*km* area), and extends vertically from 80 to 500 *k*m. The normally selected model time resolution is five minute time steps. One of the variable parameters is the model's horizontal resolution, which sets the cell-side size from 220 *km* up to 10 *km* as a uniform mesh throughout the entire domain. The highest resolution requires a grid size of ~900x1100x43. The sum of all required variables can only be accommodated on platforms with at least 8GB of RAM. Realistically, minimal partitions required for this problem size on Cray T3E start from 40PEs, provided that at least 256 MB of RAM per processor is available. The simulation of an environment as variable as the terrestrial ionosphere requires sampling with small intervals in time, preferably at each timestep. Thus the output in full textual or binary representation of even a few variables would require a file of unpractical size.

The advantages of using Vis5D as a visualization tool in this situation are obvious. The Vis5D "byte-per-node" compression applied during the runtime can significantly reduce the output size. The Vis5D format cross-platform compatibility would allow for straightforward visualization of the MPP output, without using some intermediate representation. At that stage, the only missing link was compiling the Vis5D code for the CrayT3E–compatible binaries for direct run-time output writing in Vis5D format.

Alternative Format Vis5DR as a Porting Solution

However, merely compiling the code was not successful. The v5d.c and binio.c codes (libraries required for Vis5D output) compiled and linked with little effort, producing an output file which had the initial appearance of being a valid Vis5D file. However, the viewer would reject it. Closer investigation revealed that the header was corrupted when output was attempted on the T3E. Many of the internal subroutines of the Vis5D code force a 32-bit word size, and this produces unacceptable results on the 64-bit T3E. After attempting some of the easy fixes, such as trying out different compiler options to force IEEE format and specify integer and float sizes (which did not work well on the source code of ~100,000 lines size), a feasible solution was found. We succeeded in producing an output with roughly the same information in the same order as in the standard Vis5D header which would read and write the same on the T3E and on workstations. This was accomplished by writing the Vis5D-header values as universal human and computer-readable text representations instead of binary representations, and in general removing any dealings with native word-size. With the same values read and written everywhere, when the modified Vis5DR format is converted to standard Vis5D on a Vis5D-compatible workstation, it produces valid Vis5D files. Fortunately, the new Vis5DR format did not significantly increase the output size, since all modifications were concentrated on the database header, while the unmodified Vis5D format was perfectly fine for writing the data grids.

As it follows from the discussion above, this port does not fully include CrayT3E to the list of Vis5D compatible platforms. The proposed method involves an additional step of converting to true Vis5D format on another system. On the other hand, an absence of a graphics subsystem on the CrayT3E promotes this partial solution to a rank of the most practical approach. To the best of our knowledge, this is the only successful port of the Vis5D database format writing capability to the CrayT3E, although a number of such attempts were discussed on the Vis5D mailing list.

The standard solution to which users frequently resort is to write the outputs in one of a variety of cross-platform formats and then transfer them into Vis5D-representation on some supported platform out of the MPP realm. Although a transfer to true Vis5D is currently also necessary for the Vis5DR representation, the Vis5DR format is extremely close to the native Vis5D format. As such, Vis5DR facilitates very fast transfer to Vis5D, utilizes the “byte-per-node” Vis5D compression, and holds promise to be directly accessible by a Vis5DR-based viewer in the future. Such a possibility at this time is under serious consideration.

MPP Environment: Difficulties and Approaches

The decision to depart from the standard Vis5D format for the output solution provided more freedom to deal with substantial deviation from Vis5D’s implicit scalar paradigm in the MPP distributed environment. The MPP I/O of the entire 3-D array leads to the well-known challenges of synchronization, excessive data and/or control transfer, and other partition-specific bottlenecks. Conceptually, the Vis5DR format is free of these difficulties. The Vis5DR paradigm is “one processor – one file”. The output is accomplished via writing the content of local memories into processor-specific files. Thus, an nPE-partitioning (including the case of one PE) would result in n-files. In addition to the data grids, each file header also saves information about which part of the domain it represents. If the simulation is time-dependent, the output can be done in the main loop by appending each file on every time step.

Though the standard Vis5D uses implicit global addressation (Ix = 1, MaxX; Iy = 1, MaxY; Iz = 1, MaxZ), the Vis5DR writer explicitly requires the limits of 3-D sub-arrays in each dimension (Ix = Xbegin, Xend; Iy =Ybegin, Yend; Iz = Zbegin, Zend). This organization supports practically arbitrary partition if based on rectangular elements. Extensive testing was performed for the parallel code immediately available to authors, which included both “vertical slabs” and “horizontal slices” in partition of the 3-D domains. More complicated partition types may require additional testing and/or coding, but the potential to handle them is inherently included into Vi5DR format as a header information.

Another major difference between the Vis5D and Vis5DR paradigms is the implementation of data compression. As it was described earlier, Vis5D compression is based on determination of global min/max of a variable and representation of this range with a certain number of gradations, starting from 256 to accomplish “byte per node” compression. Instead of determining the global min/max for each variable at the end of the run, the current version of Vis5DR employs a user-defined range for each variable. This unifying approach has proven practical for many situations. For instance, it unifies the color palette representation enabling comparisons between different runs, which otherwise would have different minimums and maximums and thus would be colorized quite differently.

The existence of many output files implies a need to combine them together at some point. There are several options in Vis5DR to handle this assembling. All of them use one processor and postpone the assembling operation until after the MPP-run is completed. This solution emphasizes a practical approach, which values the MPP batch-time compared to a single processor interactive environment. Overall, it can minimize throughput time. The partition files can be assembled into a full file on the CrayT3E or be transferred to the user’s workstation and combined into one file there. An important consideration, the decision to leave the files on the supercomputer’s temporary partitions or to move them to user’s workstation can be influenced by the size of the data. Vis5DR fully supports either way, and this appears to be an important enough feature to be emphasized in the package name (ViS5D-Repository).

As soon as the partition files are assembled (and deleted), the single Vis5DR file is ready for further transformations into true Vis5D format. In addition to straightforward conversion, Vis5DR supports a variety of down-sampling and sub-ranging options to reduce Vis5D dataset size. These options are especially important for an MPP environment, where a simulation can easily generate gigabytes and gigabytes of 3-D time-dependent data. The Vis5DR converting options emphasize a number of means to reduce the dataset size to a manageable level for comprehensive review and then to return back to full resolution in time and space for viewing of regions of special interest. The next chapter describes current functionality of the Vi5DR converters.

Down-sampling and sub-ranging options for Vis5D dataset size reduction

Vis5DR utilities and their current functionality

Together with subroutines expanding the Vis5D-format writing functionality to the MPP-environment, the Vis5DR package includes two post-processing utilities: ***joinv5dr*** and ***extractv5dr***. The ***joinv5dr*** utility assembles partition files into a single file, either preserving Vis5DR format (in this case ***joinv5d*** can be used on CrayT3E), or transferring directly into a Vis5D dataset (only on Vis5D-compatible platforms). This utility also has some minimal control functions, such as limiting the transfer at a specified time step. The ***joinv5d*** utility supports verbose mode and can provide all necessary information about the dataset if ***–info*** flag is invoked.

The second utility ***extractv5d*** facilitates operations with a single Vis5DR file, previously assembled from the partition output files by utility ***joinv5d***. ***Extractv5*** can be used both for transfers to Vis5DR formats (including applications on CrayT3E) and to true Vis5D (on Vis5D-compatible platforms only). A set of currently supported transfer options include

-v	verbose mode
-info	information about file content
-v5dr	left the output file in Vis5DR format (without this flag —transfer to true Vis5D)
-X # #	output only sub-range beginning Ix = # to end Ix = # in X-direction
-Y # #	output only sub-range beginning Iy = # to end Iy = # in Y-direction
-Z # #	output only sub-range beginning Iz = # to end Iz = # in Z-direction
-T # #	output only sub-range of timesteps: beginning # to ending #
-noVar #	do not include variable # into output
-step #	skip every #-steps in each X- and Y-directions

The options are self-explanatory and their combination allows sub-ranging the initial array in time and space and/or down-sampling its resolution. Certain discrimination against the Z-direction (no down-sampling option) is a temporary reflection of the special treatment of the vertical direction in Vis5D. It will be generalized in coming versions of Vis5DR. This would be implemented as more versatile options to skip different numbers of steps in each direction, as well as in time. However, even the limited set of options currently available can tremendously reduce Vis5D file size while preserving its representative capability. A case study applying the Vis5DR package to two realistic mid-range MPP simulations will be discussed below.

Case Study: Example of an MPP-Run of the UAF EPPIM

This section illustrates a use of Vis5DR functions in a rather typical CrayT3E MPP situation. The period–specific reconstruction of a polar ionosphere event on January 22nd, 1998 was performed with the UAF EPPIM at the Arctic Region Supercomputing Center. The model's horizontal resolution for this run was chosen at 14x14 km, which resulted in a problem size of 641x737x43 for about thirty 3-D arrays. In the case of the UAF EPPIM, the domain decomposition uses the vertical "slabs" (Ix = 1, MaxX; Iy = Ybegin, Yend; Iz = 1, MaxZ) for treatment of predominantly vertical diffusion, while the horizontal slices (Ix = 1, MaxX; Iy = 1, MaxY; Iz = Zbegin, Zend) are used to compute advection (Figure 1). The model arrays were allocated to partition of 36PEs. For UAF EPPIM this is a maximal number of PEs for "symmetric" partition (Figure 1), when each PE has both vertical "slab" and horizontal slice allocated to it (the model accounts for advection only in the range covered by 36 horizontal layers out of total 43). Thus, for the run in question the horizontal partition on 36 PEs reduced the slice "thickness" to one element, while the slice "width" and "length" included full range index addressation. As for the vertical "slab", its "width" was 19 or 20 elements per PE, while its "height" and "depth" used the full range of indices in Z- and X-direction. Such a detailed description of the MPP-partitioning of the ionospheric model is necessary to describe the MPP capabilities of Vis5DR, particularly the I/O options, which were tested during the runs.

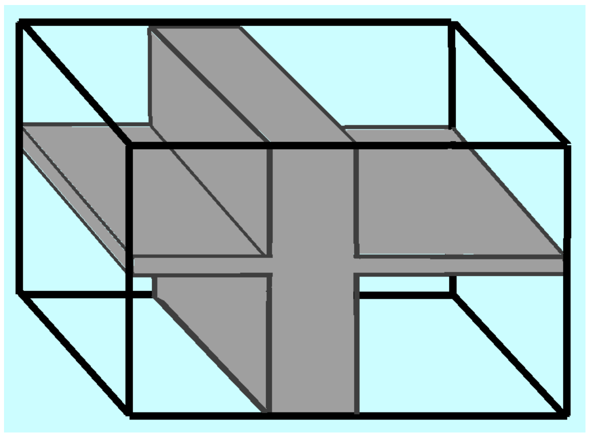

Figure 1. The UAF EPPIM domain decomposition, which minimizes the interprocessor exchange for the direction splitting algorithm of the model (the parts of 3-D arrays allocated to one PE in the partition are shaded). During the first test the I/O was performed from the vertical "slabs", during the second run the variables allocated to the horizontal "slices" were output, see text for details of the output procedures implementation.

The job was placed in a batch queue with a request for 36 PEs and 8 hours of computation time, which constitutes a "large" queue job of regular duration at the ARSC. The model solution converges to the period-specific patterns during first 60 timesteps or five hours of model time. Upon convergence, two volumetric variables of the model (out of about thirty) were output at every time step of 5 minutes together with a few 2-D variables. The output was produced only in Vis5DR format into 36 files, correspondingly to the number of PEs in partition. During the 8 hours of the run, simulations of 178 timesteps were accomplished, while last 118 timesteps were output. The resulting size of the 36 output files was in the range 132 to 138 MB due to disparities in vertical partitions by just one element (19 or 20).. Their assembly into a single file by the ***joinvis5d*** utility was limited to the first 100 timesteps of output. The size of the resulting Vis5DR file was 4.11 GB. It took ~80 minutes of CrayT3E CPU time to assemble it. in interactive mode on one PE. This operation can be done also as automatic submission in a single processor queue upon completion of the MPP run. As such, it can be considered as a measure to save MPP time to increase the system throughput.

The assembled full-resolution file of 4.11 GB was never moved from the CrayT3E. A preview of the results was accomplished by producing a down-sampled Vis5DR dataset with utility ***extractv5d*** with setting of key ***–step 16***. This setting resulted in reducing the size by a factor of 256, namely from 4.11 GB to 16 MB. It effectively down-sampled the dataset horizontal resolution to 220x220 km from the initial 14x14 km. Again, the procedure was performed by one CrayT3E PE in interactive mode and it took ~35 minutes to accomplish this transfer. The resulting Vis5DR file was transferred to an SGI O2 workstation and converted with ***extractv5d***, thus bringing the dataset to full Vis5D-compatibility. The dataset was opened for preview, and it was established that the domain central area was of special interest due to intensive plasma motions and pronounced density gradients.

In terms of X,Y-direction indices, this area was limited to a range of Ix = 200,550; Iy = 200,550. Storing the 2-D Ix, Iy-grid as the reference element of the model output definitely simplified this estimate. The height range of interest, 250 to 380 km, was also established, which corresponds to the vertical index range of Iz = 19,32. These limits constitute about 25% of the horizontal area of the full domain, while the altitude coverage is about 30% of the entire range (Figure 2).

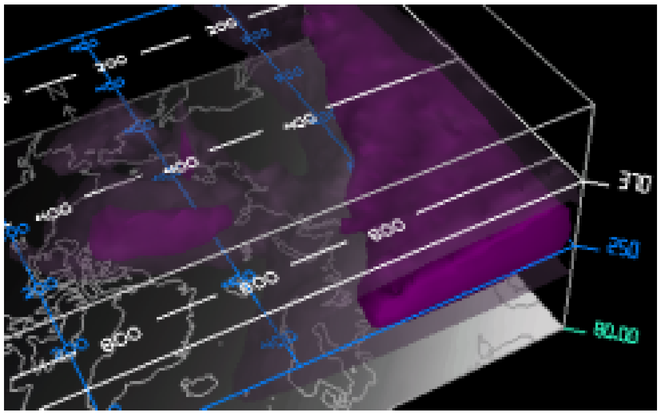

Figure 2. Searching for features of interest during preview of the down-sampled dataset (purple isosurfaces indicate ionospheric plasma high density). A placement of array’s indices into the data helps for fast identification of sub-range in X- (white grid) and in Y-direction (blue grid). Vertical range in *km* is easily established from the Vis5D display (in the right column).

The third application of ***extractv5d*** to the full-resolution dataset located on CrayT3E separated the selected sub-domain in full resolution into a Vis5DR file. Upon completion of this operation, which took ~55 minutes of interactive CPU time on CrayT3E, the resulting Vis5DR file of 310 MB was ftp-ed to SGI O2 and reverted to Vis5D representation by ***extractv5d***. The Vis5D version was opened by the Vis5D viewer and successfully rendered at full resolution for real-time analysis, including animations (Figure 3).

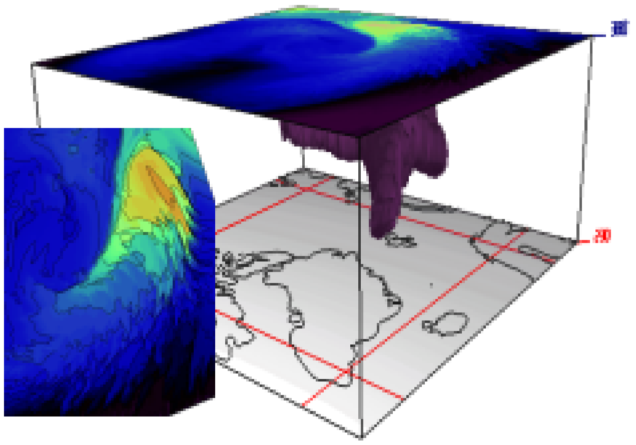

Figure 3. Rendering of selected sub-domain (Ix,Iy=250,500 is shown by red grid) at full resolution. Volume visualization techniques of Vis5D (dynamic isosurfacing, slicing, contouring) is fully applicable to the high-resolution large dataset and widely scalable for available hardware due to a step-by-step sub-ranging approach. For illustration of achieved resolution, a sample of distribution of electron density ***Ne*** is shown on the left (note the logarithmic scale, each contour represents 0.1 for *Log**Ne***, correspondingly to a gain of 1.25 for ***Ne)***

The SGI O2 workstation configured with 384 MB of RAM and with R10K 150MHz CPU, appeared to be an adequate graphical platform for the rendering task, especially for 3-D slicing and probing. Parameters of the workstation were intentionally selected in mid-range to simulate a situation of lack of local access to high-end visualization resources.

For testing purposes the entire procedure above was repeated for the second MPP run with output of variables allocated to horizontal “slices” —contrasting the initial run which used vertical “slab” output. Since the number of PEs as well as other parameters in both runs were the same, the timing results were not significantly affected. Rather, it was a test of applicability of the method and software to a different output scheme using other type of the MPP partition.

Summary and Conclusions

In addition to its sub-ranging and down-sampling options, the Vi5DR package allows for semi-direct use of the popular Vis5D format for MPP-runs on the CrayT3E platform. Summarizing Vis5DR functionality, one can conclude that the package

- represents a solution to problems of porting Vis5D to the CrayT3E and, in general, to any MPP-environment
- preserves Vis5D aggressive “byte-per-node” compression
- achieves a noticeable simplification of the MPP I/O paradigm with “one processor, one output file” approach
- shifts the time-consuming file transfers and transformations to a post-processing interactive stage
- creates an ability to use a crude resolution for the full domain preview, with later zooming in full resolution on sub-areas of special interest

The "one run, many views" approach has many advantages for supercomputer-scale output which has the potential to overload a remote user's visualization workstation hardware. The developing approach of data repository combined with its Vis5D-supported format allows for a very large amount of output data to be stored as a single large file on the very supercomputer where it was generated. Without moving this huge file over the network, a method of intelligent extraction of much smaller subsections of this domain is developed and offered as interactive process with reasonable throughput.

The sequence of steps described above shows a typical scenario of Vis5DR's intended use represented by the remote user with limited visualization and storage resources. It is important to note that the substantially-sized output of the case study run—more than 4 GB—never left the supercomputer's hard drive. Instead, the down-sampled preview file of 16 MB and full resolution subdomain file of 310 MB were assembled for ftp-ing to local visualization resources, and, upon conversion to true Vis5D format, for rendering. Overhead time for the partition files assembling, down-sampling, sub-ranging, and final format transfering appear to be rather reasonable for a task of this scale. The addition of a previewing stage with the preparatory work of selecting sub-volumes and sub-intervals of interest can be considered a small price to pay for the capability to visualize large outputs with pretty limited graphics and storage resources.

In fact, investing this time can make the difference between immediately visualizable data and just storing the simulated set for an indefinite wait. Storing the full-resolution dataset on a supercomputer with the potential of extracting much smaller subdomains of interest creates a controllable situation for the remote user. It would be not an exaggeration to say that this capability can affect the pre-run decision-making process about what is feasible to output and visualize. In our opinion, it can further promote use of volumetric output rather than the "representative slicing" paradigm for 3-D time-dependent simulations. Even though this approach is more resource-demanding, it offers many benefits, including the debugging potential of volumetric visualization.

Future work on the Vis5DR package includes plans to fully generalize its sub-ranging and down-sampling options with inclusion of the capability to skip time-steps. An ability to independently vary the skipping parameters in all three spatial directions would also be useful. Some convenience can be added by such features as determining the sub-domain boundaries not only in terms of Ix,Iy,Iz-indices, but by using the geographic coordinates and heights, which can be read from the map and from the altitude-axis, both readily available elements of the standard Vis5D display.

Selecting arbitrary elements during the resolution down-sampling process is fast, but is not the best option as far as the output visual quality and representativeness are concerned. Introducing averaging and weighted-averaging algorithms can improve the representational quality of intermediate outputs. Since these methods are more computationally expensive, it will require additional testing of the feasibility of such implementations in terms of the added CPU time. Also, as the post-processing development is seen by the authors as a single processor interactive or batch job, a potential problem here is per processor memory limitations.

Another potential improvement will be to increase the file compression aggressiveness still further. Identical numbers, if they happen to be in a sequence, can be represented in form of multipliers. One of the numbers in the 0 to 255 range can be reserved as a flag, indicating that the next value is a multiplier (which can not be larger than 255 to be represented by just one byte), and the value after the multiplier is the quantity to be repeated. A theoretical limit of such compression is a factor of 255/3 or ~85 for database of completely identical values. The compression factor range attainable in practice remains to be seen, as well as the penalty for its processing. However, it is clear that the reduction of possible values to only 256 gradations dramatically increases the chances for sequential values to be identical. Perhaps, a user-selectable choice "save size or save processing time" can be introduced as an option.

To the best of our knowledge, this port is the only working solution for a semi-direct application of Vis5D format in the MPP-environment. We would be very interested to share this package with the community as an open source project. At this time (May 2000) it is still in beta-testing. Upon completion, it will be released at the ARSC software depository (ftp.arsc.edu at /pub/software/sources).

References

[1] 1. Hibbard, W.L., B.E. Paul, D.A. Santec, C.R. Dyer, A.L. Battaiola, and M-F. Voidrot-Martinez, Interactive Visualization of Earth and Space Science Computations, *Computer*, **27**, No.7, pp. 65-72, July 1994.

[2] Maurits, S. and B. Watkins, UAF Eulerian Model of the Polar Ionosphere, *STEP Handbook of Ionospheric Models*, p. 95-123, 1996.

Acknowledgments

This research was supported in part by Cray Research and the University of Alaska Fairbanks Research and Development Grant Program, by the NSF grant ATM-9523818, and by a grant of HPC and visualization resources from the Arctic Region Supercomputing Center.