Contents

AVS Overview 3

New Software Products 7

AVS Network News 9

Using AVS for Prototyping
a Distributed Computational
Molecular Biology and Genetics System 10

Interpolating Gridded Data
from Scattered Data in AVS 13

Visualizing Molecular
Dynamics Simulations 17

AVS Version 4 18

AVS Newsgroup Update 20

Module Submission Criteria 21

Obtaining Modules 22

AVS Network News

The International AVS Center serves as a catalyst for expanding the AVS user base and for increasing AVS functionality by fostering discipline-specific module development and new AVS uses. Located at the North Carolina Supercomputing Center, the worldwide clearinghouse collects, ports, and distributes user-contributed, public-domain modules and acts as liaison between users and vendors.

AVS Network News is the Center's quarterly magazine serving users and vendors with updates on AVS programs and their use, users' color images and editorial viewpoints, and product information postings from vendors and independent software developers. For an annual subscription, send a $12 check or money order, payable to the International AVS Center, to Post Office Box 12889, 3021 Cornwallis Road, Research Triangle Park, NC 27709-2889.

Front cover design by Senior Scientific Animator Chris Landreth of the North Carolina Supercomputing Center. Cover inset, *Interpolating Gridded Data from Scattered Data in AVS*, by Wes Bethel of the Information and Computing Sciences Division, Computing Resource Department, University of California at Berkeley.

Back cover photography by
Jay Mangum Photography.

David Bennett, AVS Project Leader, **avs@ncsc.org**
Joel Page, Contributing Editor, **page@mcnc.org**
Fran Wise, Layout Coordinator, **wise@mcnc.org**

For more information
The International AVS Center
Post Office Box 12889
3021 Cornwallis Road
Research Triangle Park, NC 27709-2889
919-248-1100
avs@ncsc.org

TRADEMARKS IN THIS ISSUE

AVS is a trademark of Advanced Visual Systems Inc.
CONVEX is a registered trademark of CONVEX Computer Corporation.
CRAY Y-MP is a registered trademark of Cray Research Inc.
DEC is a trademark of Digital Equipment Corporation.
Ethernet is a trademark of Xerox Corporation.
IBM is a registered trademark of International Business Machines Corporation.
Stardent is a trademark of Advanced Visual Systems Inc.
UNIX is a registered trademark of UNIX System Laboratories Inc.
X-Window System is a trademark of the Massachusetts Institute of Technology.

AVS Overview

Visualization is a method of computing that has emerged from the combination of the latest advances in visualization software, graphics, networking, high-performance systems, and industry standards. Advanced Visual Systems Inc.'s Application Visualization System (AVS) incorporates all of these technologies into a single comprehensive visualization environment.

Visualization allows the interaction between the computer system and the user to occur through the most efficient form of information exchange — visual exchange. Sight is the primary and most efficient method for receiving, processing, analyzing, and assimilating information. Visualization is changing the way scientists and engineers perform their work.

The Complete Visualization Environment

AVS is the industry standard visualization application software and development environment. It is a fully functional visualization environment providing more techniques for visualizing information than any other system commercially available today. Available on the industry's major computing platforms, AVS is extensible, easily tailored, and applicable to a wide range of applications.

AVS accepts the data produced by instruments or by scientific and engineering simulation software. It provides you with the most comprehensive set of data types of any visualization system. It creates a visual display of your data in a variety of forms using a wide array of visualization techniques. AVS has a modular architecture comprised of many separate yet tightly integrated subsystems that each provide important capabilities.

The Building Blocks of AVS

Modules are the building blocks of AVS. A module is an independent computing element (C or FORTRAN) that is represented by a rectangular icon on the screen. Each module performs a specific visualization function or set of functions. You have access to more than 110 modules provided with standard AVS. Alternatively, you can build your own modules to extend or customize AVS. External programs or subroutines written in C or FORTRAN can be converted easily into AVS modules.

A wide range of data input, filter, mapper, and data output modules also is included in AVS. Filters transform data into data, e.g., contrast stretch or edge detect. Mappers transform data into geometry, e.g , isosurface or arbitrary slice. And data output modules write data to files, send data to peripheral devices, or render data, e.g., displaying geometry, images, and volumes on the screen.

Public domain modules are available via email or ftp. For more information, contact David Bennett via email at **avs@ncsc.org**, or send mail to the International AVS Center, Post Office Box 12889, 3021 Cornwallis Road, Research Triangle Park, NC 27709-2889. A limited number of modules are currently available during the testing phase, and full functionality should be available in spring 1992.

Visual Program Development

The combination of the Network Editor, AVS modules, and the graphical user

interface provides you with the premier visual programming environment. The Network Editor is a powerful, easy-to-use tool for quickly developing complex visualization processing networks or application programs. Simply by connecting together modules through mouse-driven, point-and-click operations, a nonprogrammer is capable of creating complex graphical applications without writing a single line of code.

The networks created in AVS are interactively reconfigurable and reusable by you. They may be saved and stored for future use and modification. Modules can be dynamically added, connected, and deleted from new or existing networks.

The AVS Network Editor supports rapid prototyping of visualization applications. For repeated production use, AVS also allows you to save a complete network with all the user-defined interactive controls and layout specifications as a complete, finished application.

The Extensible Visualization Environment

The capability to create additional modules for AVS — and even libraries of modules — enables you to extend the base capabilities of AVS for a specific application. The ease of integrating additional modules into AVS allows you to create complex scientific and engineering applications with little or no programming. You easily may incorporate your existing software programs as modules in AVS, integrating your in-house codes into the leading graphical visualization system resulting in a true integration of computation and visualization.

Distributed Visualization

AVS was designed from the outset to operate in distributed, multivendor, heterogeneous computing environments. The support for remote module execution allows modules to execute on computational servers in the network. X-terminal support allows AVS to display on eight-bit color X-terminals or workstations running the X-Window System. These distributed capabilities enable you to incorporate AVS into the networked, distributed computing model already established today. The two major architectural aspects that enable AVS to excel in a distributed environment are its modular architecture and its adherence to industry standards.

The Industry Standard for Visualization

AVS is now available on the major UNIX-based workstation and visualization systems. Vendors currently funding the International AVS Center include Advanced Visual Systems Inc., CONVEX Computer Corporation, Digital Equipment Corporation, Hewlett-Packard Company, IBM Corporation, SET Technology Corporation, Stardent Computer Corporation, and Wavetracer Incorporated. AVS has been ported to many additional platforms, and information on these ports is available from Advanced Visual Systems Inc.

AVS Features

AVS Geometry Viewer

The AVS Geometry Viewer gives you full control over a range of low-level hardware dependent graphics capabilities with simple menu-driven parameter selections. The AVS Geometry Viewer takes advantage of all of the graphics functionality of the computer system on which it is operating. On more advanced visualization hardware

systems, for example, you have full control of wireframe, Gouraud, or Phong shading; 16 individually controlled colored light sources, selectable as point, directional, or spot lights; surface properties such as specularity and transparency; and real-time 2-D and 3-D texture mapping and antialiasing.

AVS allows you to create multiple windows with different views of the same geometric object or simultaneously display multiple objects. Alphanumeric titles and labels in a variety of fonts can be added in screen or object space. You also can create scenes with a hierarchy of objects and manipulate them individually or as one or more groups. These scenes can be saved, preserving all viewing selections for later redisplay. You can create and save a sequence of images and cycle through the sequence to provide animated views of dynamic behavior, all in real time.

AVS Image Viewer

AVS provides a complete 2-D and 3-D image display capability, including real-time pan and zoom, region of interest operation, rotation and transformation, flipbook animation, and support for eight-bit, 24-bit, and floating-point images. Imaging filters include look-up-table operations such as contrast stretching, pseudocoloring, and histogram balancing, and data resizing operations such as interpolation, cropping, and sampling.

Volume Visualization

Through the use of predefined AVS networks, volume data may be read into and displayed by AVS and then viewed using the visualization techniques described in the Geometry and Image Viewers.

A new class of algorithms recently has evolved for visualizing volumetric data called direct volume visualization. An AVS module — tracer — implements a highly optimized ray tracing technique for 3-D volumes of cells. This means that it works as well for irregular field data and unstructured cell data as it does for regularly gridded voxel arrays. Since it is implemented in portable software, it runs efficiently on all AVS platforms. The module can create high-resolution, perspective views of colored and/or shaded volumetric data from any arbitrary orientation.

AVS Graph Viewer

The AVS Graph Viewer allows you to visualize and output your data in the form of 2-D graphs, line charts, bar charts, or contour plots. A menu interface allows for the selection of types of lines, plots, labeling, and annotation and for changing the scales and styles of grids and axes.

Layout Editor

The Layout Editor enables you to easily and completely define the user interface. The specific control parameters may be selected, and a custom interface may be defined, thus simplifying the default interface of a control panel for each module in a network. You also may quickly assign user interface peripherals to control parameter widgets. This is an extremely powerful productivity tool when tailoring AVS for specific end-user applications.

Data Probes

Coordinates and values at any given point instantly are obtained and displayed using data probes. The probe is positioned using the mouse, and, with a click of the mouse button, underlying data values instantly are printed on the screen.

AVS Data Types

The following data types are available in AVS.

- Field data
- Volume data: N-dimensional (scalar, vector)
- Geometry data: uniform, rectilinear, irregular
- Unstructured Cell Data (1-D, 2-D, 3-D): physical/space coordinates
- Chemistry data
- Image data
- User-defined data

Command Language Interpreter

The AVS Command Language Interpreter (CLI) is an ASCII language that can be used to drive most of the AVS system. This allows advanced users and application developers to type AVS commands or develop scripts containing AVS commands. This feature gives you the option of developing applications that communicate directly with the Command Processor without the Network Editor. It can be used directly by you or indirectly from an AVS module to create and modify AVS networks, query and change module parameters, and control the user interface layout.

Labeling and Annotation

The ability to label and annotate images is very important, especially when producing hard copy output for demonstration or publishing purposes. AVS provides you with this required capability in all of its Viewers.

AVS Animator

In the process of doing day-to-day research, a scientist frequently will need to visualize events that unfold through time. Animation can be a powerful tool in viewing these events and in presenting the results to colleagues. The AVS Animator is designed to allow the scientist to produce a high-quality animation in which control of object/camera/light placement and AVS module parameters can be handled on a per-frame basis. A sophisticated keyframe editor allows you to control these parameters. Output can be sent to the display or recorded on various types of video hardware.

The Data Viewer

The Data Viewer is an AVS application comprised of an intuitive, easy-to-learn, user-extensible, point-and-click interface to the most frequently used visualization techniques. This application was designed with you in mind for learning about AVS. The AVS Data Viewer is also a convenient shortcut for doing similar operations on a day-by-day basis and can be used as an aid to building AVS applications for use by others. The full power and flexibility of the AVS Network Editor is still available for users who wish to customize their applications.

Unstructured Cell Data

AVS support for Unstructured Cell Data (UCD) contains a subroutine function library and a collection of AVS modules for data input, filtering, and mapping operations. UCD is used with finite element problems most commonly found in the mechanical computer aided engineering and computational fluid

dynamics fields. The set of UCD modules includes isosurfaces, streamlines, 2-D slices, annotation, data probes, hedgehogs, and volume rendering.

Upstream Data

Upstream Data is used in the Data Probe and in direct manipulation of mapper modules. Upstream Data passing allows data to flow backwards through a network for the purpose of ascertaining the value and coordinate information at a particular point. This information is required for the mapper module (e.g., slice plane) or Data Probe to pass on in order to display or take other action with the information.

Module Library Management Tools

Included in AVS is a set of tools to enable the creation, addition, deletion, and modification to modules and entire libraries of modules.

Continued Evolution to Meet Future Visualization Needs

Enhancements and extensions to AVS continue, building on this industry-leading environment to meet future visualization needs. With its broad industry acceptance, AVS undoubtedly will continue to define the leading edge in visualization technology. To learn more about AVS, how to acquire AVS for your computing environment, and the newest enhancements, contact Advanced Visual Systems Inc., any AVS vendor, or the International AVS Center.

New Software Products

Chemistry Viewer

The new Chemistry Viewer is available from Polygen/Molecular Simulations in Sunnyvale, CA. Call Richard Hedges at 408-732-9090 for more information. This is one of the first commercial packages to be available in the AVS environment as an add-on application. This excellent software package has many features.

The Chemistry Viewer displays and manipulates the results from quantum mechanics and molecular mechanics calculations simply by clicking menu choices and on-screen virtual controls. Results can be displayed as surfaces or maps and then can take advantage of built-in volume visualization techniques to gain insight into properties derived from quantum wavefunctions. It supports the visualization of results from ab initio quantum chemistry and semi-empirical quantum chemistry (such as GAUSSIAN and MOPAC), molecular mechanics, and molecular dynamics.

More than 25 modules are provided with the Chemistry Viewer. The Molecular Editor Module creates molecular structures for input to other programs. Readers are available for Molecular Design Limited's MOL files and files from GAUSSIAN and MOPAC.

Full support is also available for Molecular Simulations' BGF file formats, which allows interactions with BIOGRAF, POLYGRAF, NMRgraf, and CERIUS programs. You can take wavefunctions from a variety of programs and calculate the volume of data corresponding to user defined molecular orbitals. A Periodic Table Module generates missing data (atomic radius,

atomic weight, atom color, etc.). The Connectivity Module generates a complete connection table for the user supplied molecule. Displaying your image can be accomplished in many ways, such as with the MO Surface Module that displays the isosurface of the molecular orbitals.

Maple V

The Maple V package can be obtained from Waterloo Maple Software in Waterloo, Ontario, Canada. Call 519-747-2373 for more information. Maple V is a computer algebra system that now supports AVS.

Maple V is used worldwide by mathematicians, engineers, and scientists for teaching, research, and commercial applications. Maple V frees users of the drudgery of complex symbolic and numeric computations, allowing them to creatively manipulate, investigate, and explore mathematical models.

You now can interactively conceptualize your mathematical models with the visualization capabilities of AVS. Maple V is renowned for its ability to solve a comprehensive range of problems as well as for its ease of use, speed, accuracy, and minimal memory requirements.

Maple V provides natural, easy-to-use operations for polynomial manipulations, differentiation, definite and indefinite integration, solutions of equations, and more. Its comprehensive library of more than 2000 high-level math functions is written in the Maple V programming language itself, allowing review and customization of the algorithms that are used.

Packages are available for linear algebra, number theory, statistics, group theory, linear optimization, differential forms, student calculus, and many others. To extend or redefine the built-in computational facilities, users can write their own Maple V programs in the same language used for coding the library.

VBASE

VBASE is a development of the Marconi Research Center in the United Kingdom; it is not yet productized. Requests for VBASE should be made to Dr. Rob Brooks, manager of the Marconi Research Center Space and Defense Research Laboratory, United Kingdom, at telephone number 44-245-73331.

Vector Database (VBASE) is a spatial single or multiple database system currently being integrated into AVS. It is a flexible and rapid means of storing and accessing features extracted from imagery and maps. VBASE has structured, easily updated databases for use by high-level algorithms and supports multiple databases.

The need to manipulate raster and vector data exists in any advanced image analysis or geographic information system. Raster data manipulation can be accomplished by many available techniques. The management of vector data, particularly very large vector data sets, is more difficult and is what VBASE does.

It is designed to hold 2-D vector data and associated numerical or textural information and allows rapid retrieval and update of this information. VBASE is designed to store regions, lines, points, and their relationships and classifications. It is capable of storing very large datasets

and accessing data via a tiled structure, which is transparent to the user and has a C library to manipulate objects.

The database can be created from software acting on pixel data or from topological maps containing fields, woods, water, roads, buildings, etc. Applications include feature extraction, data fusion, change detection, feedback to a lower level process, and model-based segmentation.

Crystal Viewer

Crystal Viewer is a package for 3-D data analysis. It can be obtained from Crystal Image Technologies in Huntsville, AL. Call 205-534-6555 for more information.

Crystal Viewer can be used for region of interest extraction, arithmetic operations on image and volume data, image and volume data segmentation, erosion and dilation of segmented images or volumes, ray tracing of volume data, probing of readouts of relative position and values, linear and nonlinear volume data fits, distance readouts within a defined line in a volume, and for profiles of values along interactively defined lines across an image or volume. This package is ideally suited for applications such as oil exploration, medical imaging, and other true 3-D volume data applications.

AVS Network News

The inaugural issue of ***AVS Network News*** was mailed out at the end of October 1991, and response was very positive from both users and vendors. The purpose of that issue was to provide basic information, and we hope you found it useful. The success of the quarterly magazine depends largely on AVS user participation.

Each full color issue features updates about AVS programs, articles from users about their work, tips for using AVS modules, an editorial section featuring user viewpoints, and other general AVS information. The magazine includes users' color images and solicits submissions from users interested in sharing information about their work. AVS vendors and independent software developers are encouraged to post product information in order to keep the AVS user community better informed about AVS developments.

Copies of ***AVS Network News*** may be obtained from your AVS vendor. For an annual subscription, complete and mail the magazine's subscription form, along with a $12 check or money order payable to AVS, to the International AVS Center, Post Office Box 12889, 3021 Cornwallis Road, Research Triangle Park, NC 27709-2889. In addition, a magazine subscription is included when joining the International AVS Users Group.

To have your work considered for inclusion in ***AVS Network News***, send your article describing your AVS application along with corresponding color slides to the preceding address. For more information, send mail to this same address or send email to **avs@ncsc.org**. Please participate and help make the ***AVS Network News*** your forum for sharing with the AVS user community.

Using AVS for Prototyping a Distributed Computational Molecular Biology and Genetics System

Center for High Performance Computing, The University of Texas System

figure 1

Abstract

This paper relates some of our recent experiences with AVS while developing a prototype implementation for the GenTools distributed Computational Molecular Biology & Genetics (CMBG) system. Our end applications typically require supercomputing resources and are run on a distributed system having geographically remote nodes. The user interface component runs on the user's local workstation. By using data transfer at a sufficiently high level of abstraction, both the supercomputer resources and the rendering capabilities of the local workstation are used optimally. Additional functionality necessary for dynamic message-routing is provided by a separate protocol library.

1. Introduction

Algorithms in CMBG are typically very resource intensive, both in terms of CPU usage and memory requirements. Consequently, biomedical researchers often have to utilize the services of a supercomputer (or some other high-performance computing architecture) for performing these calculations within a reasonable time frame. Supercomputing resources, however, are reasonably scarce and often geographically remote from the researcher. To make use of these computer services, researchers have to physically transfer their input files to the remote system, run all of their computation there, and then retrieve their results for subsequent interpretation.

This approach is extremely cumbersome to use. Furthermore, most applications lack the necessary functionality for good interaction or for mechanisms such as interapplication data exchange.

At the Center for High Performance Computing (CHPC), we are developing GenTools, a distributed CMBG environment, which will provide users with not only a transparent mechanism for utilizing distributed computing resources but also will have a sophisticated user interface component that permits a variety of editing/processing and data-interchange operations.

2. Requirements and Solutions

2.1 An Efficient Data Transfer Mechanism

Genetics applications are frequently capable of generating high-volume data. Therefore, result messages may be of the order of several megabytes. Most raw data in this environment consists of ASCII text strings. For optimizing transfers over slow networks, compression becomes a viable mechanism. We also anticipate the integration of related non-CMBG methods into the GenTools environment.

AVS provides a data-channel abstraction that transparently handles type-conversion and that also is optimized for data movement by using techniques like shared memory for modules communicating on the same machine. Other functionality such as compression and piecewise transmission of large data sets also may be added on as needed.

Another nice feature of the AVS system is the ability to specify module connectivities in a separate ASCII-text resource file. Modules, therefore, do not need to have this information hard-coded within, making it easier for network configuration managers to be set up independently.

For our GenTools application, we transmitted character-string data as 1-D scalar byte fields to overcome the 8K-string-length limitations of AVS 3.0. We also provided a bidirectional data transfer by creating a shadow feedback port for each connection using the AVS autoconnect feature.

2.2 Dynamic Packet Routing

One of the important features of the GenTools environment will be its interapplication data exchange mechanism. In a GenTools editing environment, a user should be able to progressively refine, edit, combine, or otherwise alter data created from earlier runs or from an input sequence file. Potentially, every component module within the system is capable of wanting to communicate with every other module in the system.

GenTools will grow rapidly in functionality as more and more applications are integrated. Users then will be provided with a loader mechanism that permits them to start up a selected subset of the entire GenTools connectivity graph. This kind of incremental network loading is useful in minimizing overall system overhead.

Also planned is a module migration mechanism. This would be used to evaluate current system loading and optimally migrate computation units to other less loaded machines in the distributed environment.

AVS currently has a static data-flow model. Module connectivities are defined when the network file is loaded, and a central flow executive arbitrates data transfer along these predefined channels. To permit the kind of dynamic connectivity GenTools requires, additional functionality was necessary.

A somewhat naive approach would be to have each module maintain a channel to every other module in the system. This means that for an n-module system the number of connections is O(n2). This rapidly becomes unmanageable with a growing system.

The approach we chose for this prototype was that of a centralized registry service to which all other modules in the system are connected. This reduces the number of connections to O(n). Each data set generated is encapsulated into a packet and prefixed with a header that describes the intended target for this data and the nature of the data within. A routing capability also is added to each module, allowing the module to look at these headers and to send the packet onwards to its final destination. The routing library can handle multicast packets and can accept specifications that cause the packet to visit a sequence of nodes in serial order. Type information can be used to invoke appropriate conversions.

figure 2

figure 3

2.3 User Interface Design

Many of the computation modules we wanted to provide within the GenTools system originally were designed with a batch-processing environment in mind. We found the AVS widget set and the interactive design layout editor very effective while designing user interfaces for the system. Additionally, in order to provide some molecular-level visualization capabilities to the system, we used the available PDB molecular display modules within AVS and within our user interface.

The ability to specify widget layout as an ASCII-text resource file was found to be very useful in keeping the definition of interface functionality and appearance separate.

3. Test Results and Conclusions

We tested the GenTools system over a T1 link at 1.5 Mb/sec, between the CHPC in Austin and the University of Texas Health Science Center in San Antonio — a distance of about 80 miles. Computation modules resided on a CRAY Y-MP in Austin, with the interface components being on Sun/Solbourne SparcStation 2's in San Antonio. Response time was found to be almost comparable to running the system locally on Ethernet.

This configuration allows the interrupt-driven user interface input mechanisms and the rendering operations to be relegated to the workstation, while the computationally intensive main application runs on the remote supercomputer. Data transfer occurs at a high level of abstraction, thereby minimizing overall network data flow and improving response time. A less optimal solution would be to use an X-Windows-based, client-server model. In this case, the data abstraction reduces and data volume increases, thereby degrading response times.

GenTools is currently under alpha test, and we are evaluating biouser feedback for enhancing the system. So far, the reaction has been very positive regarding both the feel of the interface and the response time for distributed processing. Currently under development is a hybrid data-routing mechanism that permits the creation of temporary high-speed links called wormholes. Wormholes are channels where normal type-conversion and routing are suspended. Wormhole connections will be created dynamically, depending on the amount of packet traffic between modules.

We would like to see certain additional functionality in future releases of the AVS system. A clean method for permitting network reconfiguration changes dynamically at run time would be one example. A module migration mechanism would nicely complement the functionality of this reconfiguration method. It would be desirable to be able to invoke type-specific methods for compression/decompression of data flowing across machine boundaries. In terms of the widget set, a multiline editable text widget is necessary. A more powerful programming interface to the widget set also would be very useful. When drag-and-drop becomes an accepted X11 standard, it would be nice to have this functionality integrated into the AVS system.

The AVS-based GenTools system has been extremely valuable to us in terms of establishing basic functionality and providing a working system within a very short time.

Interpolating Gridded Data from Scattered Data in AVS

by E. Wes Bethel, Information and Computing Sciences Division, Computing Resource Department, University of California at Berkeley

Abstract

In this article, we survey some of the issues involved in interpolating gridded data from scattered data and show several applications of these techniques in the context of a visualization project. This article serves as a reference guide for four AVS modules used to perform this type of interpolation. These modules are available for use by the user community and may be obtained from the module pool of the International AVS Center at the North Carolina Supercomputing Center.

1. Introduction

Scatter data refers to data that cannot be organized onto a grid or array structure. This type of data typically is thought of as being stored as a one-dimensional list (one computational dimension) but existing within an n-dimensional coordinate space. An example of scattered data is the elevation of the ground taken at arbitrary locations throughout some region. Each of these data values in this (one-dimensional) list has an associated spatial location (in three-space) and some scalar or vector value.

As only a limited number of visualization techniques are applicable to scatter data (e.g., bubbleviz/scatter dots), one of the goals of interpolating scatter data is to produce a gridded representation of this data that subsequently may be used by a wider class of visualization tools, such as isosurface extraction, false-colored slice planes, and so forth.

Following a description of some of the issues involved in making the scattered to gridded leap, we describe four modules that have been submitted to the International AVS Center. We will describe the use of these modules on several sample data and describe various pitfalls of each, along with some analysis about the behavior of the module parameters.

2. Issues

There are two broad classes of techniques that may be used to compute a gridded representation of scatter data. The first makes the assumption that the scattered data may be approximated, with a suitably small error, by a continuous function of some type. After determining the coefficients of the underlying approximating function, determining the data value at a given output grid location is a straightforward matter of substituting the grid coordinate into an equation, thus obtaining an output data value at that grid location.

A good example of this type of operation is the least-squares fitting of a function to data. A function of some type is assumed (linear, quadratic, etc.), and function coefficients are computed from the input data so as to minimize either global or local error. The primary drawback of this type of interpolation is that the results can be biased by outlying data values. The not-so-obvious drawback of this approach arises when the input data, in reality, can not be described by a continuous, differentiable function.

The second class of interpolation technique uses an approach in which all input points lying within a user-defined search radius are weighted for each output grid location (inversely proportional to distance from the output grid location), resulting in the output data value. The

figure 1

figure 2

figure 3a

figure 3b

primary drawback from this type of interpolation is that some trial-and-error is required to arrive at a suitable search radius for the interpolation algorithm. If no points lie within the search radius, then a value of unknown is output at that grid location (which may or may not be a problem).

3. Two-Variable Interpolation

In general, two-variable interpolation refers to the case in which we have a list of (x, y, scalar) values, and we wish to obtain a new data value at (X, Y) where X and Y are arbitrary. In both of the two-variable interpolating AVS modules, the user specifies an output grid in terms of minimum and maximum X and Y coordinates and the number of steps along each axis, resulting in a new rectilinear grid of values (AVS field).

The names of the two modules that may be used to perform two-variable interpolation are **scat 2d** and **bivar**. The module **scat 2d** assumes the data may be interpolated using a biquadratic function, while the **bivar** module uses a weighting algorithm of points within a search radius.

One of the parameters in the **bivar** module allows the user to set the size of the search radius. Increasing this value will cause more input data values to influence an output data value. The other parameter may be used to control the weighting itself. Increasing the value of this parameter will cause input points closer to the output and coordinate to have greater influence than those further away. There are no parameters under user control for the **scat 2d** module except those used to define the output grid location and coarseness.

4. Three-Variable Interpolation

Three-variable interpolation refers to the case in which the user has (x, y, z, scalar) values and wishes to obtain a new data value at (X, Y, Z) where the X, Y, and Z coordinates are arbitrary. In the case of the AVS modules, the user specifies the extents of the three-dimensional grid in terms of minimum and maximum in X, Y, and Z along with the number of steps along each axis.

The names of the two modules that may be used to perform three-variable interpolation are **scat 3d** and **trivar**. Like the two-variable interpolating modules, the **scat 3d** module assumes the data may be interpolated using a triquadratic function, while the **trivar** module uses a weighting algorithm.

The parameters for the **trivar** module are analogous to those used in the **bivar** module. The **scat 3d** module provides two extra parameters that allow the user to control the behavior of the underlying numerical code. These two parameters are exactly analogous to those used in the **bivar** and **trivar** modules.

5. Examples

5.1 Two-Variable Interpolation

In the first sequence of images, the scientists at Lawrence Berkeley Laboratory were studying the geologic formations beneath the site of the laboratory. The depth of each of several known formations is obtained by drilling a hole into the ground and observing the depth at which the formation was encountered. There are several such formations, but, for the sake of illustration, the formation used in these images is the elevation of the soil surface.

Figure 1 shows the scatter data that represents the elevation of the soil surface (visualized using scatter dots). The spheres in this image represent the height of the soil (i.e., altitude) in the region of interest. The visualization using spheres to depict the location of the soil surface at the sampling location serves to illustrate the scattered nature of the data. The goal in the next two images is to process the altitude data so as to provide a height map. This example is interesting because it is verified easily by empirical data.

Figure 2 shows the results of the use of **scat 2d**. Note the extrapolation — the areas of high altitude (indicated by the red color) that are not bounded by any input data points. These areas of extreme altitude, in fact, do not exist. The conclusion that can be drawn from this example is that the numerical code in **scat 2d** (and **scat 3d** as well) uses partial derivatives that are quite sensitive and can give erroneous results in areas of extrapolation.

Figures 3a and 3b show the results of using **bivar** in which the size of the search radius is varied. The search radius indicates the size of the region, over which points will have an influence at a given output grid location. Note the differences between the two figures. In Figure 3a, a search radius of 100 units is used. By referring to the labeled axes, it is possible to see how there can be gaps in the data. The gaps are locations on the output grid to which no input data points have any influence (because they are too far away). The **bivar** module allows the user to specify the value for the gaps or the unknown value. In this case, we have set the value of unknown to some minimum altitude. Hence, the gaps appear as ravines in the relief map. These ravines are artifacts.

figure 4

figure 5

figure 6

figure 7

By adjusting the search radius in Figure 3b up to 300, all of the ravines are eliminated, and the resulting relief map is in fact closer to reality than those in Figures 2 or 3a.

The difference between the **bivar** and **scat 2d**, from a user's point of view, is that the user must know something about the input data when using **bivar**, while no such knowledge is required to use **scat 2d**.

5.2 Three-Variable Interpolation

In this example, 46,000 earthquake events from 1967 to 1982 are used as input to an algorithm, developed at the Center for Computation Seismology at Lawrence Berkeley Laboratory and used to compute velocity values on a grid of points where each grid cell is roughly 5 by 5 degrees and 200 km in depth. The scientists were interested in studying large-scale structures within the earth's mantle.

A subset of the original data, that lying at the earth's surface, is depicted in Figure 4. In this image, the data points are false-color coded according to velocity value. Additionally, the continental boundaries are provided to give a frame of reference.

Like the previous example, we show the use of two different techniques for processing this data. Figure 5 results from using the **scat 3d** module, while Figure 6 results from using the **trivar** module. Figure 7 uses **trivar** as well but uses a smaller search radius. More detail or high-frequency components appear in Figure 7 than in Figure 6, as one would expect when less averaging is performed on the input data. Unlike the previous two-dimensional example, the results cannot be empirically evaluated. However, it is possible to see the extrapolation effects present in the **scat 3d** image that are not present in the **trivar** image in which the small search radius is used.

6. Conclusions

The conclusions that can be drawn from these examples are as follows. First, using one of **trivar** or **bivar**, along with a small search radius, will result in a gridded dataset that most closely approximates the input dataset but with a minimal amount of interpolation. Using a larger search radius will increase the number of points being averaged and will reduce any high-frequency component in the data. This low-frequency bias can be adjusted somewhat by the parameter called weight function. This parameter has not been explained in this article, but increasing this value causes points closer to the output grid location to have more influence than those farther away.

Second, the use of **scat 3d** and **scat 2d** may be able to give better results in those instances in which there are not very many data points.

As always, experimentation is encouraged.

7. Acknowledgement

This work was supported by the U. S. Department of Energy, Energy Research Division, under contract DE-AC03-76SF00098.

Visualizing Molecular Dynamics Simulations

by Charles Williams and Upul Obeysekare, Naval Research Laboratory, Research Computation Division, Visualization Laboratory, Washington, DC

A popular method to visualize time-dependant results from Monte Carlo and molecular dynamics simulations has been to use the Macatom program on the Macintosh computer. A typical user would generate an ASCII Macatom file with radius and color information for each atom coordinate along with a flag to separate time-dependant frames. Although this method is inexpensive, the speed of the Macintosh graphics may not be appropriate for large simulations. Also, animating a time series is not trivial.

figure 1

figure 2

figure 3

figure 4

In response to requests from scientists at the Naval Research Laboratory (NRL), the Visualization Laboratory has developed a method to read Macatom files for visualization using AVS. As shown in Figure 1, the AVS network reads the Macatom file and outputs 2-D field data with atom coordinate, color, and radius, where the first dimension of the field corresponds to the frame numbers and the second dimension to the atom numbers. The orthogonal slicer module extracts individual frames from the field data and output to the scatter dots module. This module converts the atom data into spheres in 3-D with the specified colors and radius. Render geometry and display pixmap modules convert the geometric information into pixmaps. Using the animate integer module, one can change the orthogonal slice and animate different frames.

Shown here are four frames at different time steps from a molecular dynamics simulation of detonation in a solid high explosive with crystal defects. Each frame consists of approximately 8000 atoms. This data was generously supplied by Lee Phillips from the Laboratory of Computational Physics and Fluid Dynamics at NRL.

AVS Version 4

The International AVS Center intends to keep you updated on what's new in AVS. Detailed information is available from Advanced Visual Systems Inc. and from your local AVS vendor representative.

Animation Application

AVS has many new features, but the one asked for most frequently is the AVS Animation Application. This includes the AVS Animator module, modules for reading and writing sequences to/from disk, and modules for recording image sequences on videotape.

The AVS Animator has VCR-like controls for viewing animated networks. Its keyframe editor allows you to establish snapshots of AVS networks with control over entities to be animated on a frame-by-frame basis.

You also can specify the number of frames to be generated automatically between keyframes and the algorithm to be used for interpolation. This makes it very easy to generate highly sophisticated animations of complex networks with multiple parameters, transformations, and attributes varying continuously from frame to frame.

Module Generator

Another exciting feature is the AVS Module Generator, which generates skeleton code for AVS modules. Using interactive, graphical widgets, users can specify what types of input and output ports as well as parameters they want their new module to have.

Source code can be created in either C or FORTRAN, a Makefile can be generated, and you can create man pages. You can edit, compile, load, and debug from the Module Generator. It is an easy way to learn to write modules since the mechanics of AVS module code structure and library calls are generated automatically.

Data Viewer

A new AVS Data Viewer is available for those who don't want to be bothered with the complexities of AVS. This is a simplified user interface of the most frequently used visualization techniques. It has fixed menus and windows and automatically performs the module and network construction and editing that requires a higher learning curve when first using AVS. AVS Data Viewer uses a preconstructed set of the most commonly used networks for structured and unstructured data visualization.

ADIA

One past concern about AVS was the importing of foreign data into the system. The new AVS Data Interchange Application (ADIA) consists of two modules that assist the AVS user in importing foreign data. The "file descriptor" module uses a file-in-the-form interface paradigm that allows you to specify what your external data file format looks like.

Once an external file format has been described, the data form can be saved for later use. Since data forms are simple ASCII files, it is easy for users to exchange these forms. The "data dictionary" module allows users to import/export files that have been described by the data forms generated in the file descriptor module.

Geometry Viewer

The Geometry Viewer has many enhancements. One is the limited ability to access hierarchical Geometry Viewer flipbooks in the Action submenu. You now can use Step Forward/Backward, Continuous, Bounce, and Delete Current Frame buttons when operating on an object containing a "child" object that is a flipbook. The operation will be performed on all of the children that are flipbooks.

The new AVS software renderer is to be used on nonhardware accelerated graphics platforms and includes many advanced features such as texture mapping, volume rendering, arbitrary clip planes, and transparency. The new Geometry Viewer module combines the functionality of render geometry and display pixmap. It also allows an AVS image output port that provides network access to images contained in the geometry window and removes the need for the pixmap data type.

This unifies the handling of geometric images on all AVS platforms and allows true-color image output on non-true-color X-servers for animation, image storage, and postscript. The camera transformation editor provides an alternate method for transforming the camera. You now can control the view parameters — from, up, at, angle, front clip plane, back clip plane, and depth cueing.

Graph Viewer

The AVS Graph Viewer now supports legends, and you have control over location within plot window as well as coloring and font types. The Command Line Interpreter (CLI) has been added for the Graph Viewer. In addition to standard integer and floating-point numbers, you now can read in scientific notation numbers from external ASCII data files.

In AVS 3.0, plot dimensions always were normalized to fit the data. You now can choose whether or not to normalize the dimensions (which is useful for creating strip charts). Now, with an image output port, images can be sent to display image or output postscript.

Network Editor

The new AVS Network Editor has two exciting enhancements. The available module libraries now are represented by a set of radio buttons on top of the Network Editor panel, and you can change the names of columns in a module library for customized sets. You also can move a module from one category to another.

The other enhancement, and perhaps one of the most requested by users, is the hierarchical module approach. You now can group a "subnetwork" consisting of modules, connections, and widget layout into a higher level macromodule. This module then can be instantiated like any regular module. Using an arbitrary number of module hierarchy levels allows you to construct extremely complex AVS networks while maintaining a high degree of network organization.

New Modules

New modules include clip geometry, colorize geometry, geometry viewer (software renderer), generate axes, set view, extract graph, excavate brick, image to postscript, unstructured cell data (ucd) print, ucd extract scalar, ucd extract

vector, ucd vector magnitude, and ucd cell to node. Enhanced or improved modules include tracer, streamlines, downsize, volume bounds, field math, print field, compare field, output postscript, write image, write volume, compute gradient, generate histogram, read ucd, write ucd, ucd to geom, ucd anno, ucd probe, ucd crop, ucd hog, ucd streamline, ucd extract, ucd offset, ucd iso, and field to ucd.

Flow Executive

The AVS Flow Executive has been improved by adding direct module-to-module communication. Currently, all modules communicate through the AVS Kernel. This can be wasteful of memory because the Kernel has a copy of all data traveling between modules. It also can waste time because the data is serialized and deserialized twice instead of once.

The direct module-to-module communication support allows modules to communicate directly, bypassing the XDR translation and using shared memory when available. The socket to the Kernel is now called the "control" socket to distinguish it from the new "data" socket. The "data" socket will listen for connection requests and process commands containing data for input ports and parameter ports. During initialization, the socket port number will be sent to the Kernel and stored with other module information. A single data socket can be shared by many modules in the same process.

Miscellaneous

Lastly, several memory allocation tools have been provided. Dynamic memory allocation is always a problem. It is often difficult using conventional C programming tools to track writes beyond allocated memory, use of memory after being freed, making incorrect assumptions that malloc has cleared memory, failure to check for NULL returned form malloc, specifying single argument to calloc, or failing to free memory (a memory leak).

AVS 4 provides a layer of functions between the application and the system memory allocation routines in a library called MEM routines. If you include this special new header, all memory allocation routines (malloc, free, realloc, calloc) will be mapped to the new MEM routines and provide the developer with a means of tracking memory allocation problems.

AVS Newsgroup Update

The AVS newsgroup, **comp.graphics.avs**, provides a means for communicating anything related to AVS. AVS users' comments are encouraged in order to help shape the directions of the International AVS Center. The need for subset newsgroups for specific fields is anticipated as the AVS community grows. If you have a problem that is not being addressed, please send email to **avs@ncsc.org** and we will try to assist you in finding a resolution.

Input from attendees at the first international AVS User Group conference held in February will direct future operations of this newsgroup. If you were unable to come to the conference but wish to have input into shaping this direction, send email to **avs@ncsc.org**. We will post a summary of these comments to the newsgroup after the conference.

Module Submission Criteria

We encourage modules of all types. We are not looking for perfect productizable code but rather code that is used everyday and would be useful to users. Several users have indicated that their code was not good enough or would not be useful to the AVS user community. This is not true.

We have many different levels of users — from the total novice to the experienced professional — and code is needed for all levels. All we require is a release form (which can be adapted to your special circumstance), source code, a Makefile, and either a README file or man page. Data, networks, etc. are strictly optional.

Release form — A release form (enclosed in this magazine), signed by the individual submitting a code, **must** be mailed to the International AVS Center before the code will be made available to the AVS user community. This signed form indicates that the individual is authorized to submit the module into public domain and releases the Center from any liability from copyright violation. Individuals who submit modules prior to completing a release form will be sent an electronic version of the form for completion.

Source file and Makefile — Source code and Makefile **must** accompany all modules submitted (unless the module is data only). We encourage extensive commenting and ask that port-specific lines be so commented.

Module description — A detailed description of the module **must** be included with its submission. This may be in the form of a man page or ASCII text format (as in a README file). Module description is one of the most important criteria for submission. Please note if multiple modules are in one source code file. The International AVS Center staff will rewrite the documentation as necessary.

Data — If your module uses a nonstandard data format, you **must** include an example (not necessarily real) of that data. Explanations of how to read the data should be placed in the required manual or README file.

Miscellaneous — You are **requested** to provide sample network and scripts when appropriate.

To submit modules, *ftp* to **avs.ncsc.org** and *cd* to the **SUBMIT** directory. You will be prompted in how to create a directory. You will receive a message when the directory has been created and prompted on how to *cd* to that directory. The directory itself will be invisible so that others may not inadvertently copy over your files. When you are in your directory, use standard ftp protocol to submit your modules and associated files. Information on using ftp for basic submission and retrieval can be obtained by sending a request to **avs@ncsc.org**.

You also can submit modules and associated files by email to **avs@ncsc.org** and by tape (tape will be returned if requested) to the International AVS Center, Post Office Box 12889, 3021 Cornwallis Road, Research Triangle Park, NC 27709-2889.

The following disclaimer will be placed with all files donated to the International AVS Center.

The International AVS Center Warranty Disclaimer

This AVS module and the files associated with it are distributed free of charge. They are placed in public domain, and permission is granted for anyone to use, duplicate, modify, and redistribute them unless otherwise noted. Some modules may be copyrighted. You agree to abide by the conditions also included in the AVS Licensing Agreement, Version 1.0, located in each module directory.

The International AVS Center, MCNC, the AVS Consortium, or the individual submitting the module and files associated with said module provide absolutely **NO WARRANTY OF ANY KIND** with respect to this software. Any risk as to the quality and performance of this software is entirely with the user. **IN NO EVENT WILL THE INTERNATIONAL AVS CENTER, MCNC, THE AVS CONSORTIUM, OR THE INDIVIDUAL SUBMITTING THE MODULE AND FILES ASSOCIATED WITH SAID MODULE BE LIABLE TO ANYONE FOR ANY DAMAGES ARISING FROM THE USE OF THIS MODULE OR ASSOCIATED FILES INCLUDING, WITHOUT LIMITATION, DAMAGES RESULTING FROM LOST DATA OR LOST PROFITS, OR ANY SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES.**

If you wish to contribute toward the improvement, modification, or general performance of this module, please send us your comments, such as why you like or dislike the module, how you use it, and, most importantly, how the module helps your work. Email your comments, as well as any bug reports, to **avs@ncsc.org**.

Obtaining Modules

Modules may be obtained by two basic methods. The first is the standard ftp protocol. When you *ftp* to **avs.ncsc.org**, a menu will come up that will guide you in using the ftp site and will point you to the **AVS_README** and the **AVS_LICENSE** files. The AVS license agreement must accompany all modules obtained and is self-explanatory.

The **AVS_README** file shows the directory structure listing. The **AVS_FLATLINE** file is in flatline database format delimited by a colon <:> and will fit into most current database programs. Instructions on the use of the **AVS_FLATLINE** file are also in the **AVS_README** file. This flatline file will not be available until later this year.

When you login to the AVS directory at **avs.ncsc.org**, you will see subdirectories called **DATA**, **FILTERS**, **MAPPERS**, **RENDERERS**, **MISC**, **SAMPLE_DATA**, and **SUBMIT**. Some modules have multiple purposes and will be placed in the most appropriate directory. **SAMPLE_DATA** will be for data that is donated but has no associated modules.

When you *cd* to the subdirectory, you will see many Makefiles, such as **make.titan** for each port of the module and its associated files. This will include source code, help files, **README** files, scripts, sample networks, etc. This is a change from the test procedure originally introduced. User input determined that it was too difficult to traverse through all the directories and that this is the best alternative.

The second method for obtaining modules is through standard email. When you send an email message to **avsemail@ncsc.org**, you receive an

automated listing of all available modules along with a request form. When you return a request form to **avsorder@ncsc.org**, you receive the modules and associated files (subject to size considerations) through normal email channels. Full information on this procedure is sent with the initial automated mail response.

While this process presently is not difficult, it quickly will entail a large maze of directories. By early to mid-1992, we expect to have at least 1000 modules. Therefore, we will provide a selection of X interfaces and a command line program using **WAIS** protocol by the middle of 1992. This is a hypertext search-and-query program and will be modified for AVS. Later, an **AVS_WAIS_README** file will explain how to use this method.

When you *ftp* into the International AVS Center, you will see a new directory called **WAIS** with a code to download and compile for a variety of machines and interfaces. This will be extremely useful as the number of modules and information grows. You will be able to type a sentence in standard English language (e.g., <Show me all the filters that are related to molecular modeling.>) and receive a listing of everything relating to that particular request.

You then can read the man pages or look at the source code and decide what you want. You will be able to click on a window and have the modules and their associated files or any subset *ftp*'d to you automatically. You also will be able to use the **WAIS** interfaces for exploring information from AVS newsgroups and other AVS related topics.

If you do not have network capability, you may request a tape via U.S. mail. Send a tape and a $10 postage-and-handling fee along with your request. You should state the vendor, machine type, operating system, and tape density. We will then send you all available files for that architecture, such as DEC5000, Ultrix 4.1.

If you do not provide a tape with your order, you must include tape cost. Tapes will not be available from the Center until its porting facility is completed and tape needs and module-to-tape downloading times have been determined. As additional order information becomes available, it will be posted to the AVS newsgroup, included in ***AVS Network News***, and made available through your AVS vendor.

Special requests for tapes of modules can be made to **avs@ncsc.org** and will be handled on an individual basis for now. An automated procedure to be completed by spring 1992 will accelerate this process.

The Center eventually will provide a full version of **WAIS** as a repository module and a number of service modules such as **MAKE A SLIDE** or **MAKE A TRANSPARENCY**. These service modules will be available for a fee to members of the International AVS Users Group and will allow users to obtain a slide, transparency, color photo, and eventually a video over the net.

These service modules will be provided initially by the International AVS Center but after a reasonable period will be turned over to commercial companies. This will be especially useful to those whose budget prevents obtaining these services.

The North Carolina Supercomputing Center (NCSC), site of the International AVS Center, began operations September 1, 1989, with the mission to promote computational science education and research in North Carolina institutions and to foster technology-based economic development through the application of high-performance computing to industrial problems. A division of MCNC, NCSC is located in Research Triangle Park, North Carolina.

The proximity of a number of high-tech oriented companies, major research universities, medical research centers, and state government facilities provides a wide variety of opportunities for NCSC to fulfill its mission. CONCERT, the outstanding state-supported communications network, which connects NCSC to the Internet, makes NCSC's resources available to various institutions and corporations.

NCSC's Visualization Lab enables scientists and engineers to develop graphical representations of complex scientific processes or data. An intensive effort is under way to build a flexible, statewide visualization environment with transparent remote access that will leverage existing tools from other supercomputer centers as well as locally developed enhancements.

To support the extensive data storage requirements of its users, NCSC is developing a virtual mass storage system that potentially will increase online storage resources to trillions of bytes (terabytes) and will be accessible at very high speeds from any NCSC computer system. All NCSC computers, including visualization workstations, will be connected on a very high speed internal network built around ANSI standard 800 megabit-per-second high-performance parallel interface channels.

The International AVS Center