JOURNAL OF STRUCTURAL BIOLOGY **116,** 56–60 (1996)
ARTICLE NO. 0010

IVE (Image Visualization Environment): A Software Platform for All Three-Dimensional Microscopy Applications

HANS CHEN, DIANA D. HUGHES, TAN-AN CHAN, JOHN W. SEDAT, AND DAVID A. AGARD[1]

Department of Biochemistry and Biophysics and the Howard Hughes Medical Institute, University of California at San Francisco, San Francisco, California 94143-0448

Received June 7, 1995, and in revised form June 23, 1995

IVE (Image Visualization Environment) is a software platform designed from the outset to handle all aspects of modern computerized multidimensional microscopy. This platform provides users with an execution environment in which 5D data (XYZ, wavelength, and time) can be easily manipulated for the purpose of data collection, processing, display, and analysis. During the entire process, powerful data display functions are readily available for extracting complicated three-dimensional information through data visualization. By employing both the shared memory and multitasking features of the UNIX operation system, individual functions can be implemented as separate programs, and multiple programs can access the same data pool simultaneously. This enables users to combine the functionalities of different programs to facilitate each unique data analysis task. Furthermore, by defining an appropriate program execution model, commonly shared functional components such as data display, data I/O and user interface, etc. can be implemented using simple IVE library calls. This dramatically reduces the program development time and ensures consistency throughout the entire software system. As a result, users can quickly master the microscopy software system and new functions can be easily integrated, as different functional requirements arise for different research projects.

INTRODUCTION

Although various forms of 3D imaging technology have proven to be powerful methods for examining biological specimens, their application to solving real biological problems has been more limited than the technology merits. These limitations arise mainly from the lack of suitable software tools for analyzing the resultant complex 3D data sets. Without proper tools, 3D data sets cannot provide much more information than the conventional 2D data. In the past, because of hardware constraints, only large-scale computers had sufficient computational power for processing large 3D data sets. As a consequence, most of the data processing jobs were executed in batch mode, which resulted in a long delay between processing steps. This, in combination with the limited data display capability, greatly impaired the effectiveness of these software tools.

Due to the recent dramatic advances in computer technology, powerful desktop graphic workstations can now deliver processing performance comparable to that of past supercomputers, and their advanced graphic capabilities are well suited for rendering 3D data. Moreover, the trend for this revolutionary advance in computer technology is expected to continue well into the next century. The improved hardware performance makes it possible to interactively manipulate multidimensional data sets which only a few years ago would have required many hours of calculation. Not surprisingly, there is a pressing need to design a software system which can take advantage of these hardware advances to provide the desperately needed software tools for data analysis. Crucial is the ability to rapidly and efficiently incorporate new concepts within the software system. This allows the capabilities of the system to keep pace with the ever-expanding needs arising from tackling new scientific problems.

In this paper, we will present such a software system, the Image Visualization Environment (IVE), which is designed to be the foundation for all of our 3D microscopy software development. We will describe the basic programming tools available and the application programs which have been developed on this platform thus far.

IVE: IMAGE VISUALIZATION ENVIRONMENT

The first step in designing our software system was to identify an execution model which is applica-

[1] To whom correspondence should be addressed. Fax: (415)476-1902.

1047-8477/96 $18.00

ble to every 3D microscopy application. According to this model, there are three major components in every software task cycle: user interface, function modules, and display. A user initiates a cycle by selecting a function with a set of parameters, to process a multidimensional data set. The function module then carries out its processing task based on these parameters, generating a new data set. The new data set is sent to the display system allowing the user to view the result. This concludes one complete task cycle. Depending on the result, the user will repeat the cycle until the desired information is obtained.

Based on this model, we concluded that the user interface, data I/O, and data display are the commonly shared components needed by each microscopy application. By providing the Widget Menu Library (WML), Image Window Library (IWL), and a data display system, IVE addresses each of the components, respectively. As a result, software development is greatly simplified and consistency in areas such as user interface style, data format, etc. is well maintained among all IVE programs.

Image Window Library

The IWL is central to the IVE architecture, with its major role being data management. Acting as an interface layer between application programs and the data pool, IWL manages data in both permanent disk files and temporary computer memory. Each data set is assigned a header structure which describes all of the information relevant to the entire data set. The header structure is capable of describing data of up to five dimensions (XYZ,time,wavelength) with data being stored as a stack of 2D images. An optional extended header can be added to record additional information specific to each 2D image. For data stored on computer disk, the entire data set is stored in a single file and its associated header structure and optional extended header structure are recorded at the beginning of this file, followed by the actual data. This design allows for rapid access to related data information without scanning through the entire data file. Based on this format, a set of convenient routines have been developed within the IWL, to facilitate all data I/O functions in both batch and interactive IVE application programs. The file format chosen was an extension of a file system developed at the MRC Laboratory of Molecular Biology in the early 1980s to handle both multidimensional image and X-ray data (Chen *et al.*, 1989).

By adopting the same data format to organize data stored in memory, IVE programs can access data in memory as if they were stored in disk files and they can be referenced by their pseudo-filenames (for example window numbers). Thus, without having to do anything special, any program can work either with files from disk or with data displayed on the screen. The advantage of storing data in memory is that it allows for fast data access which is crucial to interactive applications. By using shared memory to store data in IWL, multiple programs can share data information, as opposed to each program having to make its own copy, This allows for the most efficient use of memory space, which is especially important when dealing with multiple 3D, 4D, or 5D files. As computer speed becomes faster and memory size becomes larger, there will be more interactive programs and they will be able to deal with larger data sets. This will greatly increase users' ability to analyze complicated 3D data. With the IWL's parallel data I/O features, many of today's batch programs can become interactive programs in the future without any code modification.

Data Display: Monitor Process

Data visualization is a powerful technique for extracting useful information from 3D microscopy data sets. For every data set stored in shared memory, a display program, called "Monitor" is created to handle all of the data display functions. Each Monitor program manages its own display window and its display method is dictated by a set of display attributes. Frequently manipulated attributes, such as zoom, section, offset, window size, etc., can be easily set by the user with function buttons along the window border. The complete set of display attributes can also be modified by other IVE programs through simple IWL function calls. These library calls supply each application with the ability to display and work with images without complicated graphic programming. In general, the Monitor program displays data as gray-level images in its window. The mapping function between data value and screen pixel intensity can be either linear or exponential. An easy to use "Scale" program has been developed to provide users with a fast graphical way to experiment with different settings for scaling parameters. To display multiple wavelength data sets, the Monitor provides an image overlay function which shows multiple wavelength data in different colors in the same window. Figure 1 shows an example of overlaid images.

This architecture creates a flexible windowing environment in which multiple data sets can be displayed simultaneously, each with its own Monitor process. This is key in data analysis because it allows for visual comparison between original data sets with the different stages of processing. Moreover, any IVE application program has the power to work with multiple data sets at the same time, so that comparative measurements as well as visual conclusions can be made.

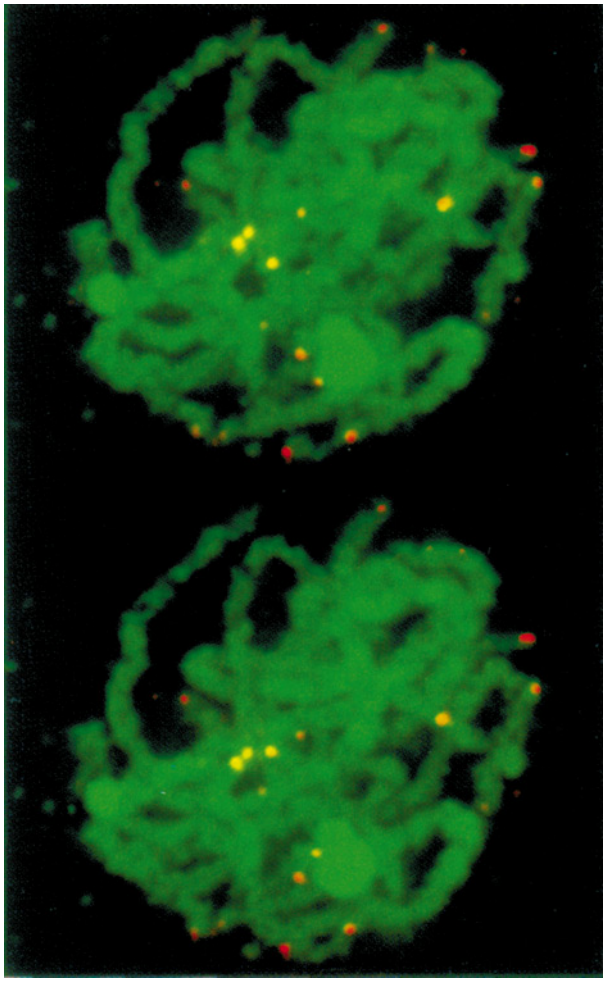

FIG. 1. A stereo pair from volume-rendered projections of a multiple-wavelength 3D data set shows maize pachytene chromosomes (DAPI, green) following *in situ* hybridization to telomeres (FITC, orange) and the 5S rDNA gene cluster (rhodamine, yellow). Original data are courtesy of Dr. Hank W. Bass (Department of Molecular and Cell Biology, U.C. Berkeley CA).

Graphical User Interface—WML

The next step in our project was to develop a library of high-level graphical user interface routines. The library first provides a rapid way to create graphical menus for IVE programs. These menus simplify the input of parameters for interactive as well as batch programs but are by no means required for either type of program to work with the IWL. Second, WML provides convenient routines which allow interactive programs to communicate with the IWL's Monitor programs so that applications can be sensitive to the currently displayed section of their input data. Finally, an "information" widget is made available so that programs can offer on line context sensitive help to users.

The WML is developed on top of the standard Motif/X libraries. With WML, no prior knowledge of the X window or Motif programming is required in order to create menus for an application. WML provides a basic set of menu building blocks (widgets) and guides application programs toward a common user interface style. Each widget can be linked to a callback function. When a user interacts with the widget, an event will be generated which signals the activation of that callback function. In this way, a general event handler controls the flow of the program.

In addition to menu creation and program flow control, WML provides ways for IVE programs to request events from the Monitor programs. In general, these events are a result of user interaction with the Monitor. There are two types of Monitor events. The first type is generated by any change in the displayed image. The other type is generated by user-controlled mouse activities within a Monitor's display window. When either type of event occurs, the Monitor will forward the event to all IVE programs which have requested events from it. This makes it possible for IVE programs to support data picking, region selection, and data display sensitive functions.

To assist IVE programs in providing on line help to users, an on-line help daemon was developed. WML has a specific "information" widget that allows an IVE program to summon this help daemon and with a help key that points to a specific help subject. Once inside the help program, the user can browse through the entire help directory. Adding new help information to the system involves writing a text file according to a very simple format and placing that file in the help directory.

IVE SOFTWARE APPLICATIONS

Based on the IVE platform, we have developed the following software packages for light and electron microscopy: Resolve3D, QPM, EMACT, EMCAT, and Priism. Resolve3D is a data collection package for the Sedat/Agard wide-field optical microscope system. QPM consists a set of processing programs which offers functions such as light intensity normalization, bleaching correction, deconvolution, etc. (Chen *et al.*, 1996; Agard *et al.*, 1989) for data collected with Resolve3D. EMACT and EMCAT are electron microscope tomography data collection and reconstruction packages. These two packages will be described in detail in an accompanying paper. Here, we will focus our discussion on Priism which is a general purpose data analysis package.

Priism

An image analysis package, Priism, has been developed using the IVE platform. This package includes various programs which address issues such

as image enhancement, segmentation, and quantitative analysis. Special display features such as volume rendering and digital movie displays are handled here as well. There are two major objectives in designing Priism: The first one is to create an interactive software environment in which data visualization techniques can be easily employed throughout the data analysis process. Second, new functions should be easily integrated to address special biological questions. These two goals can be easily accomplished by using features implemented in both IWL and WML. Moreover, a data processing program template has been created to greatly speed the process of developing new Priism functions.

Special display features. Volume rendering (Dreibin *et al.*, 1988) is a powerful rendering method for showing 3D information in volumetric data sets. An on-line volume rendering function has been implemented in Priism. It offers easy control of viewing region and angle and offers several methods for rendering. Combined with the data blending function, in Priism, users can volume render a 4D (XYZ and Wavelength) data set with each wavelength in its own selected color. This provides an excellent way to visualize the spatial arrangement of multiple structures in the same specimen. Alternatively, users can precompute a stack of volume-rendered images which correspond to a series of projection views of the same 3D structure in different angles of a fixed axis. Then, the movie function in Priism can be used to display them in a cyclic manner. This rocking back and forth of the volume provides a feel for the 3D structure which cannot be comprehended by examining individual 2D cross sections.

Feature enhancement. In analyzing complicated 3D data sets, it is very common that only a small percentage of the entire data set actually contains interesting features. To facilitate the visualization of the key features, many digital processing algorithms that have been developed to help focus in on the regions of interest. Common methods such as contrast stretching, histogram normalization, high pass/low pass/band pass filters, edge enhancement (Castleman, 1979), local contrast enhancement (Andrews *et al.*, 1972), etc. are all supported in Priism. In general, users try different functions with different sets of parameters before achieving the desired result. Priism's interactive environment makes this manipulation simple. Moreover, because of IWL's architecture, users can chain different processing functions together and pipe the output of one function to the input of the next function. By keeping input and output data in IWL memory, each step of processing can be easily examined through display functionalities provided by the Monitor program.

Segmentation. The first step in segmentation is to separate the interesting features in the data from the background. This can be done by selecting a threshold, if the data features have been properly enhanced. The second step is to identify individual structural objects within the data. In Priism, there are both manual and automatic methods for object identification. One technique we've adopted is to isolate features in two-dimensional sections first and then associate these features with those in neighboring sections, thus identifying three-dimensional objects. Both automatic and manual methods for 2D feature recognition result in a list of polygons which surround the feature. The volume building program takes the set of 2D polygons and associates them based on a selected criterion. Once the initial association is made, the user is shown graphically which features were put together to create an object. If any of this was done incorrectly, the user can manually correct the result. Another automatic modeling program (Houtsmuller *et al.*, 1992) has been implemented which traces through the center of objects by following the path of highest intensity in three dimensions. This program works well in tracing simple isolated tubular shape objects.

Although automatic methods can save users' time and energy, many structures are still too complicated to decipher automatically. Therefore, an interactive modeling program, which allows users to trace 3D structures manually, has been developed. This program allows users to model with multiple windows, for example, with each containing the same data set, or some region thereof, but shown in different orientations. Utilizing the IWL's display features, the modeling program automatically updates each of the windows to the section containing a selected focus point in the data set. This point can be set by using an on screen cursor to pick data displayed in any window. This capability enables users to resolve complicated local structure by viewing it from multiple angles simultaneously. To visually link the model to the data, the 3D model is drawn to each display window and then clipped to show only the part of the model landing in the currently displayed section or volume. The complete model is always shown in a separate window in which users have complete control over viewing parameters. All of the segmentation methods write out a file to describe 3D objects that can be read in and viewed by the manual modeling program.

Quantitative analysis/data measurement. Once the objects under study are completely described, the next step is to measure the desired object properties. A data measurement function is provided in Priism for measuring properties such as volume, integrated intensity, center of mass, distance, etc. and the result can be saved to a text file for plotting.

Implementation

The image file format used in IWL is an extended version of the MRC format which supports unsigned byte, short integer, float, and complex data in both VMS and IEEE binary format. We've also adopted the original MRC's IMSUB data I/O routines as a base for IWL's data I/O routines and added new display-related functions following similar conventions. All routines in IWL and WML can be called from either C or FORTRAN programs. The platform is currently written to run on the entire line of SGI (Mountain View, CA) workstations. The minimum hardware configuration for IVE is 32 MB memory, 8-bit graphics, and 512 MB of disk space. Because the Monitor program uses SGI specific graphic library (GL) to draw graphics and display images, IWL cannot be easily ported over to other hardware platforms. To remedy this problem, in the near feature we will switch to OpenGL which is supported on other manufacturers' UNIX workstations.

DISCUSSION AND FUTURE DEVELOPMENT

This project, developing our second-generation microscopy software system, was begun 4 years ago to take advantage of the recent dramatic advances in computer technology. We chose Silicon Graphics workstations as the hardware platform because of its balanced performance between computation and graphics. These capabilities have enabled us to explore more sophisticated interactive data analysis algorithms, work with larger data sets, and greatly reduce data processing time. In addition, we can run multiple processes in a cooperative manner, further enhancing the users' ability to deal with complicated data sets. With the design and implementation of IVE we have successfully created an imaging system which offers a wide range of multidimensional microscopy software applications and allows new functions to be easily integrated to address new biological problems. Already, we have been able to adopt this powerful platform to aid in the display and analysis of multidimensional X-ray crystallographic and NMR spectroscopic data.

In the future, in addition to continuing our effort in developing software tools for analyzing 3D data, we will also focus on improving the IVE's capability to link functions together to create super functions and adopt client server architecture to allow functions running on different computers to take full advantage of the computer hardware.

For more information about IVE and its availability please e-mail Hans Chen at hans@msg.ucsf.edu.

This work was funded by the Howard Hughes Medical Institute and NIH Research Grants GM-25101 (J.W.S.) and GM-31627 (D.A.A.).

REFERENCES

Agard, D. A., Hiraoka, Y., Shaw, P., and Sedat, J. W. (1989) Microscopy in three dimensions, *Methods Cell Biol.* **30,** 353–377.

Andrews, H. C., Tescher, A. G., and Kruger, R. P. (1972) Image processing by digital computer, *IEEE Spectrum* **9,** 20–32.

Castleman, K. R. (1979) Digital Image Processing, Englewood Cliffs, NJ: Prentice–Hall.

Chen, H., Swedlow, J. R., Grote, M., Sedat, J. W., and Agard, D. A. (1995) Handbook of Biological Confocal Microscopy, 2nd ed., Pawley, J. B. (Ed.), pp. 197–209, Plenum, New York.

Dreibin, R. A., Carpenter, L., and Hanranhan, P. (1988) Volume rendering, *Comput. Graphics* **22,** 65–75.

Houtsmuller, A. B., Smeulders, A. W. M., van der Vroot, H. T. M., Oud, J. L., and Nanninga, N. (1992) The homing cursor: a tool for three-dimensional chromosome analysis, *Cytometry* **14,** 501–509.