Editors: Theresa-Marie Rhyne and Lloyd Treinish

Visualizing Visualizations

User Interfaces for Managing and Exploring Scientific Visualization Data

Kwan-Liu Ma

University of California at Davis

New and better computing technologies continue to revolutionize our capability to solve larger and more complex problems in science and engineering. One roadblock to further advance comes from our inability to manage and analyze data of increasing size. Just as gigabytes of disk and memory become standard specifications of every scientific workstation, some scientists have begun to routinely generate terabytes of data. Can they manage and understand data of this scale? If not, what's the purpose of generating these data?

Visualization transforms large quantities of raw data into graphical representations that exploit the superior visual processing capability of the human brain to detect patterns and draw inferences. Possibly the scientists can manage and digest the gigabytes/terabytes of data once condensed to a visual form. Scientific visualization—now an indispensable tool for scientists and engineers—has helped lead to many new discoveries or better designs. Consequently, generating, manipulating, and managing visualizations (a collection of images and metadata) also become a routine task for scientists.

Motivation

The process of scientific visualization is inherently iterative. A good visualization comes from experimenting with visualization, rendering, and viewing parameters to bring out the most relevant information in the data. A good data visualization system thus lets scientists interactively explore the parameter space intuitively. The more efficient the system, the fewer the number of iterations needed for parameter selection.

Over the past 10 years, significant efforts have gone into advancing visualization technology (such as real-time volume rendering and immersive environments), but little into coherently representing the process and results (images and insights) of visualization. This information about the data exploration should be shared and reused. In particular, for types of data visualization with a high cost of producing images and less than obvious relationship between the rendering parameters and the image produced, a visual representation of the exploration process can make the process more efficient and effective.

This visual representation of data exploration process and results can be incorporated into and become a part of the user interface of a data exploration system. That is, we need to go beyond the traditional graphical user interface (GUI) design by coupling it with a mechanism that helps users keep track of their visualization experience, use it to generate new visualizations, and share it with others. Doing so can reduce the cost of visualization, particularly for routine analysis of large-scale data sets.

New user interfaces

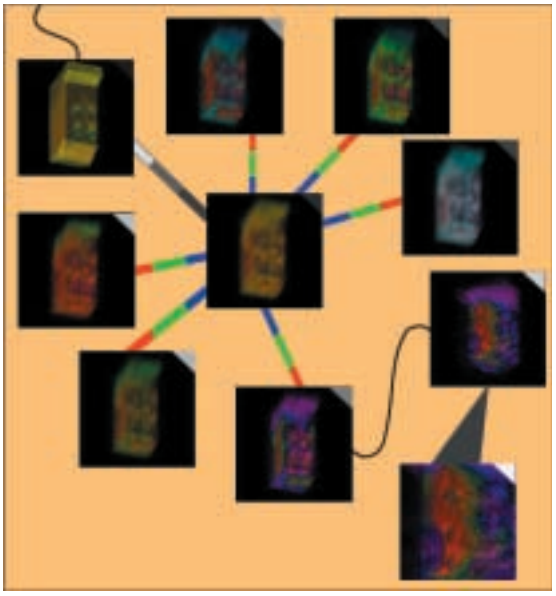
Most visualization software packages use a turnkey GUI containing a display area and a control panel with buttons, menus, slider bars, and key-in boxes. Users can create additional display areas in separate windows when needed. This setting works well when interactive viewing and manipulation of the visualizations are possible, and when the parameter selection step is fairly simple.

Some packages also come with a visualization development environment that lets the user define each task-oriented visualization pipeline as, for example, a dataflow network. The networks can be reused and shared among users. But no mechanism exists for comparing and reviewing images generated at different times, nor a way to visualize the iterative process of parameters selection and different sequences of data processing and rendering steps.

Little previous work in user interface designs incorporated any of the aforementioned features. Image graphs[1] offer a way to represent the data exploration process. Each node in an image graph consists of an image and the corresponding visualization parameters used to produce it. Each edge in a graph shows the change in rendering parameters between the two nodes it connects. Edges on the graph vary in appearance according to the type of relationship they represent. Image graphs thus make searching for desirable rendering parameters more efficient by showing how changes in parameters affect the visualization output for a given data set. They aid in the process of reviewing and recording the interesting structures found in the data set.

Figure 1 shows an image graph generated from the exploration of a furnace data set. The goal was to reveal the temperature distribution inside the furnace. From this graph, you can see the user first searched for an appropriate color transfer function before deriving the

0272-1716/00/$10.00 © 2000 IEEE

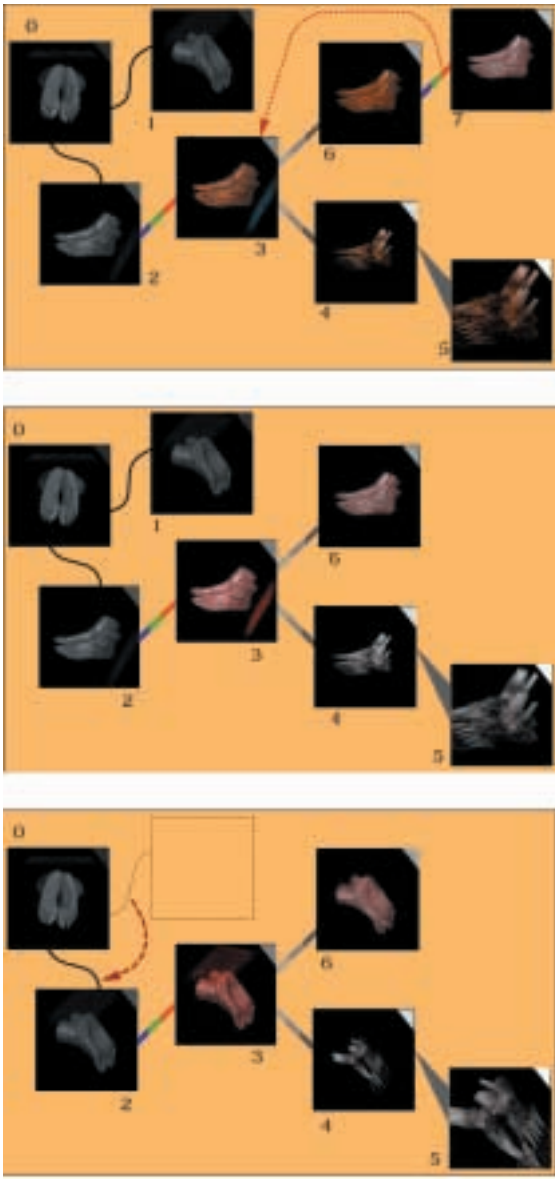


1 A graph representation of a set of images produced from exploration of a data set in a combustion engineering simulation. The user first experimented with a variety of color maps, then produced images by changing rotation and finally zoom factor.

desirable visualization shown in the lower right image by changing views (a rotation followed by zooming in). Note that nodes with similar parameters lie close to each other in the graph even though not created in sequence.

Image graphs aren't just static representations. Users can interact with a graph to review a previous visualization session or to perform new rendering. They can apply operations to nodes or edges, and propagate the resulting changes in rendering parameters through the graph. For example, a desirable visualization may result from taking the union of two opacity transfer functions (by combining two edges). An example of forward propagation appears in Figure 2. The top graph arose from the study of a magnetic resonance imaging (MRI) foot data set. Figure 2a shows a color edge being detached from node 7 and reattached to node 3 in the graph. This action will replace the color transfer function of node 3 with the color map of node 7 and trigger a rerendering at node 3.

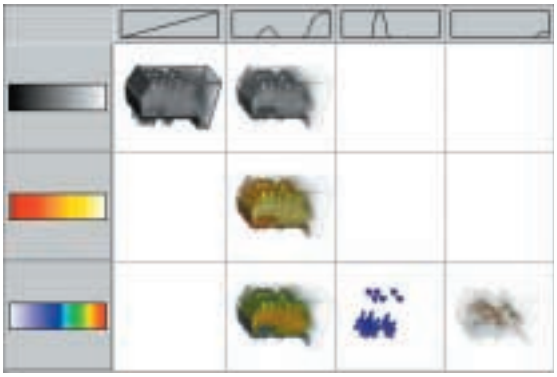
Furthermore, the effect of using a new color transfer function at node 3 will propagate through its peers. The middle graph (Figure 2b) shows the resultant image graph after this forward propagation. Compared to the images in the top graph, note that nodes 3, 4, 5, and 6 have all been updated. Node 7 has been removed, since it has become redundant to node 3. The bottom image (Figure 2c) shows the result of replacing one rotation edge (between nodes 0 and 2) with the other (between nodes 0 and 1). Compared to the images in the middle graph—except the upper left image (node 0)—all other images are updated using the new viewing angle. Note that in this way we can create new visualization results by operating on an existing graph and without introducing new graph nodes.



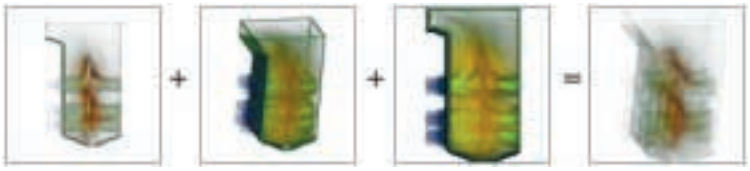
2 Property propagation of image graphs. The middle and bottom graphs show the impact of two edge operations on the whole graph.

An image graph, or any graph representation, can become too large to display effectively. In addition, applying operations to several different related paths of images proves difficult, especially for graphs displaying different data sets with no paths between them. These concerns are addressed by another design, which extends the standard spreadsheet interface and includes some traits of image graphs.

This spreadsheet-like interface[2] displays a 2D window into visualization parameter space that users manipulate as they search for desired results. The idea of using a spreadsheet-like interface for visualization isn't new. We've seen many employing similar ideas, such as Spreadsheet for Images by Marc Levoy (http://graphics.stanford.edu/projects/spreadsheets), Spreadsheet for Information Visualization by Ed Chi (http://www-users.cs.umn.edu/~echi/phd/), and Distributed Image Spreadsheet by Fritz Hasler (http://www.nren.nasa.gov/eos_distribution.html).



3 A spreadsheet-like user interface. Color maps (vertical axis) and opacity maps (horizontal axis) used are displayed along with the resulting images. The blank cells will be filled at the user's request. The user can also select a particular cell, use the corresponding color and opacity maps as default parameters, and bring out a new sheet to explore using a different pair of parameters.



4 Selected cells for a compositing operator. In this case, the user wishes to use the color and opacity maps of the first image, the view position of the second, and the zoom factor of the third. The fourth image displays the desired visualization.

Figure 3 presents a spreadsheet-like interface for volume rendering. Unlike previous spreadsheet work, the interface shown here matches rendering parameter values with image results to aid exploration. The tabular organization allows quick comparison of rendered results. Similar to image graphs, changes in parameter value will propagate through the spreadsheet interface. Operators provided on the parameters and images generate new values. In Figure 3, the points in the visualization space represent the volume-rendered images specified by the combined parameter values.

In conventional numerical spreadsheets, the correspondence between data and display is trivial. Visualization space is higher dimensional and more complicated to display. A set of default and selected parameters identify a spreadsheet cell. Changing one of the default parameter values translates the spreadsheet window; the cells update automatically. A method for referencing cells in the multidimensional space allows the application of operators and formulas across several different stacks of the interface. The visualization process becomes a process of maneuvering the spreadsheet window through parameter space.

Like image graphs, the spreadsheet interface assists the visualization process in two ways. First, the spreadsheet's structure provides an organized means of exploring the space of visualization parameters. Second, its dynamic capabilities speed the search for parameter values, letting users generate new parameters by combining a range of older values. The spreadsheet's structure permits applying operators to a wider range of values than in other interface designs. Figure 4 shows a possible combining operation to generate a new visualization.

Unlike image graphs, a spreadsheet-like interface doesn't display the order of operations, nor present the data exploration strategy used. The spreadsheet mainly shows the relationships between parameters as well as between parameters and visualizations. However, a spreadsheet-like interface seems particularly effective for manipulating information on the increasingly popular display wall. Clearly, either the image graphs or the spreadsheets can help organize visualization data in a way that facilitates review, reuse, and sharing.

Research directions

A pressing need exists for new user interface designs facilitating more efficient and effective data exploration and visualization in a collaborative setting. Any new interface should help reduce the number of iterations for parameter selection—required for effective visualization. The image graph and spreadsheet-like interfaces offer two examples demonstrating how an enhanced user interface design may help streamline the process of data visualization. Before incorporating the concept of visualizing visualizations into general visualization tools or even into advanced user interfaces like immersive VR, we need further research in the following directions.

Study of user and visualization requirements

Understanding the fundamental visualization process and user requirements is vital to good designs for interfaces and interaction techniques. Application scientists must participate in this effort and educate the designers. Can a design adapt to an individual's data analysis strategy or application-specific requirements? At what level should users interact with the visualization parameters?

Researchers in the field of ergonomics and human-computer interaction have addressed some of these issues. A book by Dix et al.[3] comprehensively describes a range of models for use during the user interface design process. For task-specific designs, we can learn much from the case study done by Treinish[4] on the notion of user/task-driven customization of content and interface in operational weather forecasting.

Fundamental user interface designs

We've recognized that traditional user interfaces can't support the increasingly complex process of scientific data

exploration. A fundamental change in the conventional designs must be made. Whether the new graph-based, image-based, spreadsheet-like, or other innovative user interface design offers more intuitive interaction and enhanced perception remains for investigation.

What is the semantics of a new user interface like the image graphs? What is the theoretical basis of, for example, property propagation, and how can a theoretically sound design lead to a practically effective interface? We can learn from user interface research for nonvisualization applications. Places to look for relevant discussions include the technical meetings ACM Symposium on User Interface Software and Technology, ACM SIGCHI Conference on Human Factors in Computing Systems, IEEE Symposium on Visual Language, and IEEE Symposium on Information Visualization.

Database issues

Developing mechanisms for keeping track of all types of metadata proves crucial to the overall interface design. The metadata can be associated with and result from visualization processes, ranging from the text-based descriptions of parameters used for a particular visualization algorithm to the annotation of intermediate and final visualization results. Users should be able to query these metadata through a database-centered tool in a collaborative and interactive visualization environment.

Some of these requirements resemble those for managing Web-based information systems. A more specific example is the metadata editing and browsing tool developed for management and visualization of large-scale scientific databases (http://www.llnl.gov/asci/pse/dem/dem.html). Essentially, a data model is created for capturing and sharing simulation data from application codes and for organizing, searching, and managing a variety of data.

Integrating the new interface design

Persuading users to adopt a new interface can prove as difficult as asking Fortran users to switch to using an object-oriented language. Ideally, the new design would result in an interaction mechanism so intuitive that users can grasp it right away. This isn't likely for sophisticated scientific data analysis. One way to convince scientists to try a new user interface design is to integrate the new design into existing visualization systems they've been using. This integration can be nontrivial, but it might help reveal areas to improve in the new designs.

User studies of the new designs

Finally, and most importantly, a comprehensive user study must be conducted with users from different disciplines. These users should include both scientists and novice users, such as their assistants. The user study should measure how quickly the new interfaces help derive desirable images of large, complex scientific data sets and investigate the extent to which the visualizations may be reused to produce new visualizations.

Faster rendering and larger storage space can only partially solve the large data visualization problem. The fundamental designs of user interfaces and interaction for data visualization need revisiting. The new designs must also account for representing the accumulated knowledge obtained during the visualization process as a part of the interface. Scientists are used to conventional interfaces and tools. Those of us who practice scientific visualization should guide scientists to harness the power of next-generation user interfaces to enhance scientific understanding. Otherwise, the gap between their ability to produce and digest data will continue to increase. ■

References

1. K.-L. Ma, "Image Graphs: A Novel Approach to Visual Data Exploration," *Proc. Visualization 99 Conf.*, Oct. 1999, ACM Press, New York, pp. 81-88.
2. T.J. Jankun-Kelly and K.-L. Ma, "A Spreadsheet Interface for Visualization Exploration," to appear in *Proc. Visualization 2000 Conf.*, ACM Press, New York, Oct. 2000.
3. A. Dix et al., *Human-Computer Interaction*, 2nd edition, Prentice Hall Europe, Hemel Hempstead, Hertfordshire, UK, 1998.
4. L.A. Treinish, "Task-Specific Visualization Design," *IEEE Computer Graphics & Applications*, Vol. 19, No. 5, Sept./Oct. 1999, pp. 72-77.

Readers may contact Ma at Department of Computer Science, 2063 Engineering II, University of California at Davis, One Shields Ave., Davis, CA 95616-8562, e-mail ma@cs.ucdavis.edu.

Readers may contact the department editors at rhyne.theresa@epamail.epa.gov and lloydt@us.ibm.com.