Using the World Wide Web to Provide a Platform Independent Interface to High Performance Computing

David W. Robertson and William E. Johnston[1]

Lawrence Berkeley Laboratory, 1 Cyclotron Road

Berkeley, CA 94720

Abstract

We have developed[1] a set of techniques for providing interactive 3D graphics via the World Wide Web (WWW) as part of the "Whole Frog" project[1]. The success of this project indicates that the Web and its associated browsers can serve as an easily used and powerful front end to high-performance computing resources.

We utilize the Common Gateway Interface capability of WWW servers to provide an interactive 3D visualization front end through Web clients. These techniques have been used to make a "Virtual Frog Dissection Kit", available as http://www-itg.lbl.gov/vfrog. A student using this Kit has the ability to view various parts of a frog from many different angles, and with the different anatomical structures visible or invisible (Figure 1).

1.0 Introduction

This paper describes work demonstrating the potential of the World Wide Web architecture to provide a uniform interface to high-performance computing applications. The back end computation can take place on as complex and powerful a distributed system as needed to support the information display on the front end. All of this is hidden from the user, and the front end is the same Web browser (e.g. Univ. of Illinois' Mosaic, or Netscape Communications Corp.'s Netscape Navigator) used to access many other kinds of information.

Use of the Web provides a platform-independent interface to applications. Browsers (Web front ends) are available for use with most common operating systems and window systems (UNIX, X-Windows, PCs, Macs etc.). The Web support of interactive input from browsers (via "forms") provides a significant subset of the functionality included in Graphical User Interfaces (GUIs) such as X Windows "toolkits".

The Web's design ([2], [3]) has proven very versatile, providing for a variety of high-level functions. These include: (1) publishing research results from high-performance computing, including interactive exploration of the data and methods used; (2) support of education by providing access to activities that are too computationally expensive for most secondary school resources, and; (3) mechanisms for collaboration and sharing possibly scarce and expensive remote experimental facilities.

We have used scientiÞc visualization as a sample high-performance application for the development and demonstration of Web capabilities to support front ends for distributed processes. We also hope to provide an example of ways that Web based user interfaces to scientific analysis and simulation codes can be built.

The use of visualization in conjunction with the Web means that end users of scientiÞc visualization do not necessarily require high-performance rendering systems at their location, and can use a variety of personal computers and workstations. One of the primary goals of the "Whole Frog" project is to enable K-12 students in particular to learn about and use scientific visualization (and frog anatomy), and to find out about high-performance computing and networking.

1. This work is supported by the U. S. Dept. of Energy, Energy Research Division, Office of ScientiÞc Computing, under contract DE-AC03-76SF00098 with the University of California. The authors may be reached as davidr@george.lbl.gov and wejohnston@lbl.gov. This is report LBL-36657.

Presented at IEEE Compcon 1995

The problem that we have selected as representative of many kinds of scientific visualization is that of visualizing large, 3D scalar fields. The example data set is a finely sampled 3D colored grid of the volume of a frog. The data is characterized by having significant internal surfaces (the anatomical organ boundaries).

In addition to exploring the interactive capabilities of the Web architecture, we are also presented with an interesting exercise in how to provide (experimentally) a computationally intense service to a large number of users with resources that are simultaneously (and primarily) used for other purposes. For example, the experimental package being described here - the Virtual Dissection Kit - has been accessed by 50,000 different sites, in over 50 different countries in the past 7 months. The generation of the nearly 200,000 rendered images satisfying these requests has been done on four or five large-ish Unix workstations on a non-interfering basis.

The number of accesses to a WWW page (an individual Web document) used for publication or education can vary considerably, depending, for example, on how much media attention it is receiving at any particular moment. Unless access to a page is restricted, the HTTP server (the system that receives and processes requests from Web browsers) must be prepared to handle a potentially large number of accesses per minute. If a page's intended use is interactive visualization, then long waits for an image are not acceptable. Given these considerations, one of our principal design goals was to have all of the rendered images of the frog available in a few seconds, or less, after the user interaction.

2.0 The Process of Visualization

There are a variety of ways in which 3D scalar fields can arise in math, science, and many other fields. They are trivially generated by evaluating a 3D function on a grid - a process that can lead to complex data sets and visualizations (see, for example [4]). 3D scalar fields also arise from various types of reconstruction problems that result in serial sections (e.g. tomography). In this case we obtained a data set from sectioning the body of a frog.

Obtaining a 3D data set that represents the internal structures of an animal starts with building a voxel data set (voxels are small cubes - the 3D equivalent of pixels) that can be used as input for surface and/or volume rendering software. Generation of this data set for a frog required mechanical sectioning: magnetic resonance imaging did not provide sufficient resolution, due to differences between mammalian and amphibian physiology. Each 125 micron slice was photographed and digitized, thereby providing a representation of the frog internals as roughly 10,000,000 tiny volume elements. Subsequently, semi-automatic segmentation (isolation and identification of structure boundaries) provided a "mask" representing each organ. The details of these techniques are available in the Whole Frog Technical Report [5].

Visualization of 3D scalar fields is typically done by "volume rendering" or "surface rendering." Volume rendering is a process that treats the voxels as transparent and counts up the contribution of all voxels in the line of sight. This type of visualization is computationally expensive, and works best with a display system supporting a very large color space (e.g. a 24 bit, RGB frame buffer). Surface rendering treats contours as "real" surfaces (e.g. represented as a polygonal mesh) and renders these surfaces accordingly. The rendering process is much faster, color resolution is not quite so critical, but the generation of the mesh representing the contour surface is still expensive. One of the optimizations in the Virtual Dissection Kit, in anticipation of heavy use, is that the contours are pre-computed, leaving only the rendering of the 3D object for responding to interactive requests. (That is, multiple objects and a viewing angle can be selected via the Web interface.) The masks mentioned above are the first step in generating a contour surface representation. The masks are converted to the voxel lists that define a contour surface by enumerating its intersection with the volume. We use this representation of the contour surface for rendering.

This approach of fixing the contour surfaces is reasonable for visualizing a frog, but may not be reasonable for a mathematical function or real-world data set where the structure is not well defined or known in advance. There is no reason that a general contouring capability could not be added to the Dissection Kit - it is only a question of supporting the computational load that such a capability would generate.

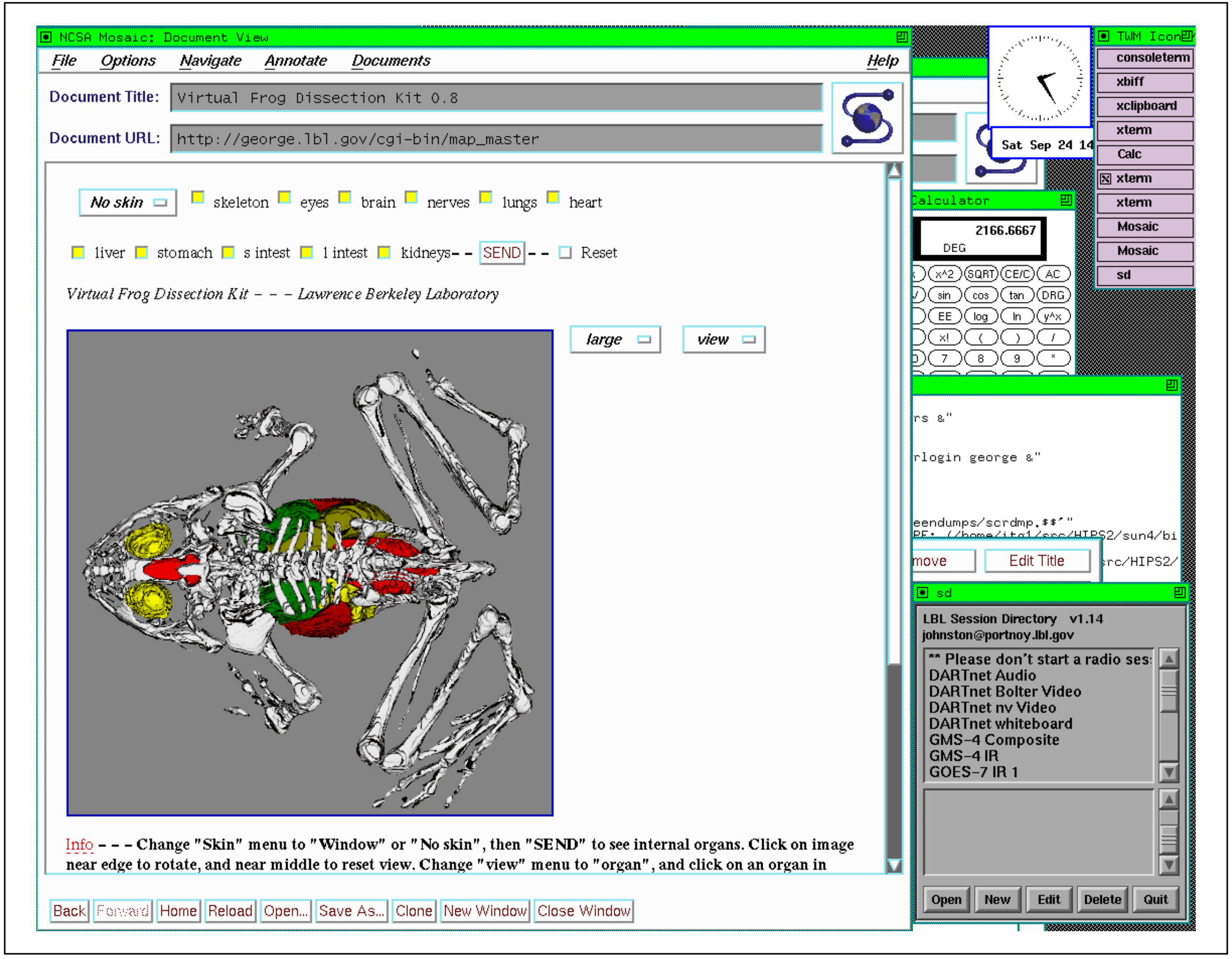
Given the frog volume data and organ masks, an intermediary list of voxels representing the frog was produced using the Dividing Cubes method [6] for 3D surface generation. This method has been modified by using mask values instead of algorithmic thresholding to identify the voxels that will be utilized in the surface generation step. The modified Dividing Cubes algorithm is used separately on each organ mask, resulting in lists of voxels for each organ. The surface normal vectors (needed for computing the color of a voxel given its original color, and the viewing angle and lighting conditions for the particular viewing specified by the user interaction) are similarly pre-generated and stored in this list.

The resulting list contains approximately 450,000 - 3D points representing the surfaces of the frog organs. If an organ is not selected for viewing, its portion of the point list is skipped. Rendering, that is, producing a 2D image representing the projection onto the screen of the 3D objects that are selected for display, is straightforward by using a z-buffer for hidden surface removal [7]. The alternative of rendering all images beforehand is not feasible due to the number of possible images (over a million when all possible views and all possible combinations of organs are taken into account).

Images are compressed before being sent back to a user. Since a typical frog image contains a high proportion of constant background, especially if only a few organs are selected, the GIF method of encoding [8] used by the HTTP protocol typically provides a compression ratio of at least 4:1.

Even with this form of compression the amount of image data to be sent can be up to 60 kilobytes, which takes about sixty seconds using a 9600 baud modem. With this in mind, a forms parameter on the Web page can be set to generate images of three different screen sizes (205 x 205, 304 x 304, and 480 x 480) resulting in transmitted image files of 5 kilobytes for those with slower connections, up to 20 kilobytes for normal use, and up to 60 kilobytes for high quality.

The use of point (voxel) primitives and previously generated normals results in rapid image generation (one second or less for the standard-sized image using a typical Unix workstation as the rendering server) with a good level of detail. This approach, in combination with the image-size options, meets the requirements mentioned above for rapid



Document Title: Virtual Frog Dissection Kit 0.8

Document URL: http://george.lbl.gov/cgi-bin/map_master

Virtual Frog Dissection Kit - - - Lawrence Berkeley Laboratory

Info - - - Change "Skin" menu to "Window" or "No skin", then "SEND" to see internal organs. Click on image near edge to rotate, and near middle to reset view. Change "view" menu to "organ", and click on an organ in

Sat Sep 24 14

2166.6667
DEG

```
rs &"

rlogin george &"

eendumps/scrdmp.$$'"
PE: /home/ito1/src/HIPS2/sun4/bi
src/HIPS2/
```

sd

LBL Session Directory v1.14
johnston@portnoy.lbl.gov

** Please don't start a radio ses
DARTnet Audio
DARTnet Bolter Video
DARTnet nv Video
DARTnet whiteboard
GMS-4 Composite
GMS-4 IR
GOES-7 IR 1

Figure 1 - Screen Dump Showing Rendered Image and Interaction Mechanisms

turnaround between a forms submission and return of the viewable image.

3.0 The WWW Forms Interface

The Web provides mechanisms to allow links from one document to other text, audio, image and movie documents residing anywhere on the Web. In addition, the “forms” capability provides such GUI features as text input, and checkboxes and menus for selection among enumerated choices. User interaction is also possible through clicking on images in the document (which returns the x and y location of the pointer in that image) providing the mechanism for functions such as rotation and graphical object selection. Taken together, these capabilities are a powerful means of allowing interactive user input and data exploration.

Figure 1 shows the forms interface for the Virtual Dissection Kit, with a rendered image of the frog. This interface provides three ways for the user to interactively control the graphical display and learn about frog anatomy: controlling which organs (objects) are visible, controlling the angle from which to view the frog, and using a mode of interaction that brings up brief descriptions of the organs seen in the image. In Figure 1 the first two of these options have been exercised to produce the image. A separate paper [9] describes the “Virtual Dissection Kit’s” interface in detail.

A user sets forms parameters and then submits them to the HTTP server. The Common Gateway Interface (CGI) provides a means to run a program from the HTTP server that handles the forms submission, performs the appropriate action, such as rendering an image, and returns the resulting document back to the client. This document can be anything that can be written in the Web’s Hypertext Markup Language (HTML).

In the case of the Virtual Dissection Kit, the document returned gives the appearance of a screen in which the only change is in the “window” containing the image of the frog. This is accomplished by returning the HTML document creating the screen that appeared before form submission, updated with a pointer to the new rendered image, and with the latest form settings.

As an example of the flexibility provided by forms, a menu setting in the Kit interface indicates to the CGI script that it should return a form document that is a translation of the user interface into a different language (the Virtual Frog Dissection page is currently available in English, Spanish, French, Dutch, and German). A user can also click on an organ in the rendered frog image and discover its name and function, which are returned as text. In general, the provider of information and research results has wide leeway in allowing user interaction and exploration, and in formulating a response, since documents can be generated on the fly.

4.0 The Virtual Dissection Kit Architecture

When a user submits a form from the client, the form settings are passed over the network to the HTTP server, and then to a CGI script. The script parses the incoming data and sends the result to an already running process (server) on a different machine. The load on the HTTP server system is minimal; most of the computation is performed on the rendering server.

In the event of multiple accesses, the load is further spread out by having several (currently five in the case of the Frog Dissection Kit) different systems available for graphics rendering. One of these systems is chosen at random to perform the rendering, which takes about a second for the standard image size. There has not been a noticeable impact on the performance of these systems, which are regularly used for other purposes by our group (during peak periods in December 1994, there were up to 12 accesses per minute that caused image rendering).

The rendering server generates an image from the data list representing the frog. The data list, consisting of points and associated surface normals, is about 5 megabytes. It is loaded once upon the instantiation of a rendering process, and is thus always mapped in memory.

After the image is generated by the 3D rendering, it is GIF encoded and written to a temporary file. The CGI script rewrites the Web page with the new forms settings and the location of the image file in HTML format to standard output, where it is intercepted by the HTTP daemon and sent back to the client. Before writing the form settings out, the script checks to see if any of them have changed during the current invocation, as a result of rendering or other actions, and updates them as necessary.

5.0 Conclusions

The World Wide Web model provides a flexible and powerful method for providing access to documents and data, and resources such as high-performance computing. The availability of Web browsers for the most common platforms and the presence of the Web throughout the world ensure very wide dissemination of information.

The ease of use of Web development tools has enabled the implementation of a distributed system such as the Virtual Dissection Kit as an information server. Computational resources are extended beyond the machine running the HTTP server by using Unix interprocess communication mechanisms for communication with rendering servers on other machines.

The Virtual Dissection Kit is an example of how scientific visualization in particular can be made available over the Web. It provides a capability for students that typically would be too computationally expensive to provide at a school site. An alternative to the dissection of a real frog, it also adds new capabilities such as un-dissection. The spatial relationships of organs to each other are easily seen, in combinations that would not be realizable in a real dissection. Future plans include techniques to build movie sequences on the fly (based on user input) to better visualize 3D relationships among organs; a virtual cutting tool; and finding mirror sites in places such as Europe, where the connectivity to LBL is slow.

Visualization using the Kit requires substantial use of computing power and memory resources. Providing this without causing an undue load on our site was made possible by distributing the task between the HTTP server and the rendering server, and having a pool of machines available for rendering. In general, a well-equipped facility can provide services such as the Kit on a non-interfering basis by spreading the load among several workstations. Powerful workstations, and the capabilities such as visualization that they provide, can be made much more accessible to sites such as K-12 schools through use of the Web.

Due to their typically heavy load, use of more powerful systems than high-end workstations for Web publication or education may not be feasible. However, the Web offers the potential for research collaborations using supercomputers and large, scarce experimental facilities. In a collaboration there are not nearly as many users, and access to resources can be limited to a research group through use of Web capabilities. An example of the use of the WWW for such a collaboration is the Bay Area Gigabit Testbed [10], performing research into high-speed networks.

6.0 Acknowledgments

This work was sponsored by the U. S. Dept. of Energy, Energy Research Division, Office of Scientific Computing, John Cavallini program manager. Several people in the LBL Functional Imaging Group of the Life Sciences Division, headed by Dr. Tom Budinger, were instrumental in assisting with generating the frog data set. Several computer science students, including in particular, Wing Nip, have provided invaluable assistance, and several high school teachers from the LBL Center for Science and Engineering Education summer program contributed to various aspects of the project.

7.0 References

1) Johnston, W.E. *The Whole Frog Project* (Web page). http://george.lbl.gov/ITG.hm.pg.docs/Whole.Frog/Whole.Frog.html, University of California, Lawrence Berkeley Laboratory, Berkeley, CA (1994).
2) T.J. Berners-Lee, R. Cailliau, J-F Groff, B. Pollermann, CERN, “World-Wide Web: The Information Universe”, published in *Electronic Networking: Research, Applications and Policy*, Vol. 2 No 1, pp. 52-58 Spring 1992, Meckler Publishing, Westport, CT, USA.
3) http://www11.w3.org/hypertext/WWW/Bibliography.html
4) G. Fischer, ed., *Mathematical Models*, Friedr. Vieweg and Sohn Verlagsgesellschaft mbH, Braunschweig, Germany (1986).
5) Nip, W. and Logan, C. “Whole Frog Technical Report.” LBL-35331, University of California, Lawrence Berkeley Laboratory, Berkeley, CA (1991).
6) Cline, H.E., Lorensen, W.E., Ludke, S., Crawford, C.R., and Teeter, B.C. “Two algorithms for the three-dimensional reconstruction of tomograms.” *Medical Physics 15* 3 (May/June 1988), 320-327.
7) Foley, J. D. and van Dam, A. *Fundamentals of Interactive Computer Graphics*, 2nd ed. Addison-Wesley Publishing Company: Reading, MA (1990).
8) Graef, G. “Graphics formats: A close look at GIF, TIFF, and other attempts at a universal image format.” *Byte 14* 9 (Sept. 1989), 305-310.
9) Robertson, D. W., Johnston, W.E., and Nip, W. “Virtual Frog Dissection: Interactive 3D Graphics via the Web.” *Proceedings, The Second International WWW Conference ‘94: Mosaic and the Web,* Chicago, IL (1994).
10) http://george.lbl.gov/BAGNet.html