Visualization Over The Internet

Andreas Kolb
ak@teleweb.at

Institute of Computer Graphics
Vienna University of Technology
Austria

Abstract

Scientific visualization is a powerful tool to investigate and analyze complex problems such as CFD-simulation data or medical volumetric datasets. Especially these days where data sets in the range of gigabytes to terabytes have to be illustrated and investigated a large number of valuable visualization techniques have been developed by researchers all over the world. Unfortunately, it takes quite a long time until these scientific advances are available to a broad user community. This is the point where the Internet jumps in as becoming more and more the bridge between scientists providing advanced visualization techniques for complex systems and users all over the world who investigate their problems in various applications.

Keywords: Visualization, Internet

1. Introduction

1.1 WWW as a platform

One of the most impressing developments in communication technology during the past decade has been the amazingly fast evolution of the World Wide Web based on the Internet. The useful combination of intuitive web browsers with similar look and methods of interaction on multiple platforms and servers spreading information all over the web provides a powerful facility to distribute information to a large number of users. Information sharing was never easier than now with the facility Internet.
The main advantage of the Internet is it's independence of hardware and software and thereafter makes it possible to connect new developments in the field of scientific visualization and almost any other topic with a large community of users.

1.2 Motivation

Scientific visualization is typically used to illustrate and investigate numerical data with the aim on images that are easier to interpret than the raw underlying numbers. The development of new visualization techniques is usually done by scientists using expensive specialized hardware and visualization software which is in common ways not accessible to the ordinary users. The results of their research can only be passed to other scientists with similar hard- and software or some lucky users who are also owner of such special devices and software components. This is usually only a small subset of potential users, but to reach a broader field of audience, the developed visualization techniques have to be implemented on other, common platforms as well. This step is very time and money intense and, of course, not taken by scientists at all. Because the number of users who have a

desktop PC is significantly higher than the number of users currently working with high performance visualization systems, a visualization solution running on simple desktop PC's will improve the distribution of innovative methods from scientific developers to users significantly. The benefits for the developers would be more feedback from a broader field of users and thereafter the possibility to tune the system so that it fits the users best.
To cope with that problem the world wide web might be the key. Due to its "near" platform independence and intuitive way to use it hides the complexity of the beneath internet and connects computers from various types and platforms.

1.3 Demands on a visualization technique

To be broadly accepted by the user community a good visualization technique must cope with the following problems and include these important features:

- **Portability:** The visualization technique should be portable and run on all kind of common platforms to grant a great amount on potential users the usage. Of course, the most common platforms like UNIX workstations, desktop PC's using WindowsNT and Macintosh computers should be covered in addition to special scientific hardware. Java and VRML both provide the power to realize this.
- **Intuitive usage:** The technique offers a set of predefined visualization configurations covering common problems and presents the same intuitive interface on each platform to ease the use for people with more than one computer and/or when changing the operating system. Almost any browser released up to now meets this requirement. The user shall also be able to use at least one sample data set to test the visualization technique if it meets his requirements.
- **Flexibility:** The visualization technique should be flexible, allowing the user to visualize the data in a customized way with a minimum amount of work due to premade configuration settings. The user shall also be able to go into further details as he is given the possibility to tune all the parameters of the premade configuration by hand. It should also be of no great afford for the user to visualize his own data rather than given sample data sets.
- **Interactivity:** The illustration and the investigation of the data set through the user shall be an interactive process. The user should be able to change parameters, points of views and set filters while investigating the data to make the visualization a living process and to invite to "play with the data".
- **Unambiguousity:** The technique shall be at least rudimentary safe to misinterpretation of the visualized data due to wrong settings or parameters. This is a feature who bears much responsibility because a visualization technique with to much overwhelming parameters to change can be as useless as a technique with to much restrictions.

2. Visualization over the WWW

2.1 Advantages and Disadvantages

The usage of the word wide web at visualization techniques has initiated the development of many platform independent data and program standards. The distribution and retrieval of information has become much more easier than ever before due to standardization taking place on several levels:

- WWW Browsers from different manufacturers running on different platforms use very similar user interfaces and follow mainly the same intuitive ways to use. Regardless of hardware or platform the user interacts in the same way with the browser who handles the different problems with platform, hardware and the internet itself.

- Standardized data formats like HTML have been defined with the development of browsers making publishing information world wide very easy. The data has to be provided only once and can be retrieved from a wide range of platforms all over the world.

As HTML for textual information and JPG/GIF for two dimensional images VRML has gained importance as a modeling language. For three dimensional scenes it can easily be integrated into WWW pages opening the opportunities to present three dimensional data sets with optional user interaction.

Finally, the programming language JAVA [Jav] has created the possibility to write and compile software once and run it on different platforms without modification. JAVA is close coupled with HTML and VRML, providing a toolkit to create portable and interactive applications based on the internet. The disadvantage which comes with that independence is the lower performance in comparison to "machine native" compiled code. The reason for that loss of performance is that JAVA code is interpreted instead of being executed directly. There are several ways to cope with that problem like just-in-time compilers, which translate JAVA code into native machine code before the execution or to use a processor which accepts JAVA as machine language. Such an approach is made by the Sun company – the development of the "pico Java" kernel as an inexpensive network computer.

2.2 Visualization pipeline

The most common reference model of visualization over the word wide web is the visualization pipeline described in Upson [Ups] and McNabb[Nab]. This model sees the visualization process as a pipeline in which a source of data is fed in and filtered, mapped and rendered to create a final image to envision the data.

The filter process selects the data of interest, the map process creates an abstract geometrical representation of the data – perhaps a contour map – and the render process takes the 3D geometry from the map process and applies lightning, shading and projection to create a final image. This pipeline is shown in the figure below:

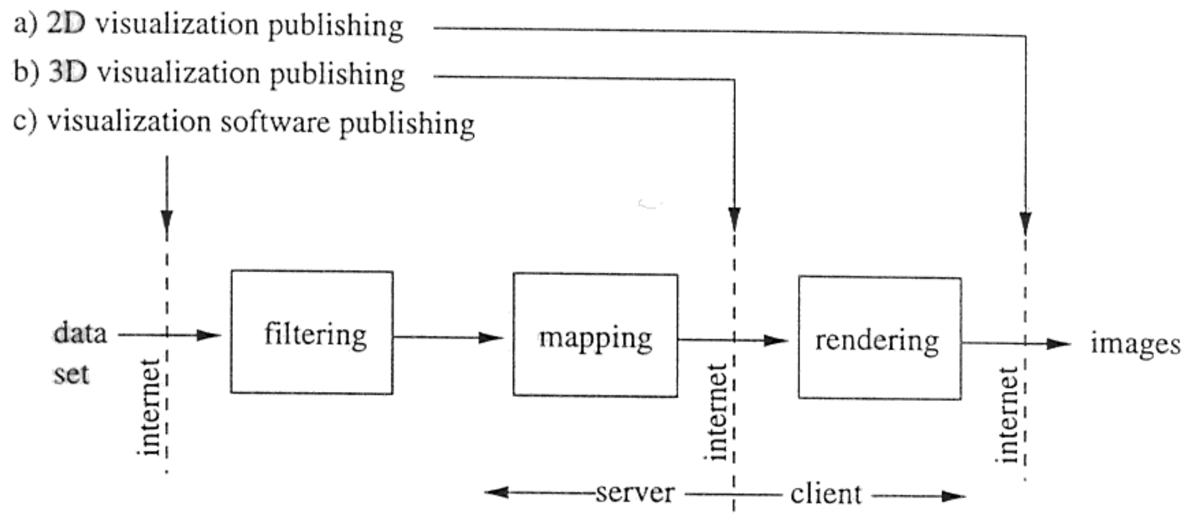

a) 2D visualization publishing
b) 3D visualization publishing
c) visualization software publishing
data set
filtering
mapping
rendering
images
internet
internet
internet
server
client

Figure 1: Visualization Pipeline

This model can also be applied to visualization over the web. The flow begins with the data set and ends with the final visualization image at the viewer, but different scenarios occur when considering who has responsibility for the intermediate processes and where to cut this pipeline to use the internet as a connecting medium. Generally three scenarios can be distinguished which differ mainly in their characteristics of flexibility and demands on the visualization server and the client performance which are explained in the next sections in detail.

3. Methods of visualization

3.1 2D visualization publishing

The user sends a request to the publisher in which he tells his needs and wishes by selecting the desired parameter in, for example, HTML forms. The publisher uses CGI scripts to process the information given by the user and creates the visualization as an image or video sequence and posts it on the web. The viewer can investigate this depiction either directly with his browser or download it for using a helping application such as an MPEG player or an image processing program. This scenario is shown in the figure below:

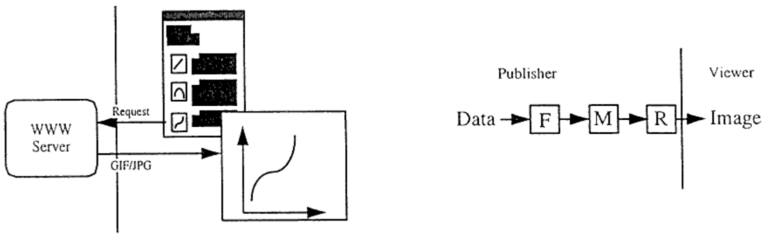


Figure 2: 2D visualization, getting images across the WWW

The advantages and disadvantages are clear: This is the least expending method for the viewer to get his visualization as all the filtering, mapping and rendering is done by the visualization server. The client has only to download the picture or video sequence that was rendered once. A major disadvantage for the publisher is that he has to render a new image for every new view the users selects or parameter changed. This means high demands of performance on the visualization server and makes interactive working by the user almost impossible.

3.1.1 Graphical Address Browser of Vienna

The municipal authorities of Vienna provide a very special service over the net [Gab]. The users can enter an address with the name and number of any street or place of Vienna, choose a resolution and are served with the according sector of Vienna's map. Further more the users can change the viewpoint by clicking into the image or choose another resolution. It comes in very handy that at some resolutions the routes of all kind of public transport are printed on the map with the according stations. A sample of such a map is shown on the figure below.

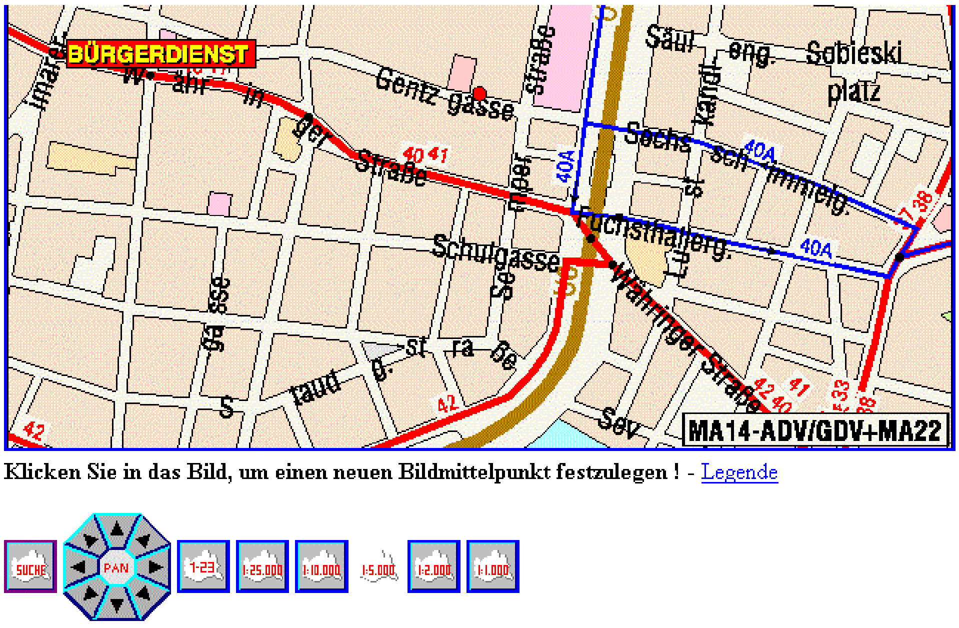

Klicken Sie in das Bild, um einen neuen Bildmittelpunkt festzulegen ! - Legende

Figure 3: Sector of Vienna’s map

3.2 3D visualization

With this method the publisher transmits the visualization as a three dimensional model to the user. The most common way to do this nowadays is using VRML, which allows to shape a world rather than a simple image. The viewer has the possibility to render the model as he wish with a special browser and may change the viewpoint to gain more insight into the visualized data.
With a VRML visualization the viewer has much more freedom to investigate than viewing a simple image but the abstract model of the data has often been predefined by the publisher and can’t be changed by the viewer even if there are some visualizations where the user can set parameters according to the model to be visualized via HTML forms. The figure below depicts this method and the place, where the visualization pipeline is cut by the internet. With this method the visualization work is divided into the viewers part – the rendering - and the remaining server part which presupposes more computing speed at the viewers side due to the rendering part. But even with the rendering part passed to the user’s side the visualization server still has to compute visualizations for many users which costs much performance and resources.

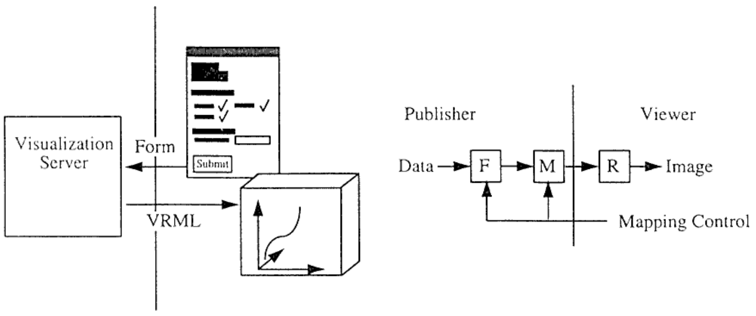
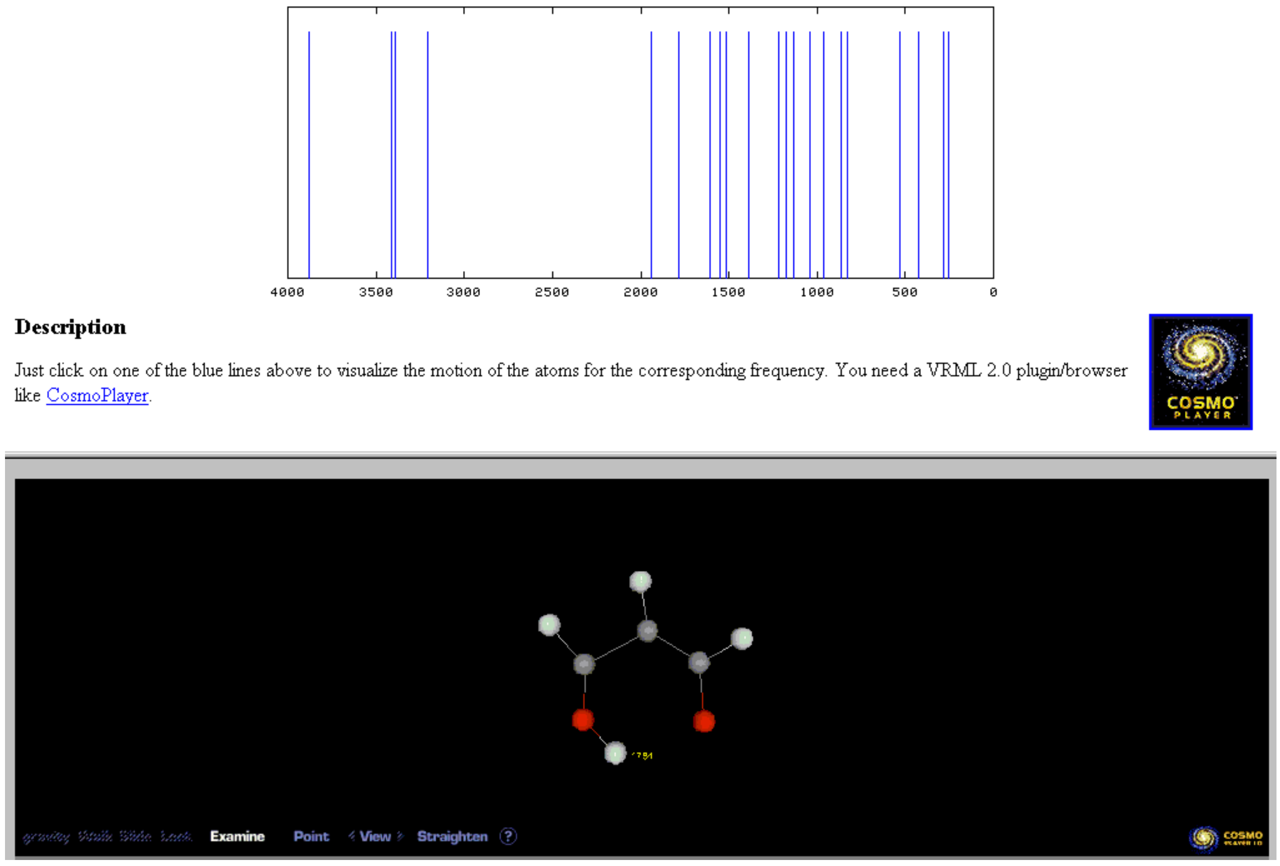


Figure 4: 3D visualization, getting objects building a world across the WWW

3.2.1 Normal Mode Visualization

The figure below shows an example of 3D visualization [Nor] at the University of Darmstadt, Germany. The upper part is a clickable image where the user can choose a frequency and below he can see atoms move at his chosen frequency in a VRML file. There he can interactively change his viewpoint at acceptable performance due to the rendering takes place on the viewers machine.



Description

Just click on one of the blue lines above to visualize the motion of the atoms for the corresponding frequency. You need a VRML 2.0 plugin/browser like CosmoPlayer.

Figure 5: Screenshot of a 3D visualization with a VRML file

3.3 Visualization Software Publishing

The third method occurs when the viewer is doing the whole visualization and no computational power needs to be provided by the publisher. This bears the most advantages for the publisher as the resources he has to provide to facilitate his visualization technique are minimized. In terms of the visualization pipeline described earlier the whole pipeline moves to the viewer.

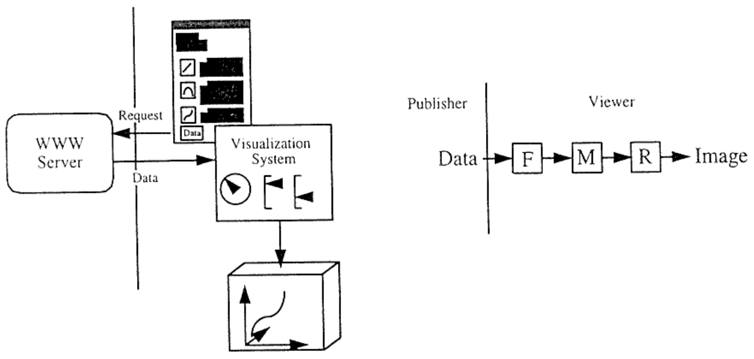


Figure 6: Visualization software publishing

Another major advantage of this method is the independence of the viewer from the publisher during the visualization. But on the other hand the disadvantage of this method is that the client has to do all the computations and so must provide high computing resources. A supreme example of visualization software publishing is the VizWiz JAVA applet which is described in the next sections in detail.

4. VizWiz

4.1 Introduction

To describe a scientific visualization tool in detail the Java applet VizWiz [Viz], developed at the University of California San Diego, is introduced here in a brief way. As a Java applet VizWiz is completely platform independent and usable over the whole world wide web. It provides basic interactive scientific visualization functionality, such as isosurfaces, cutting planes and elevation plots. The data can be uploaded into the applet by the user via the applet's web server and the visualization is generated and can be investigated interactively as shown below.

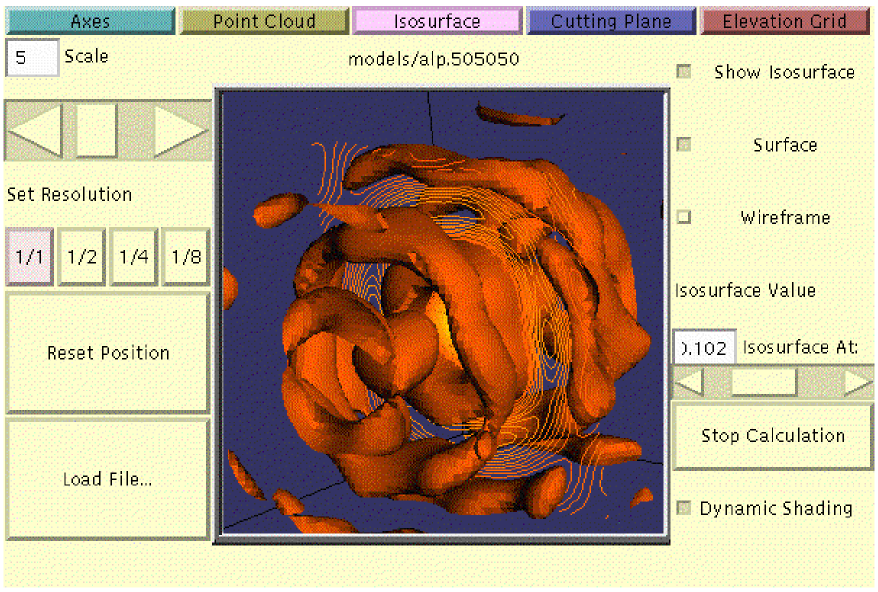


Figure 7: The VizWiz user interface

The unique feature of VizWiz is that it makes interactive scientific visualization possible for any data set across the internet. The viewer’s only prerequisite is a Java enabled browser which are wide spread and available without any fee like Netscape’s Communicator or Microsoft’s Explorer. VizWiz is not only a useful platform independent visualization system but also a demonstration of both the pros and contras of implementing such a complex application as a Java applet.

4.2 Motivation of development

Plenty of scientific visualization tools are available today but all of these applications share a major disadvantage – they are platform dependent. Even some of them are released for multiple platforms, but they are only ported from the source, native platform onto other platforms. This means great work due to the structure of visualization systems. They are graphic intense and often use platform if not even hardware specific routines that are not portable easily. Additionally, developers of multi-platform systems must test the code on every platform, support every version separately and – of course – have deep insights into every platform they develop for. If changes or new releases are made they have to bee made and tested for every platform, too. VizWiz as a Java applet has only to be developed, coded and tested once and also works on every Java supporting platform ranging from a simple 486 PC’s to a SGI or SPARCstation. Nowadays there is no way to perform 3D visualizations on such a wide variety of platforms with so little overhead for both the user and the programmer.

Another main disadvantage of most of the visualization systems out by now is that they must be download or purchased and installed by the user before they can be used or tested. Although this is not really difficult but very annoying, time intense and often disappointing when the system does not meet with the users requirements. In other words it is not possible at all to try out a tool without downloading it, installing and configuring it. Making an application tool available as a Java applet copes with that problem as it runs automatically when the web page in which it is implemented is download by the viewer’s browser. VizWiz also automatically load a sample data set which can be immediately used to get in touch which various functions of VizWiz. Another motivation to develop a system using Java was that VizWiz is resident on the internet and nor locally on the viewer’s machine and can be used from everywhere through the internet.

As a conclusion the main topics of the motivation developing using Java were to provide

- a new kind of scientific visualization tool,
- almost complete platform independence,
- a simple and intuitive usage, especially at the first time,
- a trivial to try out tool and
- to develop a system with which it is possible to check out the advantages and pitfalls of Java as a scientific programming language.

4.3 VizWiz and Java

Java has been rumored as the language that will enable a shift in storage of application from the desktop to the internet. The powerful combination of CGI scripts and the WWW itself opens a wide range of possible applications. But these applications have a shared feature. They are based on user input into an HTML form and thus are not interactive in the sense a Java applet makes possible. The main problem while developing VizWiz in Java was that a Java applet is not allowed to access the file system on the machine it is executing for security reasons. How the developers cope with that problem is explained in the next chapter.

4.4 Goals and implementation

As briefly mentioned in the chapter about motivation of development there were several goals the developers wanted to archive which are explained in detail below:

1. The tool had to be completely platform independent.
2. It had to be uncomplicated and easy to use, especially the first time.
3. It had to provide the expected basic set of 3D visualization functionally.
4. Its performance had to be acceptably interactive and its graphic acceptable attractive, even on low end machines.
5. Users had to be able to easily use this tool to visualize their own data sets.

4.4.1 Platform independence

Although it is impossible to archive real platform independence, Java is the best way to approximate it. The major problem using Java was the performance as the only way to implement 3D graphics without requiring that the user have installed any supporting software besides a Java enabled browser is to use the standard Java API. As Java programs are interpreted and not executed directly they are generally slower than their counterparts in native code would be. But the loss of performance due to the interpretation is only the smallest one. All the windowing and graphics code VizWiz uses are made through Java which means by software rather than by machine specific accelerators and other hardware. As VizWiz is both computationally and graphically intensive sacrificing of direct hardware graphics support bears the major loss in performance.
This problem will be solved in future releases with the Java 3D API since it will enable to take advantage of 3D hardware.

4.4.2 Ease of initial use

Using Java for developing VizWiz the applet is downloaded automatically and running it the first time is no more difficult than simply visiting a web page. The developers cope with the security problem that Java applets are not granted direct file access in a very special way. This feature is very understandable from the security aspect but is a very hindering one when developing a useful application where it is necessary and obvious for “normal” programs to load data from the users machine and save settings, outputs and temporary files. VizWiz uses CGI scripts and HTML forms to solve this explained in point 5.4.5 “Visualization of users data”

4.4.3 Visualization functionality

As VizWiz needed to provide a set of basic visualization functions the developer included support for isosurfaces, cutting planes, point clouds and elevation plots. All the surface functions are polygon surfaces which can either be filled or visualized using a wireframe. All visualization objects are implemented in the same manner as a set of vertices and connection indices. Additionally a base color for each line or polygon is specified. Furthermore, all visualization services use the same rendering engine which makes it easy to add new services.
In addition to the static position of every polygon two more parameters are calculated by VizWiz. A color based on the base color and the angle the polygon makes with the light source and a distance from the camera used for polygon sorting during rendering.
The control panel on the left side of the window allows to change parameters affecting the display like resolution, scaling and loading files. On the right side there is an inspector panel which is context sensitive, a different one for each object, which allows to change individual parameters for the particular display object. The display area in the center shows the actual visualization and can be rotated by dragging the mouse over it.

4.4.4 Interactivity

To archive both an adequate performance and an interactive visualization tool various techniques had to be used while developing VizWiz. To compensate the loss of performance due to Java’s inability to use potential 3D hardware the developers used smart caching to avoid duplication of computations, simplified rendering and interactive resolution control through the control panel.

4.4.5 Visualization of users data

As VizWiz is running in standard browsers like Netscape or Internet Explorer it has no direct access to the file system of the viewers machine. HTML is not entirely restricted from reading the local file system so the developers used a HTML/CGI workaround to upload the viewers data. The name of the file containing the data to be visualized can be entered by the viewer into an HTML form and the contents of the selected file will be uploaded to the VizWiz server and processed by a CGI script. This script reads the data and generates a new file on the server which is readable by the VizWiz applet and downloadable by the viewer.

5. Conclusions

VizWiz is an intuitive to use tool with the ability to visualize own data sets at an acceptable performance. But it is obvious that tools like VizWiz should use Java3D as in natural Java the performance penalties due to the missing opportunity of using machine dependent 3D hardware is the most hindering point.
Generally as the computing speed of desktop PC’s of “ordinary” users increases the visualization pipeline will move to the users in the next time. Applets like VizWiz will be the next visualization techniques as every publisher can host such services at almost no cost.

6. References

[Gab] WWW, “Graphisches Adreßortungssystem”, http://www.magwien.gv.at/gismap/cgi-bin/wwwgis/adrsuche/

[Hab] R.B. Haber and D. A. McNabb, “Visualization idioms: A conceptual model for scientific visualization systems” pp74-93, IEEE, 1990

[Nor] WWW, “Normal Mode Visualization”, http://www.pc.chemie.th-darmstadt.de/vrml/vib/

[Ups] C. Upson et al, “The Application Visualization System: A computational environment for scientific visualization”, IEEE Computer Graphics and Applications, Vol 9, Number 4, pp30-42, 1989

[Viz] WWW, “VizWiz, a Java applet for interactive 3D scientific visualization on the Web”, http://sdsc.edu/vizwiz/vizwiz.html