Web-Based 3D Visualization in a Digital Library of Mathematical Functions

Qiming Wang, Bonita Saunders

National Institute of Standards and Technology
qiming.wang@nist.gov, bonita.saunders@nist.gov

Abstract

The National Institute of Standards and Technology (NIST) is developing a digital library of mathematical functions to replace the widely used National Bureau of Standards Handbook of Mathematical Functions [Abramowitz and Stegun 1964]. The NIST Digital Library of Mathematical Functions (DLMF) will provide a wide range of information about high level functions for scientific, technical and educational users in the mathematical and physical sciences. Clear, concise 3D visualizations that allow users to examine poles, zeros, branch cuts and other key features of complicated functions will be an integral part of the DLMF. Specially designed controls will enable users to move a cutting plane through the function surface, select the surface color mapping, choose the axis style, or transform the surface plot into a density plot. To date, Virtual Reality Modeling Language and Extensible 3D (VRML/X3D) standards have been used to implement these capabilities in more than one hundred 3D visualizations for the DLMF. We discuss the development of these visualizations, focusing on the design and implementation of the VRML code, and show several examples.

Keywords: virtual reality modeling language, VRML, extensible 3D, X3D, digital library, 3D visualization, special functions

1. Introduction

Fundamentally unchanged since its initial publication in 1964, the National Bureau of Standards (NBS) Handbook of Mathematical Functions [Abramowitz and Stegun 1964] continues to be a popular resource in the mathematical and scientific community. Consequently, the National Institute of Standards and Technology (NIST), the successor organization to NBS, has decided to update the handbook, publishing it on the web as the Digital Library of Mathematical Functions (DLMF) [Lozier 2002]. The contents of the DLMF, also being published in hardcopy format, will include mathematical formulas, references, methods of computation, graphs, and links to software for more than 40 different functions. The web site will feature interactive navigation, a mathematical equation search, and dynamic interactive graphics. It is slated for completion in 2005, but a mockup version with limited capabilities can be found at http://dlmf.nist.gov .

High level, or special, mathematical functions such as the Bessel functions, gamma and beta functions, hypergeometric functions and others are important for solving many problems in the mathematical and physical sciences. The graphs of many of these functions exhibit zeros, poles, branch cuts and other singularities that can be better understood if the function surface can be manipulated to show these features more clearly. A digital library offers a unique opportunity to create cutting-edge 3D visualizations that not only help scientists and other technical users, but also make the library more accessible to educators, students and others interested in an introduction to the field of special functions.

After investigating web-based 3D graphics file formats, we selected the Virtual Reality Modeling Language (VRML) to create our visualizations. VRML is a standard 3D file format for which browsers and plug-ins are publicly available on the Internet, and its functionalities satisfy our web-based visualization requirements. The emerging XML based standard, X3D, is the successor to VRML and will support all VRML functions. We have successfully translated and tested several of our VRML files to confirm that conversion to X3D format is both feasible and practical.

The DLMF web site will be organized into chapters, and most chapters will have a section called "Graphs and Visualizations" where links to 2D and 3D figures can be found. Each 3D image will be associated with a link to a VRML/X3D file that the user can manipulate interactively.

Although we continue to address such issues as data accuracy, the clipping of surfaces falling outside a specified range, and the generation of computational meshes [Saunders and Wang 2000] to capture significant features of a function, this paper assumes these issues have been resolved and focuses solely on the creation of the VRML files. In particular, we describe the development and implementation of MathViewer, a VRML prototype, or object-defining and extensibility routine, which controls the interactive manipulation of the 3D surfaces. In Section 2 we discuss the user capabilities provided by MathViewer. Section 3 describes the high and low-level VRML prototypes that make up MathViewer, Section 4 shows still images of visualizations created for the DLMF, Section 5 is the experimental result of X3D files, and finally, Section 6 offers some conclusions and suggestions for enhancements.

2. MathViewer

While customary VRML browser controls such as rotate, zoom, and pan permit a user to examine a 3D display from an arbitrary direction, we also wanted to provide options for more extensive manipulation of function surfaces. Discussions with DLMF editors and authors led to several suggestions concerning the types of interaction users might desire. We developed the VRML prototype MathViewer to implement several of these capabilities discussed below.

MathViewer generates a display consisting of a 3D function surface with coordinate axes, and a main control panel with five buttons. The first four buttons generate subordinate control displays that provide cutting plane control, axes and label style control, scaling control, and color map control. The fifth button, "none", deletes the subordinate display. Figure 1 shows the modulus of the gamma function in the CosmoPlayer VRML browser. The dashboard display is specific to CosmoPlayer.

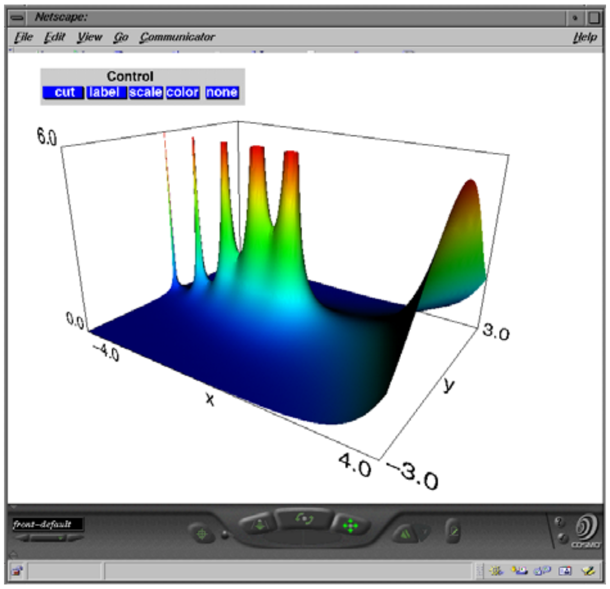

Netscape:
File Edit View Go Communicator Help
Control
cut label scale color none
6.0
0.0
-4.0
x
4.0
-3.0
y
3.0
front-default

Figure 1 Gamma function display in CosmoPlayer.

(1) Cutting Plane Control

Cutting plane control allows the user to examine the intersection of a plane with a function surface as the plane moves through the surface. The plane can be parallel to coordinate axis x, y, or z. The intersection curves are projected onto opposite faces of the bounding box, and displayed in a separate pop up window. The user moves the cutting plane by selecting VCR-like buttons to play, stop, reverse, rewind, or fast-forward, or uses the slider bar to move the plane to a specific position and clicks "go" to display the intersection. The calculation of the intersection curves occurs in real time.

Figure 2 shows a cutting plane moving in the z, or vertical axis, direction for the Jacobian elliptic function, dn(x, κ). The cutting plane control panel is below the function on the left, and on the bottom right is the pop up window showing the intersection curves. The axis labels, "x", "n", "z", displayed on the control panel and pop up window correspond to the axes shown in the surface figure. When the user moves the cutting plane with the "play" button, the location of the cutting plane is displayed with the pop up window showing the curves of intersection. When the cutting plane is moved with the slider bar, the location of the cutting plane appears under the slider bar in the control panel. When the "go" button is hit, the intersection curves and cutting plane location appear in the pop up window.

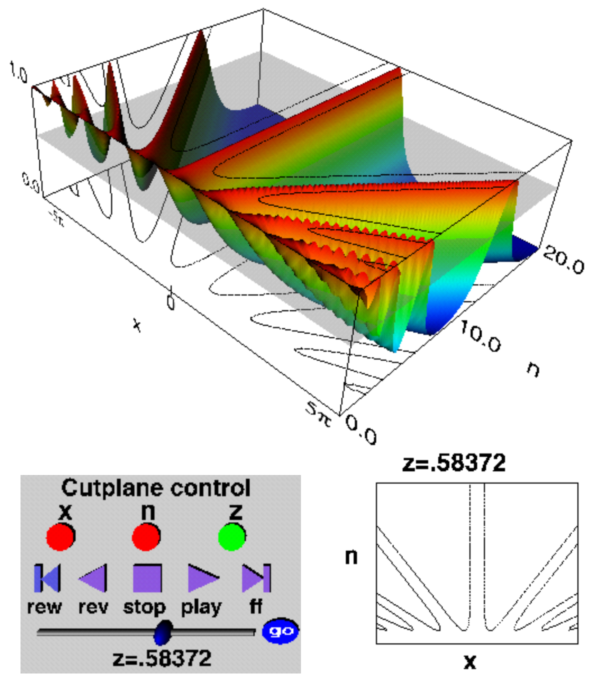


Figure 2 Jacobian elliptic function z = dn(x, κ), for $\kappa = 1 - e^{-n}$, n=0 to 20 and real argument, x, -5π ≤ x ≤5π with cutting plane at z=0.58372.

(2) Axis and Label Control

Using the axis and label control panel, a user can select the style preferred. This may be convenient if the user wishes to do a screen capture of the display for use in a publication, or elsewhere. Figure 3 shows the four axis styles that are available: front, back, triad, and no axes. Both the front and back styles allow the user to enclose the surface with a bounding box.

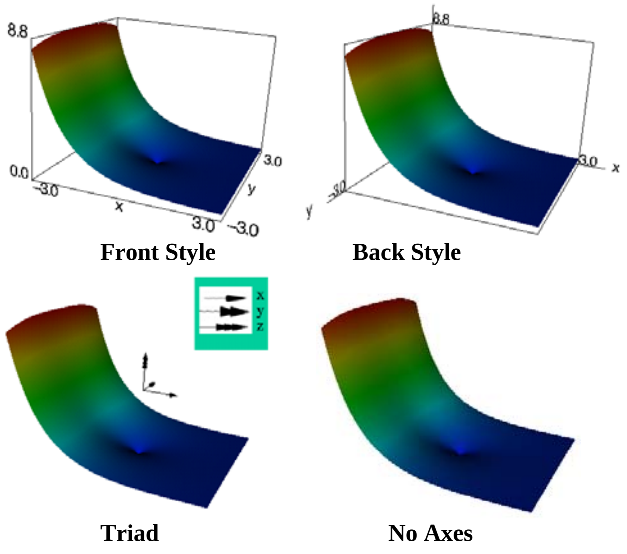


Figure 3 Axis and Labeling Styles

Expressions for mathematical functions often contain special symbols or Greek letters which are not supported by most VRML browsers. When such characters are needed we generate image files using other software and imbed them into the VRML file. Examples of this can be seen in some of the axis labels for the surfaces shown in Figure 10.

(3) Color Map Control

Visualizations in the DLMF represent either real-valued or complex-valued functions of the form, z=f(x, y). For real valued functions only a height based color map, where height = z, is provided. For complex-valued functions, the user has the option of using a height based color mapping where height = |z|, or a mapping based on the phase, or argument, of z. The height based mapping is determined by the following formula where Color is RGB color (red, green blue) and value is the height scaled to lie between 0 and 1, inclusive:

Color = (0, value/0.25, 1), if $0 \leq$ value < 0.25,
Color = (0, 1, (0.5-value)/0.25), if $0.25 \leq$ value < 0.5,
Color = ((value-0.5)/0.25), 1, 0), if $0.5 \leq$ value < 0.75,
Color = (1, (1-value)/0.25, 0), if $0.75 \leq$ value ≤ 1.

Currently the user is given two options for the phase based mapping: four color map or continuous spectrum. In the four color based method, the colors blue, green, red, and yellow are used to indicate whether the phase of z lies in the first, second, third, or fourth quadrant, respectively. The continuous spectrum based method maps the phase information to the color so that 0, π/2, π, and 3π/2 correspond to red, yellow, cyan, and blue with a smooth transition between the colors. Figures 1-3 show height based color mappings. The surfaces in Figures 4 and 5 illustrate the four color and continuous spectrum based mappings.

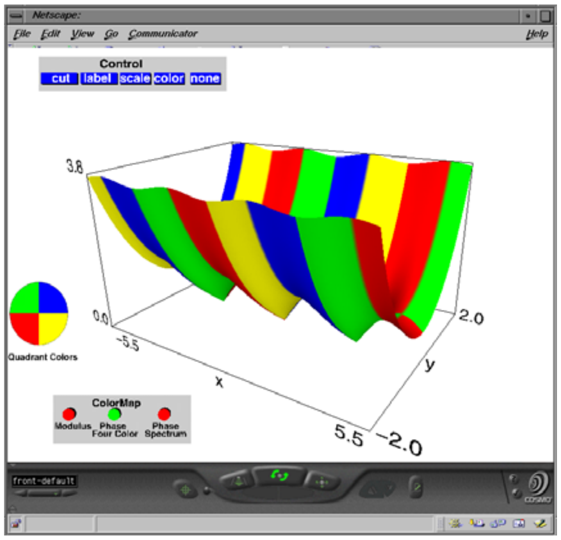

Netscape:
File Edit View Go Communicator Help
Control
cut label scale color none
3.8
0.0
-5.5
x
5.5
-2.0
y
2.0
Quadrant Colors
ColorMap
Modulus Phase Four Color Phase Spectrum
front-default

Figure 4 Modulus of complex sine function with four color map.

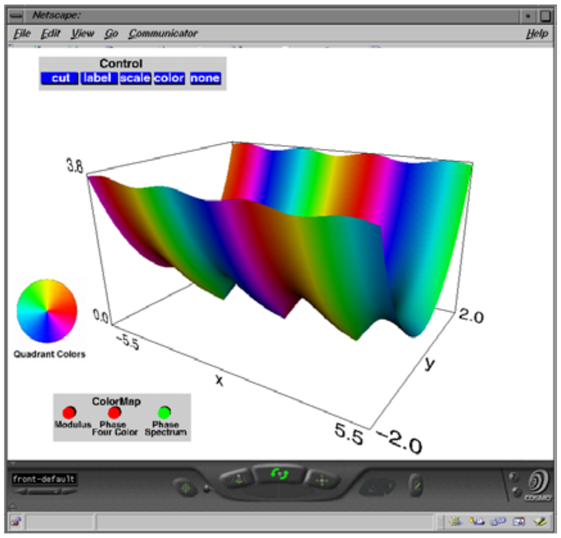

Netscape:
File Edit View Go Communicator Help
Control
cut label scale color none
3.8
0.0
-5.5
x
5.5
-2.0
y
2.0
Quadrant Colors
ColorMap
Modulus Phase Four Color Phase Spectrum
front-default

Figure 5 Modulus of complex sine function with continuous spectrum color map.

(4) Scale Control

The scale control panel has three slider bars that allow a user to change the screen length of the surface in the x, y, or z direction. The "reset" button lets the user change the figure back to its original size. As shown in Figures 6 and 7, scaling the length down to near zero in the z direction produces a simple density plot when the surface color map is based on height, or can brilliantly bring out a branch cut when color represents the argument, or phase, of the function.

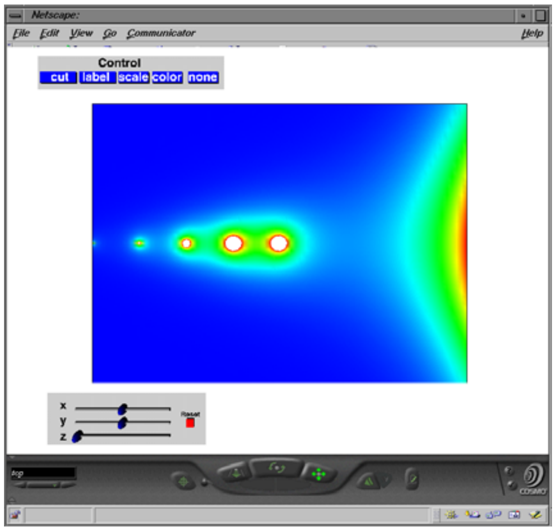

Netscape:
File Edit View Go Communicator Help
Control
cut label scale color none
x
y
z
Reset
top

Figure 6 Modulus of complex gamma function scaled down in the z direction, shown with height based color map.

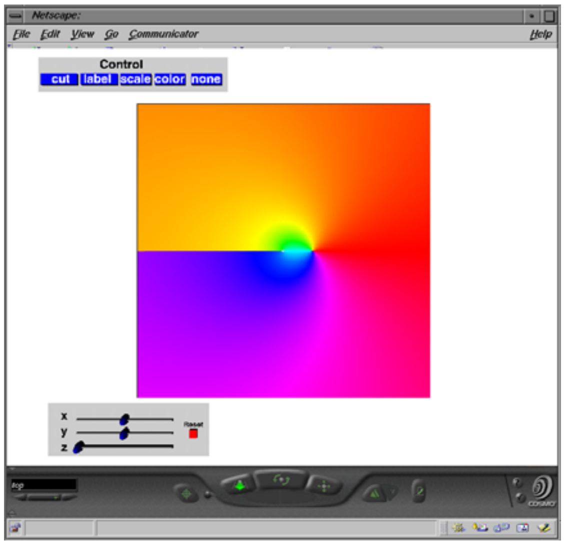

Netscape:
File Edit View Go Communicator Help
Control
cut label scale color none
x
y
z
Reset

Figure 7 Modulus of complex log function scaled down in z direction, shown with phase based color map.

(5) Viewpoints

For the convenience of users, "canned" viewpoints are provided to show the surface from several different directions. Users uncomfortable with the navigation features of the VRML browser can push a button to cycle through the viewpoints. "Front", "back", "right", "left", and "top" viewpoints are provided. The "top" viewpoint is constructed so that the user obtains a simple density plot when he scales the figure down completely in the vertical direction. To ensure that axes and labels always face the user, we defined a script node to recalculate the location of the labels when the user selects another viewpoint. Figure 8 shows several viewpoints of a Hankel function, a special type of Bessel function.

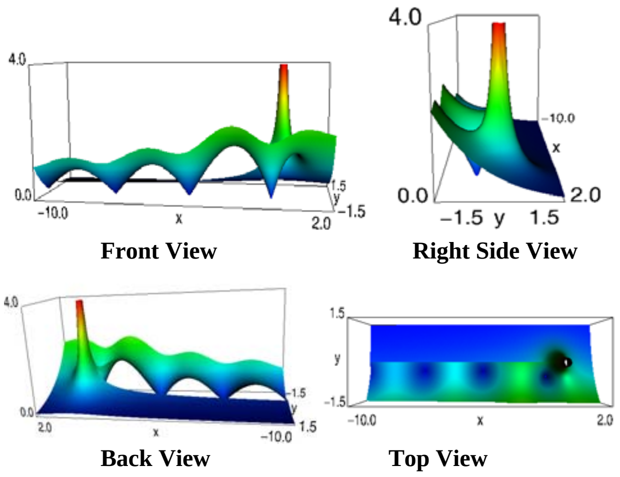


Figure 8 The display of the Hankel function from different viewpoints.

3. The Structure and Implementation of MathViewer

(1) Structure of MathViewer

We first describe the structure and flow of the components that make up MathViewer. The gamma function display in Figure 1, Section 2 suggests that the VRML scene generated by MathViewer can be divided into three primary components: a main control panel, a 3D function surface, and a set of framed coordinate axes with labels. The main control panel links to four subordinate control panels as shown in Figure 9.

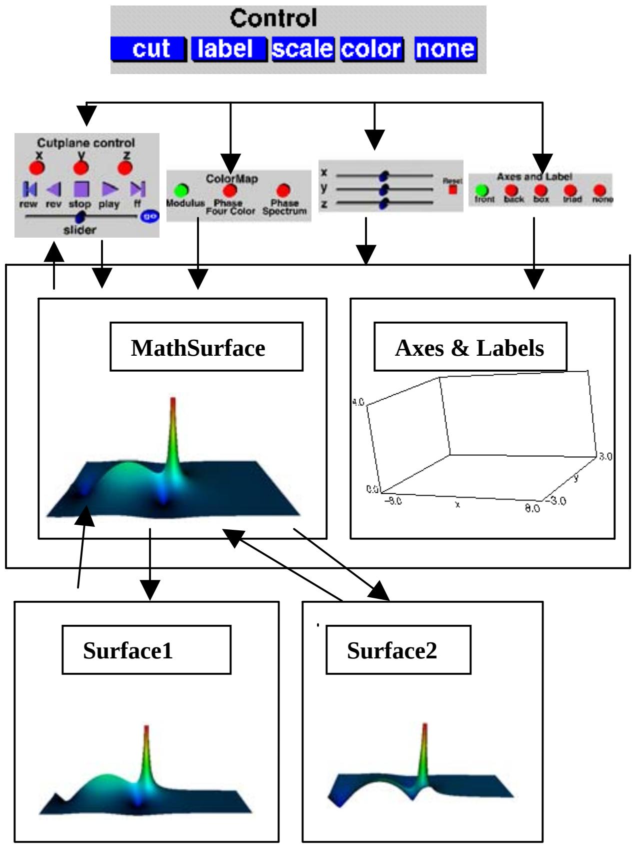
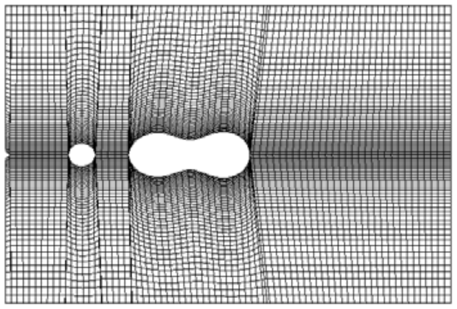


Figure 9 Structure of MathViewer.

The 3D function surface, represented by MathSurface, may be divided into several subsections if the surface is complicated, containing poles, branch cuts or other singularities. Figure 9 shows a surface divided into two sections: Surface1 and Surface2. The data points for the surfaces, i.e., points (x,y,z) where z=f(x,y), are obtained using packages such as Mathematica and Maple, or available Fortran or C codes. To capture the significant features of the function, the computations are done over specially designed grids. Figure 10 shows a grid used to compute the modulus of the gamma function up to the height z=3. The grid consists of two sections split along the line y=0. A discussion of the grid generation techniques used and the challenges involved can be found in the paper [Saunders and Wang 2000].

The axes and labels part of MathViewer is independent of the other components. It displays the coordinate system for the surface, and axes and labels related to cutting plane motion.



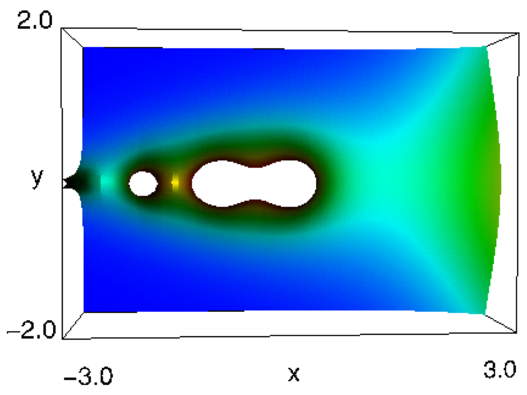


Figure 10 Computational grid and top view of gamma function.

(2) Flow Control

The cutplane control panel sends information about the direction and location of the cutting plane to the "MathSurface" routines that control the appearance of the surface and cutting plane. MathSurface then sends the information to each surface subsection routine. Each surface subsection routine calculates and displays its portion of the cutting plane intersection on opposite faces of the surface bounding box. The subsection routines also send the intersection curve information to MathSurface which combines the information and sends it back up to the cutplane control panel to prompt the display of the intersection curve in the small pop up window when the cutting plane action starts.

As indicated in Figure 9, the flow of information for color map control is quite similar except that no information is sent back up to the color map control panel. Since the scale control panel affects the appearance of both the surface and the coordinate axes, it sends information to MathSurface and to the "Axes and Label" routines. The axes and label control panel affects only the coordinate axes, hence information is sent only to the Axes and Label routines.

(3) VRML Prototypes

For the VRML implementation, we developed several layers of prototypes to create the interactive capabilities for the DLMF visualizations. MathViewer, the top-level prototype, controls the scene and the routes between all the components. The second-level prototypes referenced by MathViewer are MathSurface and AxesLabels. MathSurface controls the route between the subordinate control panels and the subsurfaces. AxesLabels, independent of MathSurface, generates the coordinate axes and labels. MathSurface references the third-level prototype, Surface.

Figure 11 illustrates the parts of the prototype definitions and an example of an instance of MathViewer named DLMF. Note that DLMF contains two instances of Surface called S1 and S2.

```
EXTERNPROTO  MathViewer [
        field       MFNode     surfaces
        field       SFInt32     numOfSets
        field       SFVec3f    world_scale
        field       SFVec3f     world_translation
        field       SFRotation   world_rotation
        field       MFVec3f      axes_point
        field       SFInt32      mapMethod
        field       SFInt32      noPhase
        …
] "dlmf_proto.wrl#MathViewer"
EXTERNPROTO MathSurface [
        field       SFInt32       numOfsets
        field       MFNode        surfaces
        …
] "dlmf_proto.wrl#MathSurface"
EXTERNPROTO Surface [
        field       SFInt32       nx
        field       SFInt32       ny
        field       MFVec3f       points
        field       MFInt32       index
        field       MFColor       colors
        field       MFFloat       phase
        …
] "dlmf_proto.wrl#Surface"

DEF DLMF MathViewer {
        surfaces [
                    DEF S1 Surface {
                    30 50
                    points […]
                    index […]
                    phase […]
                    …
                    }
                    DEF S2 Surface {
                    30 50
                    points […]
                    index […]
                    phase […]
                    …
                    }
        ]
        …
}
```

Figure 11 Prototype of MathViewer

(4) Cutting Plane Implementation

The computation and display of a cutting plane and its intersection with the function surface is implemented in the script node of PROTO Surface in real time. We use linear interpolation to calculate the intersection of planes moving in the x or y direction. The algorithm searches the data in an area around the plane to determine which surface data points are closest to the plane and then performs a linear interpolation to obtain the intersection data points. An algorithm for the calculation of contours is used to compute the intersection curves for cutting planes moving in the z direction.

When surface discontinuities such as branch cuts are present, we can divide the surface into several continuous sections, calculate the intersection for each section and display all the curves obtained. This is illustrated in the picture on the left in Figure 12 which shows discontinuous intersection curves for a cutting plane moving in the x direction through the Hankel function surface. The picture on the right shows a cutting plane that goes through a pole, a singularity where the function diverges to infinity. Actually, here the intersection curve can be drawn as though it is continuous since the break occurs at the top of the bounding box.

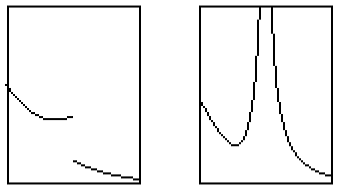


Figure 12 X direction cutting plane intersection curves for the Hankel function.

We developed a C program to generate the VRML file automatically from 3D surface data computed using Mathematica, Maple, and codes provided by the DLMF chapter authors. The C program is flexible enough to handle surfaces of varying complexity. It provides options for the designer to input the number of sub surfaces, scaling factors, special characters, phase data and color map method. Standard or "canned viewpoints" are also calculated by the C program.

4. Applications of MathViewer for the DLMF

To date we have generated more than one hundred VRML visualizations for approximately twenty chapters of the DLMF. Figure 13 shows some examples from several chapters, which illustrate poles, zeroes, discontinuities, and other special features.

In consideration of users who are not familiar with VRML or are unable to obtain a VRML plug-in or browser, we have developed an alternate version of the VRML files, which we can use to generate a QuickTime movie or AVI movie in Parallelgraphics Moviemaker. The only interaction required is a click on the surface to begin the process. Moviemaker records the animation and makes a movie that shows several viewpoints, the movement of the cutting planes in the x,y and z directions, and different color mappings of the surface.

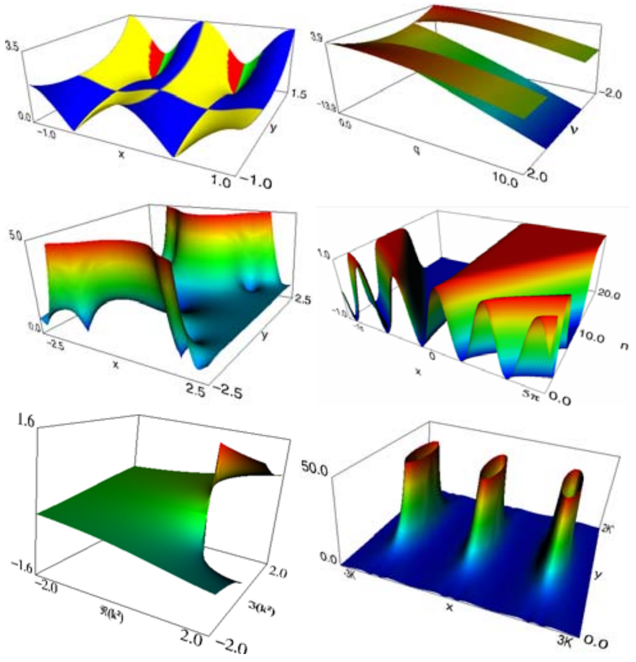


Figure 13 Examples from the DLMF.

5. X3D version of MathViewer.

XML-enabled open standard Extensible 3D (X3D) is the successor to VRML. It enables real time communication of 3D data across network applications. Since the DLMF is expected to be very widely used, we must consider the future of web based 3D technology. Ideally, the VRML graphics in the DLMF should be easily transferred to X3D with all functionalities maintained.

As an initial experiment, we successfully converted the VRML prototypes in MathViewer, and some instances of the MathViewer to X3D format using Vrml97ToX3dNist [Wang]. Vrml97ToX3dNist is a translator developed by one of the authors at NIST, and has been incorporated into the Web3D Consortium's open source X3D editor. The X3D files were tested using the Bitmanagement BS Contact viewer. The results demonstrated that all main functions are maintained. Figure 14 shows a display of X3D file Ai.x3d on the BS Contact viewer with the main menu, Airy function surface |Ai(x+iy)|, the cutting plane control panel, and z direction cutting plane with intersection curves.

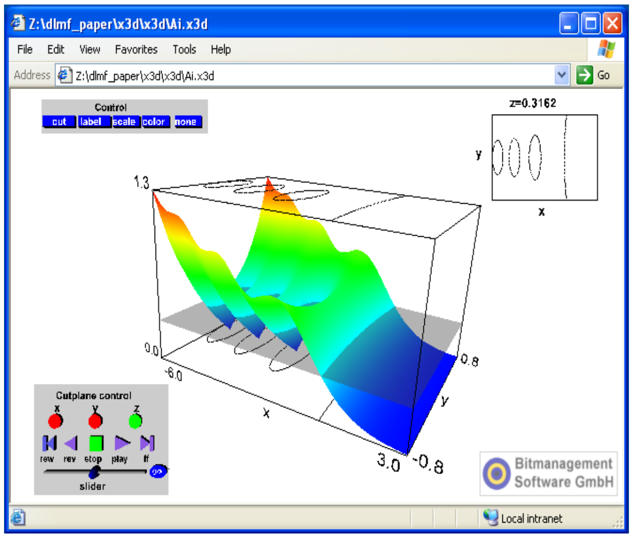

Z:\dlmf_paper\x3d\x3d\Ai.x3d
File Edit View Favorites Tools Help
Address Z:\dlmf_paper\x3d\x3d\Ai.x3d
Go
Control
cut label scale color none
z=0.3162
y
x
1.3
0.0
-6.0
x
3.0
-0.8
0.8
y
Cutplane control
x y z
rew rev stop play ff
slider
Bitmanagement Software GmbH
Local intranet

Figure 14 Display of Airy function using X3D file.

6. Conclusions

Web-based 3D visualizations for the NIST Digital Library of Mathematical Functions have been developed using VRML/X3D standards. In particular, a VRML prototype called MathViewer was developed and implemented to provide DLMF users with interactive capabilities that will allow them to explore the interesting features of complex mathematical function surfaces.

Over one hundred visualizations have been completed for the DLMF and feedback from DLMF editors, authors, and other observers has been positive. However, we have found that VRML browser portability is a problem that must be continually addressed as browser availability or operating systems change. We have found that some capabilities work in one browser, but not in others, but we have tried, with varying degrees of success, to make MathViewer run in commonly used browsers such as CosmoPlayer, Cortona, and Blaxuun Contact among others.

We continue to improve existing features and add new ones. One new feature being tested is to provide the user with the ability to click anywhere on the function surface and obtain the coordinates. Also, we are searching for ways of improving the real time calculation of the cutting plane intersection in the z, or vertical, direction. The calculation is based on a contour computing algorithm but its performance is too slow to provide a smooth animation of the cutting plane. One solution is to forgo the real time calculation, generate the contour curves off line and store them in the VRML file. We have successfully used VRML/X3D standards to develop web-based visualizations for the DLMF, but the key advantage of this work is that it can be enhanced and modified over time to continue to take advantage of cutting-edge 3D technology.

Acknowledgements

We would like to thank Daniel Lozier, Bruce Miller, Sandy Ressler for their very helpful suggestions and support for this work.

Disclaimer

All references to commercial products are provided only for clarification of the results presented. Their identification does not imply recommendation or endorsement by NIST.

References

Abramowitz, M. and Stegun, I.A. editors 1964. Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables, Vol. 55, National Bureau of Standards Applied Mathematics Series. U.S.Government Printing Office, Washington, D. C., 1964.

Lozier, D.W. 2002. The NIST Digital Library of Mathematical Functions Project, *Annals of Mathematics and Artificial Intelligence*.

Saunders, B. and Wang, Q. 2000. From 2D to 3D: Numerical Grid Generation and the Visualization of Complex Surfaces, *Numerical Grid Generation in Computational Field Simulations***.** Also , *NISTIR 6555, National Institute of Standards and Technology*.

Wang, Q http://ovrt.nist.gov/v2_x3d.html