Techniques for the Interactive Visualization of Volumetric Data

Gregory M. Nielson   Bernd Hamann

Computer Science Department
Arizona State University
Tempe, AZ 85287-5406

Abstract

Some ideas and techniques for visualizing volumetric data are introduced. The methods presented are quite different form either "volume rendering" techniques or "surface contour" methods. All of the methods are conceptually quite simple and rather easy to implement. In addition, they are intended to be used interactively.

1.0 Introduction

The purpose of this paper is to introduce some ideas and techniques for visualizing volumetric data. The methods presented here are quite different form either "volume rendering" techniques or "surface contour" methods. All of the methods we present are conceptually quite simple and rather easy to implement. In addition, they are intended to be used interactively.

Volumetric data is data with a domain of three independent variables. A example of such data would be temperature measurements taken at various positions in a furnace. The notation that we use to represent this type of data is

$$(F_i, x_i, y_i, z_i), \quad i = 1, \ldots, N. \qquad (1.1)$$

Here the three independent variables are those of position and the dependent variable is a simple scalar value. While many of our applications are like this, in that we have a portion of physical space as a domain, this is not required for the techniques we discuss here. The independent variables do not have to indicate a position in space and can be abstract in the sense that they can represent any quantity. For example we may have the data (P_i, s_i, r_i, t_i), $i = 1, \ldots, N$, where P_i is an index of the level of performance which depends on the amount of money expended, s_i, the number of people allocated to the project, r_i and time, t_i.

In this paper we cover only the case where the dependent data is a single scalar. It is a much more challenging problem to visualize multivariate, dependent data. For example, in three dimensional flow analysis, a velocity vector is given at each sample location in space and to effectively visualize such data is a difficult problem (cf. Dickinson[3], Helman and Hesselink[11], and Hibbard and Santek[13]). The techniques discussed here do not apply to the situation of multivariate range variables, except that these variables can be "uncoupled" and treated as a collection of scalar valued relationships; but this is generally not a good idea.

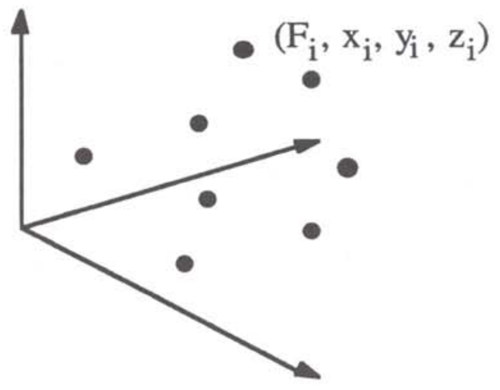

Figure 1. Volume Data

Often we will find it useful to view the dependent data as coming from the evaluation of an underlying function F so that we have $F_i = F(x_i, y_i, z_i)$, $i = 1, \ldots, N$. For example this gives us immediately the notation $\partial F/\partial x$, $\partial F/\partial y$ and $\partial F/\partial z$ to indicate the various partial derivative with respect to the independent variables. In actuality when we have discrete sampled data and partial derivatives are of interest, we will have to estimate them by divided differences or by some other means such as fitting a trivariate function to the sampled data and then computing the derivative of this approximation. For some discussion of some methods of fitting trivariate scattered data, the reader is refered to Alfeld[1], Nielson and Dierks[24], Foley[5] and Franke and

Nielson[8]. If a trivariate function, F, defined over the domain cube

$$C=\{(x, y, z) : x_0 \le x \le x_n,\ y_0 \le y \le y_n,\ z_0 \le z \le z_n\} \qquad (1.2)$$

has been fit to the scattered or "unstructured" data, the relationship can be sampled on a grid so as to yield the data

$$(F_{ijk}, x_i, y_j, z_k) ;\ i = 1, \ldots, Nx;\ j = 1, \ldots, Ny;\ k = 1, \ldots, Nz \qquad (1.3)$$

where

$$F_{ijk} = F(x_i, y_j, z_k)$$

We call this type of data *cuberille grid data* and it is exactly this type of data for which the methods of this paper apply.

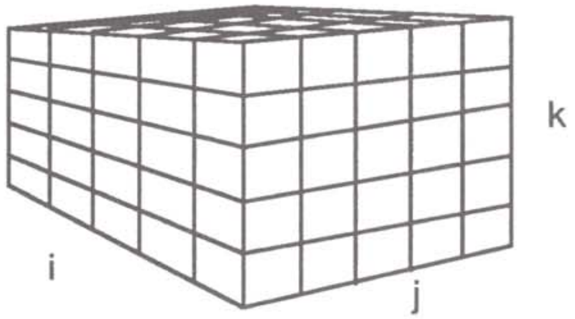

Figure 2. Cuberille Grid Data

In Sections 2 - 5, we describe a collection of techniques and ideas for "graphing" cuberille grid data. All of these techniques are quite simple and rather easy to implement. During the development of these techniques, we have been particularly concerned with allowing the user to interact with the system in order to interrogate and analyze the relationships indicated by the volumetric data.

As a final note of introduction, we mention two additional types of methods for visualizing volumetric data which we do not cover. Both are very important and much more profound than the methods discussed here. Volume rendering is a relatively new technique which is based upon accumulating intensities along rays cast through the cube of data (cf. See Figure 3). A variety of methods for doing this have recently appeared in the literature (cf. Drebin, Carpenter and Hanrahan[4], Levoy[19], Levoy[20], Levoy[21], Sabella[27], Keeler and Upson[33], Smith[29], Kaufman and Bakalash[17], Kaufman[16], Frieder, Gordon and Reynolds[9], Höhne, Bomans, Tiede and Riemer[14], Kajiya and Von Herzen[15]). Regardless of which particular method is used, one major research goal is to develop the means of accomplishing these volume rendered images in real time. We refer the reader to Foley, Lane and Nielson[7] for a technique which is based upon the coherence of images and uses interpolation techniques to accomplish near real time speeds on conventional workstations. A typical volume rendered image and its "real time" approximation computed by the image interpolation technique is shown in Figure 4.

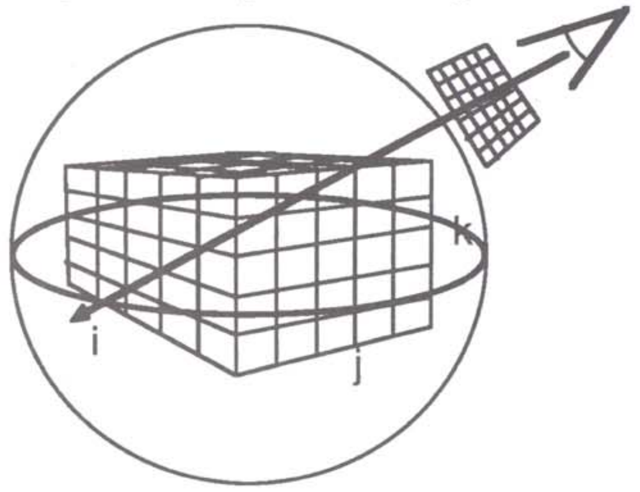

Figure 3. Volume Rendering

The second type of method we do not cover here are contour methods. The contour of a trivariate relationship is a surface and consequently these methods are sometimes called "surface based methods" (cf. Fuchs, Levoy and Pizer[10] or Artzy, Frieder and Herman[2]). Often the contour is computed as a polyhedron and this can result in a tremendously large number of surface facets. For example, the contour surfaces of Figure 5 contains approximately 50,000 triangles. In this case, the cuberille data consisted of 64 X 64 X 68 = 278,528 data points.

2.0 The Tiny Cubes Method

The basic idea of the **tiny cubes** technique is to place objects in the domain volume whose color is determined by the value of F at the location of the object The objects can be almost anything, but spheres and cubes most readily come to mind. We have concentrated our implementations on the case of cubes. Three resolution parameters are specified by the user: Nx, Ny and Nz. There will be a total of Nx•Ny•Nz color coded cubes which are displayed. In addition, the user specifies a value for the parameter, M, which controls the amount of open space and consequently the size of the cubes which are displayed. We let the width, length and height of each "tiny" cube be denoted by (Dx, Dy, Dz). The

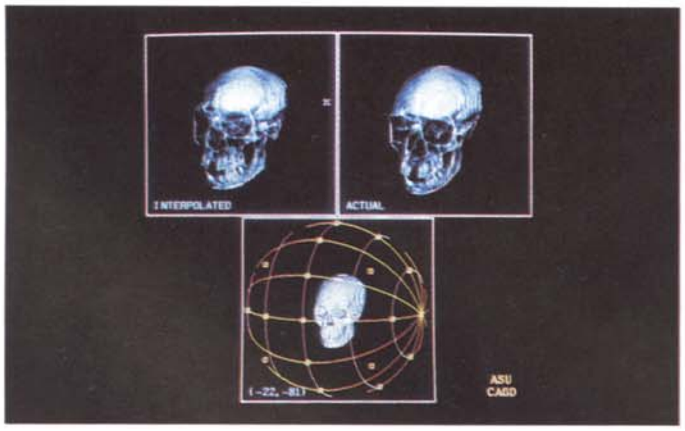

Figure 4. Volume rendered image and image interpolation approximation

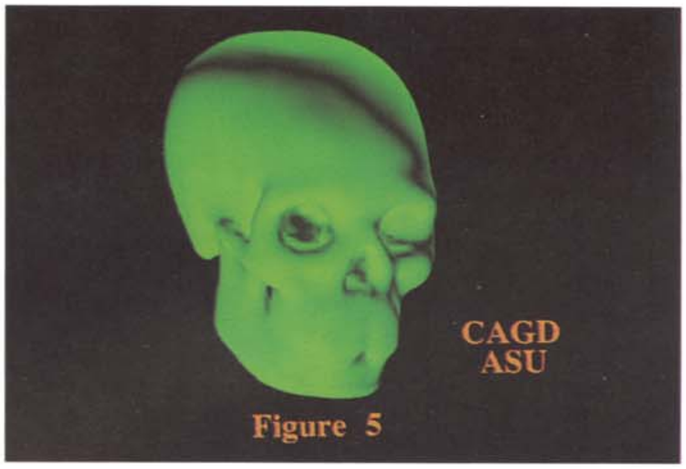

Figure 5. Surface Contour

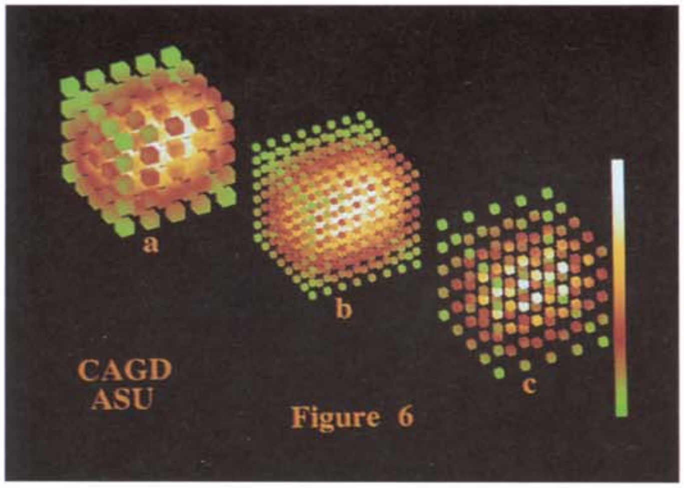

Figure 6. The Tiny Cubes Method

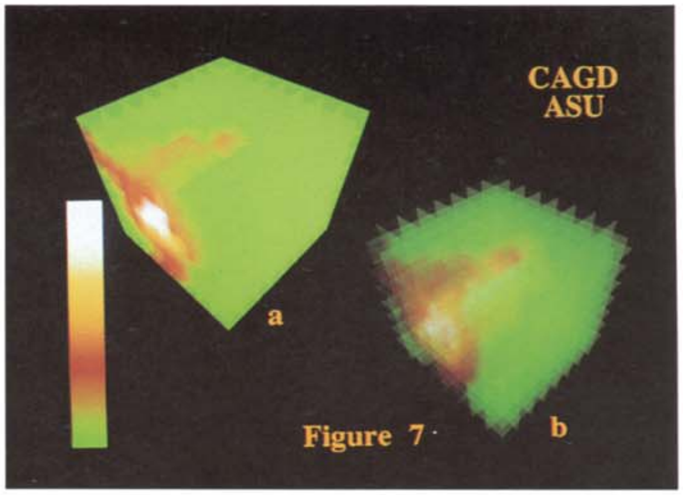

Figure 7. The Vanishing Cube Method

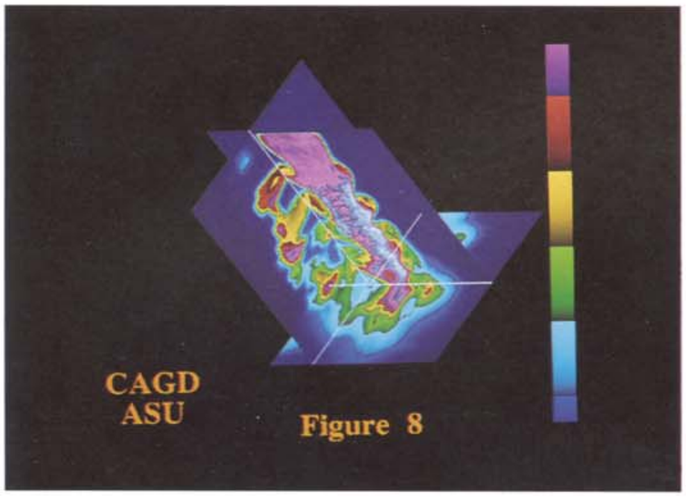

Figure 8. Color coded contour slice technique.

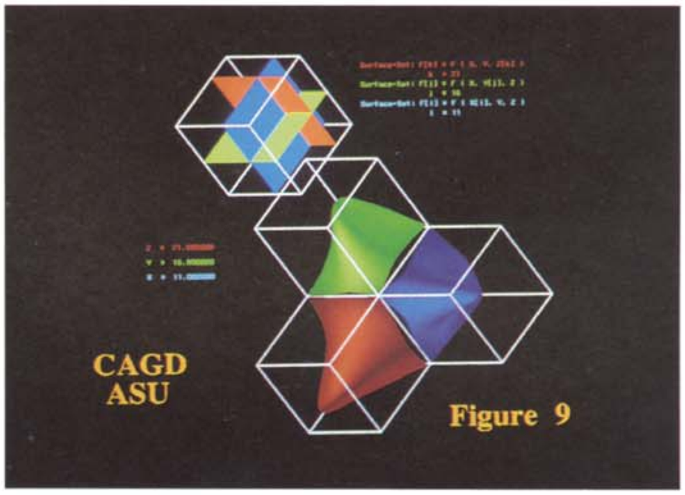

Figure 9. The surface slice method

lower-left-front corner of each cube is given by the coordinates

$$\mathbf{X}_i = x_0 + (i-1)Dx(M+1), \quad i = 1, \ldots, Nx$$

$$\mathbf{Y}_j = y_0 + (j-1)Dy(M+1), \quad j = 1, \ldots, Ny$$

$$\mathbf{Z}_k = z_0 + (k-1)Dz(M+1), \quad k = 1, \ldots, Nz$$

where $(\mathbf{X}_0, \mathbf{Y}_0, \mathbf{Z}_0)$ is the lower-left-front corner of the whole domain and

$$Dx = \frac{x_{max} - x_{min}}{Nx(M+1) - M},$$

$$Dy = \frac{y_{max} - y_{min}}{Ny(M+1) - M},$$

$$Dz = \frac{z_{max} - z_{min}}{Nz(M+1) - M}.$$

The function value, F_{ijk} and the particular color table used will determine the color that is used at each vertex. The faces of the cubes are then Gouraud shaded. Either the original cuberille grid includes the values $(\mathbf{X}_i, \mathbf{Y}_j, \mathbf{Z}_k)$ or we use some interpolation (eg. trilinear) scheme to compute estimates. For each graph of this type, we end up having to display a total of 6(Nx•Ny•Nz) rectangles. We have also found it useful to display lines connecting the centers of the cubes. Example images which illustrate our implementation are shown in Figure 6. Figure 6a illustrates the case where Nz = Ny = Nz = 5; M = 1; Figure 6b, the case where Nx = Ny = Nz = 8; M = 2 and Figure 6c the case where Nx = Ny = Nz = 5; M = 3. The cuberille grid data for these examples were obtained by evaluating the function

$$\begin{aligned}F(x, y, z) = 15\exp(-.005[\,(x-10)^2 + (y-10)^2 + (z-10)^2]) \\ + \exp(\,-.0025[\,(x-15)^2 + (y-20)^2 + (z-20)^2]) \\ + \exp(\,-.005[\,(x-25)^2 + (y-25)^2 + (z-25)^2])\}\end{aligned}$$

and the domain is $\{\,(x, y, z) : 0 \le x \le 39, 0 \le y \le 39, 0 \le z \le 39\,\}$. An important feature of this method is to be able to interactively rotate the graph in order to view different portions. This limits the size of the resolution parameters, but this often will not be a problem since the resolution parameters are also limited by the complexity of the image. We have found that for resolution values beyond, say 15, the image is too complex to be understood.

3.0 The Vanishing Cube Method

Like the methods of the previous section, the **vanishing cube** method associates a color with each data location, (x_i, y_j, z_k) and this color is based upon the value of dependent variable, F_{ijk} and the particular color table used. More precisely,

$$\begin{aligned}C_{ijk} &= (R_{ijk}, G_{ijk}, B_{ijk}) \\ &= (\,R(F_{ijk}), G(F_{ijk}), B(F_{ijk})\,)\end{aligned}$$

where i = 1, . . . , Nx, j = 1, . . . , Ny, and k = 1, . . . , Nz and the function notation R( ), B( ) and G( ) is used to represent the color code table. For example the table used for the example of Figure 8 is

index	R	G	B
i=0-255	i	255-i	0
i=256-511	255	i-256	0
i=512-767	255	255	i-512

Table 1. Color Table for Example of Figure 11

Once we have a color for each vertex, we can entirely color any of the planes parallel to the axes by using linear (or Gouraud) shading on each of the rectangles which comprise the plane. There are Nx•Ny•Nz rectangles perpendicular to each axes for a total of 3(Nx•Ny•Nz) rectangles to be displayed. Of course, if we directly display these rectangles, all that will be viewed are those on the outer faces. In order to "see in" we compute an image based upon a simple model of transparency for the rectangles. The rectangles are all sorted by distance from the viewpoint and then displayed from back to front using an α-buffer. The data for the example of Figure 7 were provided to us by Marshall Long and represents gas concentrations from an acoustically-driven forced flow. In Figure 7a, Nx = 8 and t = 0.5 and in Figure 7b, Nx = 8 and t = 0.95. While interaction is possible as far as varying the transparencey factor, it is not possible to rotate the graph in real time because of the need to sort the rectangles. Possible some clever way of using coherence to update the sorts could lead to real-time speeds.

4.0 Slice Methods

The basic idea of **slice** methods is to simultaneously display three rectangular grid data sets, each one obtained by taking a slice through the domain by holding one of the independent variables fixed. In function notation terms, we simultaneously display some type of graph of the three bivariate relationships,

$F_x(y,z)=F(\mathbf{x},y,z)$, $y_{min} \le y \le y_{max}$; $z_{min} \le z \le z_{max}$

$F_y(x,z)=F(x, \mathbf{y},z)$, $x_{min} \le x \le x_{max}$; $z_{min} \le z \le z_{max}$

$F_z(x,y)=F(x,y,\mathbf{z})$, $x_{min} \le x \le x_{max}$; $y_{min} \le y \le y_{max}$

The user is then allowed to interactively vary the fixed (but arbitrary) point (**x**, **y**, **z**). This can be done in a convenient manner with an ordinary 2D-mouse by using the "triad mouse" described in [Nielson and Olsen, 1986].

In our first implementation of this method, we used a color coded contour method for displaying the graphs of the rectangular grid. Color values at each of the vertices (x_i, y_j, z_k) are obtained by using the color table function C(x, y, z) and then smooth shading is used to color the entire plane. In Figure 8 we show an example. The size of this data set is Nx =80, Ny =130 and Nz = 20 and is the same as that used in the examples of Figure 7 . The color table is shown in the color bar to the far right.

For our next slice method we use a smooth shaded surface in order to display the three rectangular grid data sets. These three sets of surfaces could be located anywhere in the image, but we have found it convenient to have each of these graphs located on the face of a cube. We scale the values so that the minimum value is represented by a point on the cube and the maximum value is one unit in the direction normal to this plane. Also, we have found it useful to use different colors for each of these graphs and to display information indicating the value of (x, y, z) for the display graphs. This we accomplish by displaying three mutually perpendicular planes with colors associated in the proper manner. This method is illustrated in Figure 9.

Acknowledgements

This work was done on a Silicon Graphics 4D/220 GTX and a Stardent Titan. This work was supported by the U.S. Department of Energy under contract DE-FG02-87ER25041 to Arizona State University and by a NATO research grant RG 0097/88. We wish to thank the members of the CAGD group at ASU for their help and support.

References

1. Alfeld P., "Scattered Data Interpolation in Three or More Variables," in *Mathematical Methods in Computer Aided Geometric Design*, Lyche T. and Schumaker L., editors, Academic Press, New York, 1989, pp. 1-33.

2. Artzy E., Frieder G., and Herman G.T., "The Theory, Design, Implementation, and Evaluation of a Three-Dimensional Surface Detection Algorithm," *Computer Graphics and Image Processing*, Vol. 15, 1981, pp. 1-24.

3. Dickinson R.R., "A Unified Approach to the Design of Visualization Software for the Analysis of Field Problems," in *Three-Dimensional Visualization and Display Technologies, SPIE Proceedings*, Vol. 1083, Jan. 1989.

4. Drebin R.A., Carpenter L., and Hanrahan P., "Volume Rendering," SIGGRAPH 88 Conference Proceedings, *Computer Graphics*, Vol. 22, No. 4, Aug. 1988, pp. 65-74.

5. Foley T.A., "Interpolation and Approximation of 3-D and 4-D Scattered Data," *Computer and Mathematical Applications*, Vol. 13, No. 8, 1987, pp. 711-740.

6. Foley, T. A. and Lane, D., "Visualization of irregular multivariate data," these proceedings.

7. Foley, T. A., Lane, D and Nielson, G., "Towards Animating Ray-Traced Volume Visualization", to appear in Visualization and Computer Animation Journal, July, 1990.

8. Franke R. and Nielson, G. "Scattered Data Interpolation and Applications: A Tutorial and Survey", in Geometric Modelling: Methods and Their Application, H. Hagen and D. Roller (eds.), Springer, 1990.

9. Frieder G., Gordon D., and Reynolds R.A., "Back-to-Front Display of Voxel-Based Objects," *IEEE Computer Graphics and Applications*, Vol. 5, No. 1, Jan. 1985, pp. 52-59.

10. Fuchs H., Levoy M., and Pizer S.M., "Interactive Visualization of 3D Medical Data," *Computer*, Vol. 22, No. 8, Aug. 1989, pp. 46-51.

11. Helman J. and Hesselink L., "Representation and Display of Vector Field Topology in Fluid Flow Data Sets," *Computer*, Vol. 22, No. 8, Aug. 1989, pp. 27-36.

12. Herman G.T. and Udupa J.K., "Display of 3D Digital Images: Computational Foundations and Medical Applications," *IEEE Computer Graphics*

and Applications, Vol. 3, No. 5, Aug. 1983, pp. 39-46.

13. Hibbard W. and Santek D., "Visualizing Large Data Sets in the Earth Sciences," *Computer*, Vol. 22, No. 8, Aug. 1989, pp. 53-57.

14. Hoehne K.H., Bomans M., Tiede U., and Riemer M., "Display of Multiple 3D-Objects Using the Generalized Voxel Model," *Medical Imaging II, Proceedings of SPIE*, Vol. 914, 1988, pp. 850-854.

15. Kajiya J. and Von Herzen B., "Ray Tracing Volume Densities," SIGGRAPH 84 Conference Proceedings, *Computer Graphics*, Vol. 18, No. 3, 1984, pp. 165-173.

16. Kaufman A. (ed.), Volume Visualization, IEEE Computer Society Press, Los Alamitos, 1990.

17. Kaufman A. and Bakalash R., "Memory and Processing Architecture for 3D Voxel-Based Imagery," *IEEE Computer Graphics and Applications*, Vol. 8, # 6, Nov. '88, pp. 10-23.

18. Lenz R., Gudnumdsson B., Lindskog B., and Danielsson P.E., "Display of Density Volumes," *IEEE Computer Graphics and Applications*, Vol. 6, No. 7, July 1986, pp. 20-29.

19. Levoy M., "Display of Surfaces from Volume Data," *IEEE Computer Graphics and Applications*, Vol. 8, # 3, May '88, pp. 29-37.

20. Levoy M., "Volume Rendering by Adaptive Refinement," Technical Report 88-030, Computer Science Department, University of North Carolina at Chapel Hill, June 1988.

21. Levoy M., "Efficient Ray Tracing of Volume Data," Technical Report 88-029, Computer Science Department, University of North Carolina at Chapel Hill, June 1988.

22. Long M.B., Lyons K., and Lam J.K., "Acquisition and Representation of 2D and 3D Data from Turbulent Flows and Flames," *Computer*, Vol. 22, #8, Aug. 1989, pp. 39-45.

23. Lorensen W.E. and Cline H.E., "Marching Cubes: A High-Resolution 3D Surface Construction Algorithm," SIGGRAPH 87 Conference Proceedings, *Computer Graphics*, Vol. 21, No. 4, July 1987, pp. 163-169.

24. Nielson, G. and Dierks, T. "Analysis and Visualization of Scattered Volumetric Data, Submitted

25. Nielson, G. M. and Olsen, D. R., "Direct Manipulation Techniques for 3D Objects Using 2D Locator Devices", in *Proceedings of the Workshop on Interactive 3D Graphics*, ACM, New York, 1986, pp. 175-183.

26. Rosenblum L.J., "Visualization of Experimental Data at the Naval Research Laboratory," *Computer*, Vol. 22, # 8, Aug. 1989, pp. 95-101.

27. Sabella P., "A Rendering Algorithm for Visualizing 3D Scalar Fields," SIGGRAPH 88 Conference Proceedings, *Computer Graphics*, Vol. 22, No. 4, Aug. 1988, pp. 51-55.

28. Sloan K.R., Jr. and Hrechanyk L.M., "Surface Reconstruction from Sparse Data," in *Proceedings of the IEEE Conference on Pattern Recognition and Image Processing*, IEEE Computer Society Press, Los Alamitos, California, August 1981, pp. 45-48.

29. Smith A., "Volume Graphics and Volume Visualization, A Tutorial," Pixar Technical Memo 176, Pixar, San Rafael, California, May 1987.

30. Staudhammer J., "Supercomputers and Graphics," *IEEE Computer Graphics and Applications*, July 1987, pp. 24-25.

31. Tufte E.R., *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, Connecticut, 1983.

32, Upson, C. (ed.) , Proceedings of the Chapel Hill Workshop on Volume Visualizaiton, Chapel Hill, NC, May 1989.

33. Upson C. and Keeler M., "VBuffer: Visible Volume Rendering," SIGGRAPH 88 Conference Proceedings, *Computer Graphics*, Vol. 22, No. 4, Aug. 1988, pp. 59-64.

34. Wang P.C.C., editor, *Graphical Representation of Multivariate Data*, Academic Press, New York, 1978.

35. Wolfe R.H., Jr. and Liu C.N., "Interactive Visualization of 3D Seismic Data: A Volumetric Method," *IEEE Computer Graphics and Applications*, Vol. 8, No. 4, July 1988, pp. 24-30.