Projection-based Data Level Comparisons of Direct Volume Rendering Algorithms

Kwansik Kim and Alex Pang

UCSC-CRL-97-16
July 18, 1997

Baskin Center for
Computer Engineering & Information Sciences
University of California, Santa Cruz
Santa Cruz, CA 95064 USA

ABSTRACT

The focus of this paper is on comparative visualization tools for scientists and also on a mechanism for communicating their results on the web. Direct volume rendering (DVR) is one of the most popular methods for visualizing 3D scientific data including medical and molecular volumetric data. Since DVR is a relatively expensive method, much research has focused on making it faster as well as making it work with various grid structures. Unfortunately, this has lead to a host of DVR algorithms that produce images that are slightly different from each other. In situations where the differences in the resulting DVR images makes a difference – for example, in medical imaging where slight difference may lead to a mis-diagnosis, or in molecular studies where it is important to understand the binding strengths, visualization users must understand the limitations of each DVR algorithm. Previous effort in highlighting these differences have focused on image level comparisons. This paper advocates for data level comparison. That is, we take advantage of intermediate 3D information available during the rendering process to do the comparison. The main benefit of this approach is that it not only allows us to identify the regions and extents of differences among DVR algorithms, it also provides us a mechanism for explaining why they are different. One of the main challenges and contribution of this work is in finding common bases for comparing different DVR algorithms. In this paper, we describe the projection based algorithms as the basis for comparison. Using this basis, different comparison metrics are derived and used to evaluate different DVR algorithms. We illustrate the effectiveness of this approach using several case studies. The second focus of this paper is on communicating these results on the web. We provide a web interface for users to select different DVR algorithms and parameters for comparison, as well as different view angles and a set of volumetric data sets. Since the results of data level comparisons may be images or 3D analyses, the comparisons are sent as either GIF images and/or VRML files.

Keywords: Scientific visualization, uncertainty, error, difference, similarity, metrics, WWW, VRML.

1 Introduction

Direct volume rendering (DVR) is one of the most popular methods for visualizing 3D scalar data sets. It generates images directly from the data values without creating intermediate geometric representations. The basic idea behind DVR is the simulation of light interaction with matter [1, 2, 3]. DVR is also a view-dependent approach requiring recalculation each time the view point is changed. Because of the view-dependent nature and the calculations involved with reasonably sized 3D data sets, DVR is a relatively expensive approach. This has in turn spurned numerous research with the general goal of speeding up the process without sacrificing the image quality.

Unfortunately, this plethora of DVR methods produce images that are different from each other. In critical applications such as clinical medical imaging where DVR is the method of choice, this can be very disconcerting. Fortunately, more and more DVR papers address the issue of image quality. But in those that do, the norm is to use image level comparisons, and sometimes at the image summary level at best (e.g. root mean square measure). There are inherent limitations to image level comparisons. For example, while image level comparisons can provide information as to the location and degree by which two images differ, they do not provide any information as to why the two images differ. This paper addresses this shortcoming by proposing the use of data level comparison techniques. The goal is that if two images differ in a significant manner (e.g. presence or absence of a tumor from two DVR images), we want to provide an explanation of the cause(s) for such differences in terms of how the images were generated by different DVR algorithms.

2 Image Level Comparison

Most work in comparing DVR images are performed at the image level. The most popular method in this category is probably side-by-side comparison. Other methods include difference images, frequency domain analysis strategies, image processing based methods such as contrast stretching, vision based methods such as auto-correlation and optical flow fields, and summary image statistics which provide an aggregate measure such as root mean square (RMS) calculations. All these methods use images as their starting point for comparison.

In the context of comparing DVR images, the main advantage of image level methods is their flexibility. For example, it is just as easy to compare a ray-based against another ray-based DVR image as it is to compare images from ray-based against a projection-based or transform-space algorithm. (See Figure 2.1). Its main drawback is that it is operating at the image level and hence has lost all the 3 dimensional information from intermediate calculations. Furthermore, images may need additional processing to register them or to reduce image distortions prior to performing image level comparison. Finally, if the differences are very small, image level comparisons are not as effective. One should also be aware of the limitations of summary statistics derived from images. It is possible to produce cases where the summary statistics are the same, but the images are obviously different [4].

3 Data Level Comparison

The name data level comparison was inspired by the work of Trapp and Pagendarm [5] where they used it in computational fluid dynamics (CFD) applications. Data level methods incorporate intermediate and auxiliary information available during the rendering process and use these information to generate a data level comparison image.

In DVR, the intermediate information may include items related to the data values or to the volume rendering algorithm. For example, distribution of cumulative opacities, feature or similarity vector of values that contributed to a rendered pixel, and maximal or minimal values along a ray are examples of information related to data values. On the other hand, transfer functions, ray sampling locations and frequency, opacity threshold, and projection filters are examples of information related to the volume rendering algorithm. It should be noted that in some cases this distinction is blurred. In either case, these information and others can be used in metrics for generating data level comparisons which should provide more in depth analysis than is possible with image level comparisons.

The main motivation for data level comparisons is to provide more in-depth comparison of

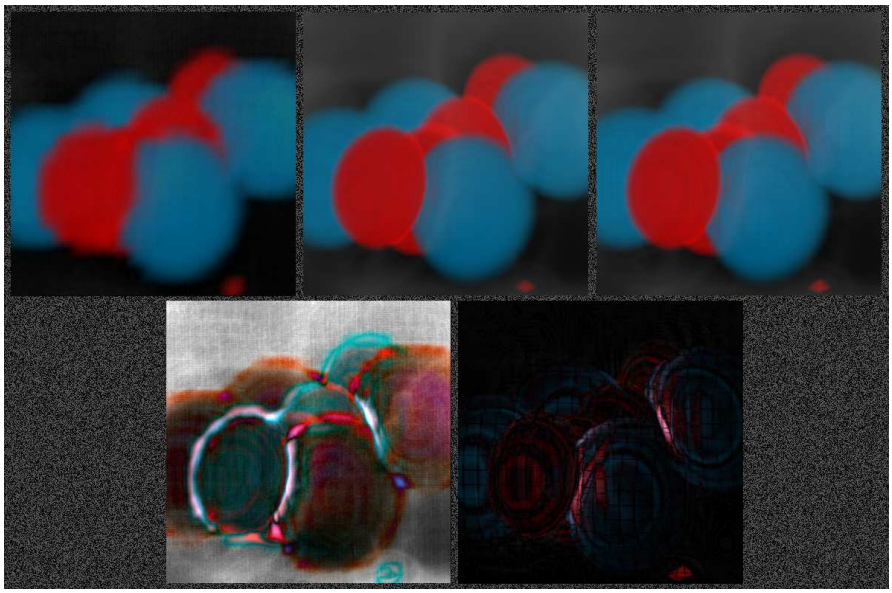

Figure 2.1: Example of image level comparisons side-by-side (top row) and difference images (bottom row). The top row shows results from three different DVR algorithms: cell projection, ray casting with regular tri-linear sampling, and ray casting with bi-linear sampling at cell faces. The bottom row shows difference images between the projection and regular sampling methods, and between the regular ray samplings and the cell face intersection samplings. Intensity indicates amount of difference, while hues are determined by the signed difference in each color bank. The images does not provide any explanation for the horizontal and vertical striping artifacts apparent in the bottom right image. The Hipip (HIgh Potential Iron Protein) data is a 64^3 scalar volume representing the quantum mechanics calculation of a one-electron orbital of a four-iron, eight-sulfur cluster found in many natural proteins, courtesy of Louis Noodleman and David Case, Scripps Clinic, La Jolla, California.

different DVR algorithms particularly in cases where the differences makes a difference. A case in point is the potential mis-diagnosis of the presence or absence of a tumor. Using image level comparisons, it is impossible to determine the reasons for discrepancies among different DVR algorithms. On the other hand, a data level approach might reveal the reason as the rays not penetrating far enough into the volume, or perhaps the sampling step is too large and the tumor was completely stepped over by the latter method, or the projection filter improperly diffusing critical cell contributions over a large or perhaps sub-pixel area.

The key point of data level comparison is the use of intermediate information available and/or that might have contributed to the resulting image. It does not preclude the use of of traditional methods such as side-by-side presentations for showing the results of the data level comparison. In addition, since the comparative information are usually being collected in 3D, other methods such as those presented in [6] may also be used.

4 Bases for Comparing Direct Volume Rendering Algorithms

Because of the varying strategies in which different DVR algorithms generate their images and because we want to utilize intermediate 3D rendering information, it is necessary to find a common bases for comparing DVR algorithms. In particular, in addition to the rigorous specification of key DVR parameters such as viewing parameters, optical models, transfer functions, etc. recommended by Williams and Uselton [4], we also want a systematic way of comparing algorithms as diverse as projection-based, ray-based, texture-based, transform space, etc. Our approach is divided into three steps:

1. Identify a common basis for comparison. In this paper, we use cell projection method[7] as our base algorithm. Specifically, ray-based algorithms are transformed and represented as projection-based algorithms. We see the process as being invertible. That is, if a ray-based algorithm can be represented using a projection-based approach, then a projection-based algorithm can be represented as a ray-based approach. It should be noted that cell projection does not exhaustively represent all existing and future DVR algorithms – for example, it is very difficult to represent Fourier volume rendering [8] using cell projection.
2. Derive comparison metrics from the common base. While mapping other algorithms to the common base is not possible or feasible for all DVR algorithms and requires intimate knowledge of both algorithms, the process allows us to derive data level comparison metrics using the common base. For example, an earlier work [9] that used ray tracing as the base algorithm, we derived a variety of ray-based metrics such as number of samples along the ray, penetration depth of the ray when the accumulated opacity reached a specified threshold, similarity measures of different ray samples, etc.
3. Evaluation. The effectiveness of various bases and metrics can be evaluated as they are used in isolation and in combination.

4.1 Projection-based Methods

One of the major advantages of polygonal projection algorithms is their ability to take advantage of the fast polygon rendering capability in modern workstations. Projection-based methods are well explained in previous works[10, 7]. Here, we give a brief description of the essential steps. Assuming regularly gridded data being projected using parallel projection, Figure 4.1 shows a schematic of how a single volumetric *cell* is projected onto the screen. Each data cell is a cube defined by 8 data points. The 6 faces of the cell are projected onto the screen resulting in up to 7 polygons. Note that other viewing directions would result in a different number and configuration of polygonal projections. In algorithms that use hardware assisted Gouraud shading, colors are computed at all the vertices of the projected polygons by interpolating and integrating cell values and using the appropriate transfer function. Individual polygons are then simply sent through the geometry engine. Rendering an entire volume consists of projecting individual cells onto the screen in back-to-front or front-to-back order [7].

4.2 Simulating Other Methods

In order to simulate other DVR algorithms, we use the standard *scanline* algorithm instead of hardware assisted polygon rendering. Our data structure stores the x, y location in screen space, the z value of each vertex, the distance between the front and back faces at each screen location, and the color, which may have been integrated between two cell faces. In normal scanline algorithms, these fields are usually calculated incrementally using gradients calculated using linear interpolation. However, in order to simulate DVR calculations using ray casting at cell face intersections, it must be noted that data and color values do not change in a simple linear fashion. For example, in Figure 4.1, values vary linearly along back edge A, while values change bilinearly on the frontal occluding polygon. Hence, we need to store bilinear parameters for the front and back of polygon vertices. This allows us to incrementally calculate bilinear parameter values of front and back face points at all pixels of a scanline. Using a software scanline algorithm also allows us to compare variations of color and data interpolation, and different ray sampling patterns. For example, if we use regular ray sampling and trilinear interpolation of colors within a cell, we are simulating *volume texture* technique [11]. On the other hand, if we resample data values at the front and back locations for each point on a scanline and integrate colors computed from the transfer function with the resampled data values, we are simulating ray casting with sampling at the cell faces. Finally, the software scanline implementation also allows us to obtain intermediate rendering information used to derive and visualize comparison metrics, which we describe in the next section.

5 Projection Based Metrics

Here, we present some data level comparison metrics derived from cell projection. Additional projection-based metrics are listed in the ongoing works section and will be included here as results become available.

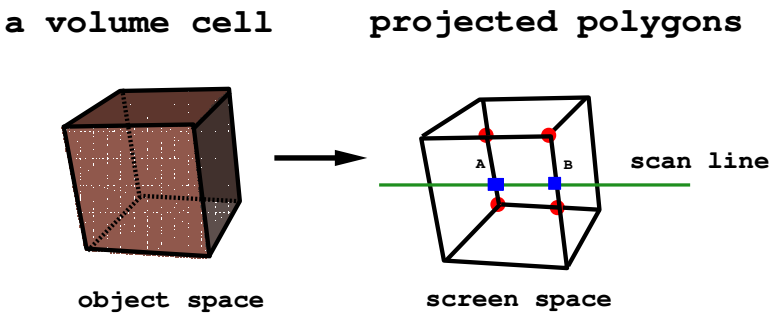

Figure 4.1: Scanline implementation of projection algorithm. The polygonal projection of a cube cell is shown over a scanline. We store the x, y locations and z values of every vertex of the projected polygons. Corresponding information such as colors and data values also stored. Care must be taken when processing the projected polygons. For example, when processing the polygon identified by the four red dots, the blue mark on left edge (A) needs a bilinearly interpolated value from the front face. Similarly, the blue mark on the right edge (B) needs a bilinearly interpolated value from the back face.

The following metrics are measured at each pixel of the DVR image subject to user provided criteria such as accumulated color or opacity intensities values. That is, comparison measurements are collected until color intensities or opacity conditions are satisfied.

1. *Number of Cells.* This metric counts how many cells contribute to each pixel until a user specified condition is satisfied. A higher number indicates that more of the volume data contributed to a pixel, it also indicates that the contributing cells have lower opacity (if that is the user specified condition). In contrast, a lower number indicates that the user specified condition was satisfied relatively soon, or it may also mean that all the volume data have been used up (such as around the periphery of the data volume for non-orthographic views).

2. *Volume Distance.* This metric is similar to the first one, except it measures the amount of contributing cells in terms of Euclidean distance in object space. That is, the distance between the surface of the entire volume and the cell where the given conditions are satisfied is calculated for each pixel. To determine, the depth of the final contributing cell, we use the midpoint between the front and back faces of the projected polygons. Again, this metric provides some indication of the amount of data actually used to generate the DVR image. Coupled with back-to-front or front-to-back traversal order, a "sculpted" view of the data volume can be generated (see surface visualization in Figure 6.2).

3. *Difference Metrics.* Since both metrics above are available from each DVR algorithm, we can also measure and visualize their differences. Note that instead of generating a data level comparison image using one of the first two metrics and then performing a difference image, we first calculate, say, the volume distance of each DVR method, and then take the difference at each pixel.

6 Results

6.1 Illustration of Metrics on Simple Data Set

The data in Figure 6.1 is a simple $3 \times 3 \times 3$ (i.e. 8 cells) hypothetical volume data with zero val-

ues that map to opaque white colors at the two corners. Image 1 is generated using a polygonal projection algorithm while the Image 2 is from a simulation of ray casting with sampling at cell face intersections. Image 1 shows the blurring effects of Gouraud shading of the projected polygons. On the other hand, Image 2 uses trilinearly interpolated data values mapped to colors as defined by the transfer function.

The number of cells metric simply shows how many cells have contributed to the given pixel before reaching a user defined condition. The volume distance metric gives a better indication of the location of cells where the pixel reached the given condition. Note that the differences in both metrics, number of cells and volume distance, appear high around the boundary of the white corner cell (labeled A) and delineates an octant from the entire volume. Note that the difference in octant A is relatively small because both algorithms reached the target opacity after compositing the first cell. On the other hand, the other white corner cell in the back did not contribute to our measurements because all pixels satisfied the given condition before either algorithm reached that corner cell.

6.2 Illustration of Metrics on Scientific Data Sets

We use the 64^3 Hipip data set for Figure 6.2. Both images in the left column are from our simulation of ray casting with cell face intersections. Rendering 1 uses interpolates actual data values, while Rendering 2 interpolates color values. The areas of high difference between the two can be seen on the upper right column (difference between the number of cells from each method). The surface visualization, in VRML format, represents the number of cells for Rendering 1. It provides provides another way of visualizing these metrics where the user can interactively change their viewpoint to gain a better perspective.

7 Web Interface

Our web interface `http://www.cse.ucsc.edu/research/avg/dvr-web.html` features Java applets to provide users the ability to specify viewing parameters, DVR rendering method, as well as select one or more data level comparison metric. The results are returned as either GIF images or VRML files. At present, users cannot upload their own data sets to run against the different DVR methods.

8 Ongoing Work and Conclusions

We presented a framework for comparing different DVR algorithms and illustrated this by mapping different DVR algorithms using cell projection. We also show that the process of simulating an algorithm with another is reversible. Specifically, there is a reciprocity between ray-based and projection-based DVR algorithms.

We then presented two new data level comparison metrics that highlight different aspects of the volume data and the DVR algorithms. These complements the ray-based metrics that we have developed earlier [9]. These metrics, used individually and in combinations, provide additional information beyond how two different DVR images are different – they seek to provide clues as to *why* two DVR images may be different.

We are also investigating several projection-based metrics and tools that will be included in this report as they are completed:

1. Pixel probe. Allows the user to pick a pixel from a DVR image and obtain graphically displayed information regarding the data cells that contributed towards that pixel.
2. Similarity measures. Provides statistical measures such as correlation, standard deviation, etc. of contributing data cells for each pixel in an image pair.
3. Contribution factor. Accounts for the amount of contribution each data cell provides towards the final value of each pixel. For example, each data cell contributes a finite amount towards a pixel reaching an opacity threshold. Likewise, each data cell also contributes a finite amount of color change in a pixel that can be quantified as a separate red, green, blue value or as an aggregate. Users can then specify threshold values of contribution factors to be viewed as iso-surfaces. Such rendering can immediately alert the analyst to parts of their data that are most relevant in a given DVR image.
4. Data probe. Similar to the pixel probe, except the user selects a particular data cell and is shown the contribution made by that cell on the different pixels of the DVR image. Note that this is different than the projection filter.

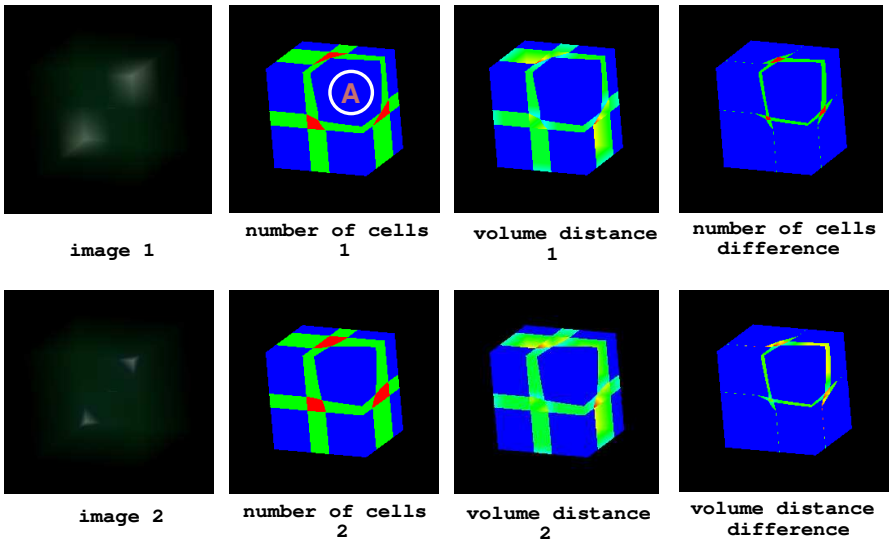

Figure 6.1: Simple 3^3 data volume with values mapped to white at two corners. Image 1 is generated using a polygonal projection algorithm while Image 2 is generated by simulating ray casting with cell face intersections using our scanline algorithm. A standard rainbow colormap is used to map the values from the metrics. Metrics are measured until the opacity has reached 0.06 at each pixel.

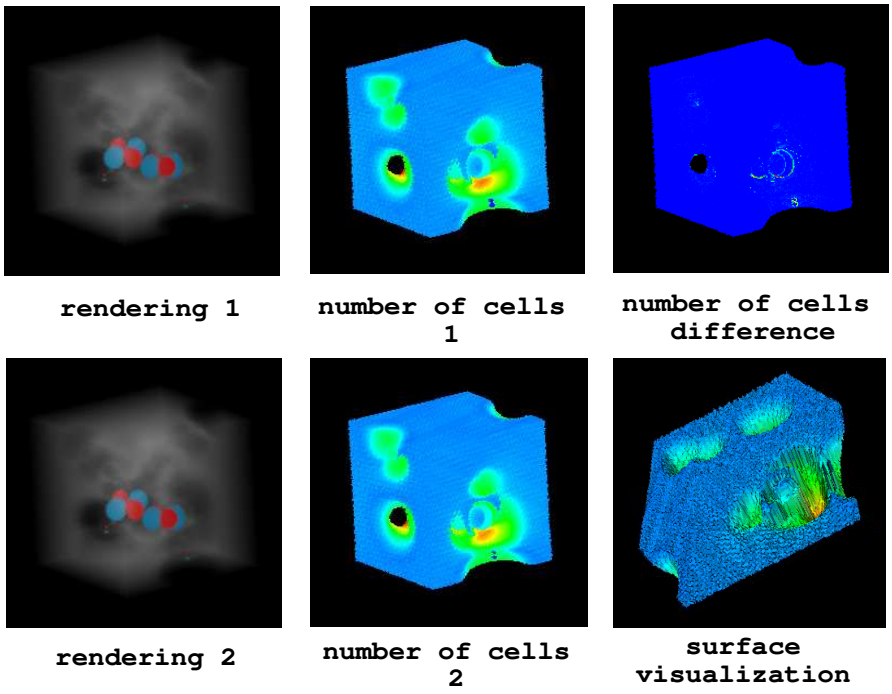

Figure 6.2: Comparisons of the 64^3 Hipip data. Opacity threshold is 0.11 for each pixel. Both methods use our simulation of ray casting with cell face intersection sampling. Rendering 1 uses data interpolation at sample points, while Rendering 2 uses color interpolation.

Finally, our web interface provides a means for the public to gain familiarity with data level com-

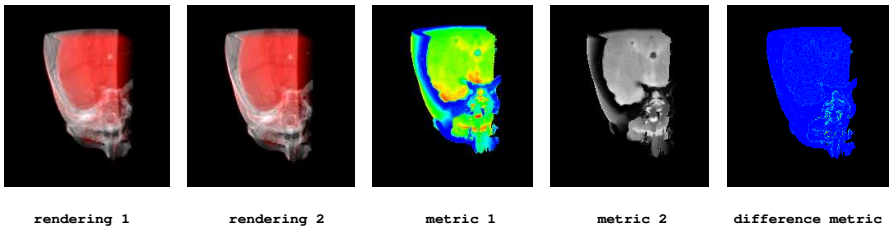

Figure 6.3: Data is a $64 \times 128 \times 56$ sub-volume from a 256^3 CT scan data set. Opacity threshold is 0.2. The data also contains air which is mapped to totally transparent cells. Rendering 1 is generated by polygonal projection while Rendering 2 is generated by our simulation of ray casting. Metric 1 shows the number of cells using the standard rainbow colormap while Metric 2 depicts the volume distance using a linear grey scale colormap. The difference metric of volume distance shows the two algorithms do not exhibit as much difference probably due to the larger volume size (i.e. smaller cell size).

parison of direct volume rendering algorithms through GIF images and VRML files. An obvious extension to this work is to provide these comparative visualization services for user provided data sets.

Acknowledgements

We would like to thank Craig Wittenbrink and Suresh Lodha for collaborative work on uncertainty visualization and Sam Uselton for comments and feedback on image level comparisons. We would also like to thank the Santa Cruz Laboratory for Visualization and Graphics (SLVG) for the wonderful research environment. This projected is partially supported by NSF grant IRI-9423881, NASA Cooperative Agreement NCC2-5207, DARPA grant N66001-97-8900, and ONR grant N00014-92-J-1807.

References

[1] Jim Blinn. Light reflection functions for simulations of clouds and dusty surfaces. In *Computer Graphics*, pages 21–29, July 1982.

[2] J. T. Kajiya and B. Von Herzen. Ray tracing volume densities. In *Proceedings of SIGGRAPH*, pages 165–174, July 1984.

[3] Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(5):29–37, May 1988.

[4] Peter L. Williams and Samuel P. Uselton. Foundations for measuring volume rendering quality. Technical Report NAS-96-021, NASA Numerical Aerospace Simulation, 1996.

[5] Jens Trapp and Hans-Georg Pagendarm. Data level comparative visualization in aircraft design. In *Proceedings of Visualization 96*, pages 393–396. IEEE, 1996.

[6] Alex Pang and Adam Freeman. Methods for comparing 3D surface attributes. In *SPIE Vol. 2656 Visual Data Exploration and Analysis III*, pages 58–64. SPIE, February 1996.

[7] Jane Wilhelms and Allen Van Gelder. A coherent projection approach for direct volume rendering. In *Proceedings of SIGGRAPH 91*, pages 275–284, 1991.

[8] Tom Malzbender. Fourier volume rendering. *ACM Transactions on Graphics*, 12(3):233–250, July 1993.

[9] K. Kim and A. Pang. Ray-based data level comparison of direct volume rendering algorithms. Technical Report UCSC-CRL-97-15, University of California, Santa Cruz, 1997. http://www.cse.ucsc.edu/research/slvg/dvr.html.

[10] P. Shirley and A. Tuchman. A polygonal approximation to direct scalar volume rendering. In *1990 Workshop on Volume Visualization*, pages 63–70, San Diego, CA, December 1990.

[11] Allen Van Gelder and Kwansik Kim. Direct volume rendering with shading via 3D textures. In *ACM/IEEE Symposium on Volume Visualization*, pages 22–30, San Francisco, CA, October 1996.