Ray-based Data Level Comparisons of Direct Volume Rendering Algorithms

Kwansik Kim and Alex Pang

UCSC-CRL-97-15

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

ABSTRACT

This paper describes and demonstrates the effectiveness of several metrics for data level comparison of direct volume rendering (DVR) algorithms. The focus is not on speed ups from approximations or implementations with parallel or specialized hardware, but rather on means for comparing resulting images. However, unlike image level comparisons, where the starting point is 2D images, the main distinction of data level comparison is the use of intermediate 3D information to produce the individual pixel values during the rendering process. In addition to identifying the location and extent of differences in DVR images, these data level comparisons allow us to explain *why* these differences arise from different DVR algorithms. Because of the rich variety of DVR algorithms, finding a common framework for developing data level comparison metrics is one of the main challenges and contribution of this paper. In this paper, we report on how ray tracing can be used as a common framework for comparing a class of DVR algorithms. While this paper focuses on comparing different DVR algorithms, we believe that similar metrics and comparison techniques are also useful for volumetric data comparisons. For example, comparison of experimental versus simulated data sets, or forecasted versus observed data sets, etc.

Key Words and Phrases: Scientific visualization, uncertainty, error, difference, similarity, metrics.

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, 4, 7]. 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.

A sampling of the variations for DVR include schemes like different transfer functions and optical models, ray casting with sampling at cell faces, ray casting with regular sampling, projection splatting, coherent projection, Fourier volume rendering, use of 3D textures, dedicated volume hardware, taking advantage of spatial coherency with octree, binary space partitioning, etc. parallelization with shear-warp, permutation-warp, multi-pass forwards, forwards wavefront, forwards splatting, backwards, etc., extensions to curvilinear grids, and combinations of the above [3, 13, 16, 8, 18, 6, 19, 21, 10, 5, 2, 11, 17].

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. 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 2 DVR images), we want to provide an explanation of the cause for such difference.

The paper is organized into the following sections: a summary of image level comparisons; what we mean by data level comparison; one specific basis for comparing different DVR algorithms; different data level comparison metrics; and several examples illustrating the utility of these metrics.

2 PREVIOUS WORK

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 [20].

3 DATA LEVEL COMPARISON

The name data level comparison was inspired by the work of Trapp and Pagendarm [15] where they used it in CFD applications. Data level methods incorporate intermediate and auxiliary information in the rendering process and use this

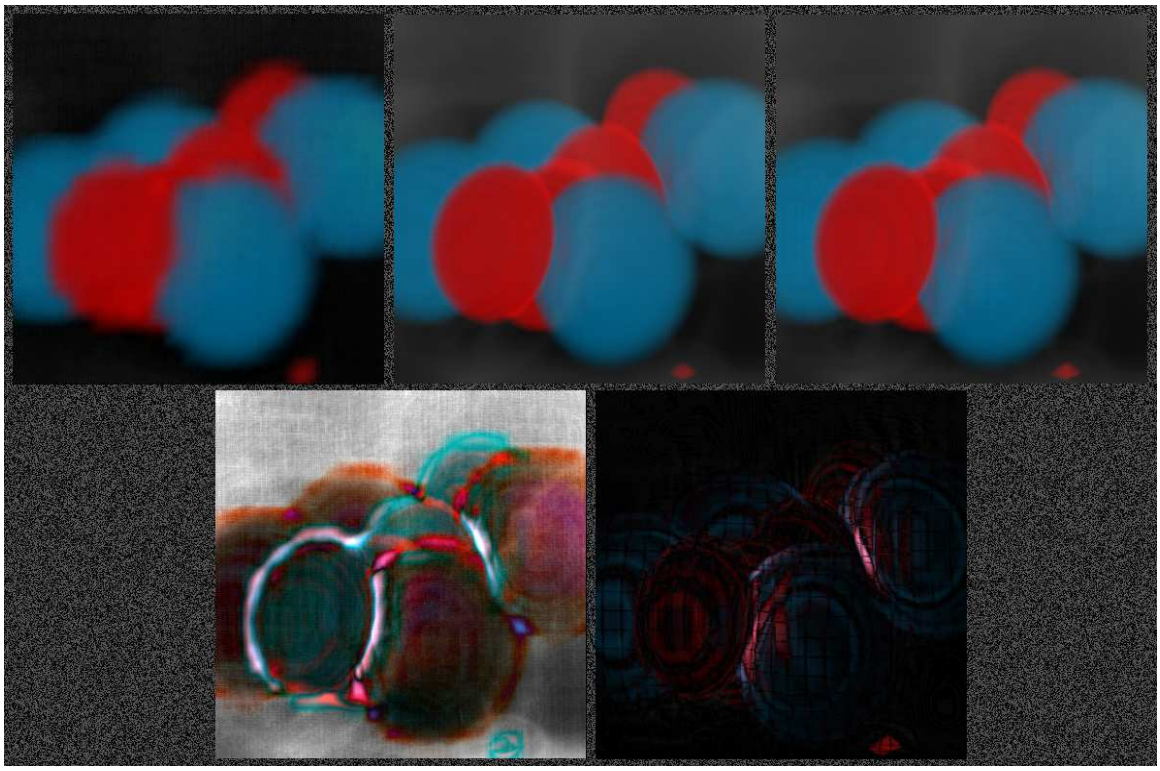


Figure 2.1: Example of image level comparisons side-by-side (top row) and difference images (bottom row). The top row shows results from three different DVR algorithms: projection, ray casting with regular tri-linear sampling, and ray casting with bi-linear sampling at cell faces. The bottom row shows 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.

information to generate a data level comparison image. For example, [12] used surface radiosity values obtained from different form factor calculation methods as intermediate values available for a variety of visual mappings.

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 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.

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 (see Figure 3.1). In addition, since the comparative information are usually being collected in 3D, other methods such as those presented in [12] may also be used.

4 BASES FOR COMPARING DIRECT VOLUME RENDERING ALGORITHMS

Unlike data level comparison of radiosity algorithms [12], there are no geometric primitives upon which to compare DVR data values. Furthermore, because of the view-dependent nature of DVR, and the wide variety of DVR algorithms, it is necessary to first define the basis for comparing these 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 [20], we must first transform DVR algorithms to a common base. For this paper, we use ray tracing as the common base. By this we mean that projection based algorithms are transformed and represented as ray based algorithms. Representing different DVR algorithms using a common base allows us to derive the data level comparison metrics using the common base. It should be noted that ray tracing does not exhaustively represent all existing and future DVR algorithms – for example, it is very difficult to represent Fourier volume rendering using ray tracing. On the other hand, we see the process as being invertible. That is, if a projection based algorithm can be represented using a ray tracing based approach, then a ray based algorithm can be represented as a projection based approach. Therefore, it is possible to use other comparison base aside from ray tracing. Doing so will also result in a different set of data level comparison metrics.

Table 4.1 shows our strategy for mapping a subset of DVR algorithms using ray based approach. Three criteria are used to classify different algorithms. These are: data model – whether data is defined at voxel centers or at vertices, and their associated interpolation or distance functions; value – whether data, color, polygonal approximation values are being interpolated; and sampling strategy – either regularly along the ray or only at ray intersections with cell faces. Based on this three level classification, we can identify several DVR algorithms that can be mapped to ray based approach. This classification is not meant to be exhaustive but rather illustrative of how different DVR algorithms can be viewed in terms of their variants. Used in this manner, Table 4.1 shows that most algorithms can be simulated by ray casting as a reference algorithm with variations in data modeling, interpolation and sampling pattern. Therefore the comparisons of algorithms can be viewed as comparisons between these variations. The same principle can be used for irregular data sets as well as other reference algorithm.

The data model in column (1) comes with either an interpolation function or distance function. Tri-linear interpolation seem to be the most popular interpolation method in most DVR implementations. However, other possibilities include higher order interpolations or adaptive reconstruction [9]. We can also easily incorporate simpler voxel modeling using nearest neighbors or other distance functions into our general ray tracing for comparisons.

Both data values and color values may be used while integrating along the ray. The sampling along the ray can be done at regular fixed step intervals or only at the intersection points between the ray and cell faces. Algorithms like *shear-warp* [5] and *volume texture* techniques[17] use color interpolation at the sample point to improve speed. *Shear-warp* can be considered as a ray tracing based DVR followed by a two dimensional image level warping step.

Our polygonal approximation uses a simple but general polygon intersections of cell faces of regular and irregular volume cells. In general, cell faces project into polygons in screen space. Based on the observation that a ray always passes through two faces of a cell, we simply intersect between all possible pairs of cell faces after we transform all cell vertices into screen space. Note that in degenerate cases, a cell face may project into a line in screen space. Each ray marches through cell by cell along the viewing direction and intersects with one of these non-degenerate projected polygons. The integrated color at the intersection point can be obtained using the following strategy. Since each vertex of the projected polygon is the intersection of two edges of the cell, there is a front and back point corresponding to these two edges. Data at the intersection points along these edges are interpolated,

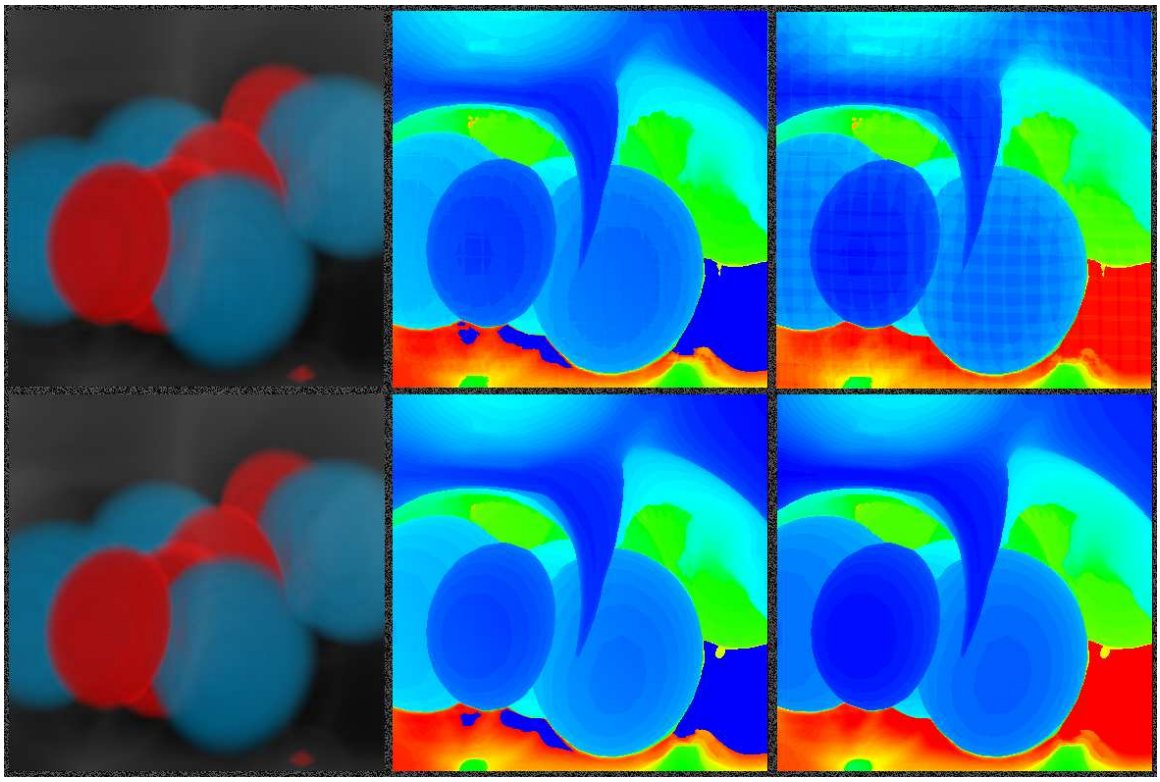


Figure 3.1: Data level comparison of ray tracing with cell face intersection algorithm on the top row and regular ray sampling on the bottom row. Left column shows images from the two algorithms. The middle column shows colormapped images of the distance from the eyes to a point along the ray where a certain opacity threshold (0.11) has been reached. The right column shows the number of samples visited by the rays in each method. Red indicates higher values, while blue shows lower values.

the color at those points evaluated, and then integrated. Gouraud shading is used to obtain the sample color of the point intersected by the ray and the projected polygon. For simplicity, we limited the projection to orthographic since all the cells can be represented by simple translations of a single cell. This allows us to precompute the parameter values in terms of the four vertices of each cell face during projection. Then, during rendering, actual color intensities for both front and back intersections are computed in order to sample a point along the given ray.

Projection algorithms, like coherent projection [14, 19], usually use color interpolation for the points within the projected polygons and data interpolation at the vertices of the polygons. They can be simulated with ray tracing using the polygonal approximations of the cell as described above. We use sampling locations as data to be collected for our metrics. The sampling locations are assumed to be the intersection points with the cell (or at cell faces) and we further assume the sample color is interpolated with the polygon approximations. This is consistent with our other ray sampling algorithms that uses average color of two sample colors and the distance between two sample points in order to calculate the integrated colors for compositing. Therefore, if we have n sample points along the given ray, we have $n-1$ sample colors to be composited in all of our ray casting algorithm variations. This choice of sampling colors is one of the variations in volume renderings. Here, we chose the average color between two samples so as to maintain the consistency of data collected for our metrics.

Splatting [6] uses voxel data modeling and a function that describes the influences of the given voxel in terms of the distance from the voxel location. The ray sampling patterns of splatting algorithms can be considered irregular along the viewing direction since the viewing ray may or

Data Model	Value	Sampling	DVR Algorithm
Cell model with Tri-linear Interpolation	Data	regular	ray casting
		cell face	ray casting
	Color	regular	volume texture
		cell face	shear-warp
	Polygon Approximation	cell face	coherent projection
Voxel model with distance function	Color Distribution	irregular	splatting

Table 4.1: Illustration of how different DVR algorithms can be expressed in terms of ray tracing by changing the data model, value being interpolated, and sampling pattern.

may not pass through the influencing voxels.

Using ray tracing as the common base, *volume texture* or *volume slicing* algorithms can be represented as a ray tracing algorithm that uses color interpolation and regular sampling. Likewise, projection algorithms can be represented by ray tracing algorithms that samples rays at cell face intersections and uses color interpolation within the projected polygons of each cell.

5 DATA LEVEL COMPARISON METRICS

In this section, we present several data level comparison metrics that uses ray tracing as the common base of different DVR algorithms. These metrics reveal information about the volume data as well as the DVR algorithms.

1. Number of samples to reach a certain color (or opacity) value in back to front or front to back direction. Different parts of the data accumulate opacities at different rates. Using this metric, one can gain a better understanding of how many samples along each ray contributed to the final pixel value (see right column of Figure 3.1). Different algorithms use different sampling patterns (e.g. different step size, either regular step size along ray or at cell face intersections only, etc.) and thus a different number of samples along the viewing direction. Artifacts arising from this (e.g. see Figure 2.1) can be explained by using this metric.
2. Distance from the user's eye and distance from the bounding box of the data volume (in viewing direction) to the location in the volume data where the ray reached the given color (or opacity) value in back to front or front to back direction. We refer to these metrics as *eye distance* (see middle column of Figure 3.1) and *volume distance* respectively. These metrics provide the viewer with some idea of how far the ray penetrated the volume independent of how many samples were used along each ray. A possible use of this metric is to determine if rays were able to penetrate deep enough into the volume so that data values of interest (e.g. location of tumors) were able to contribute to the resulting image.
3. Differences of above the metrics between two different algorithms.
4. Another metric is the dot product of the sample vector along the viewing direction from two DVR algorithms. Larger dot products indicate a higher degree of similarity between the two sample vectors. Since different sample values are used for each ray, the sample values are first normalized before the dot product is calculated. This way, dot products are constrained to range from 0 to 1, allowing us to compare neighboring rays and making pseudo-coloring much easier.

   There are several variations possible with dot products because (a) there may be a different number of samples on each ray and (b) the samples may be distributed at different locations along the ray. For example, one may take the dot product of the sample vector independent of the sample location on the ray; only use the first N samples (where N is the smaller of the two); find the ray with the smaller *volume distance* then resample the other ray to this distance before taking the dot product; use the ray with the larger distance and extrapolate the other; etc. The illustration below shows two

more variations. The choice of which variation to use would depend on what we are looking for in the data or between the algorithms.

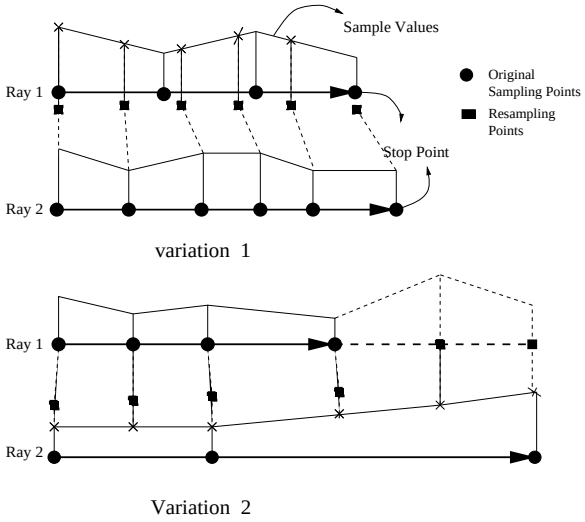


Figure 5.1: Two variations of dot product metric when number of samples and spacing along rays are different. The first variation resamples the ray with more samples to the ray with the smaller volume distance. The second variation extrapolates the ray with the smaller volume distance and resamples the ray with the larger volume distance.

5. Another way of measuring the similarity between two sample vectors is their statistical correlation. This is useful for analyzing the color intensity signals along the sample vector when comparing data versus color interpolation. Let l_1 and l_2 be two normalized sample vectors along a ray from two different algorithms. Then the statistical correlation $corr$ between these two sample vectors is defined by:

$$corr(l_1, l_2) = \frac{cov(l_1, l_2)}{\rho_{l_1} \rho_{l_2}}$$

where $cov(l_1, l_2)$ is the covariance of the two variables and ρ denotes standard deviations.

6 APPLICABILITY OF METRICS

We illustrate the utility of some of the metrics described in the previous section by a number of examples.

6.1 Number of Samples

The cross hatching artifact that is obvious in the difference image between ray casting with regular sampling versus ray casting with cell face intersection (see Figure 2.1) can be explained by the data level comparison on the right column in Figure 3.1. Figure 6.1 shows the same metric applied to the same pair of algorithm above. The data set in this case is a 4x4x4 cube. The image on the left clearly shows the discrete boundaries among the regions with different number of ray samples corresponding to the cell face intersections. This pattern leads to the cross hatching pattern observed in ray casting with cell face intersection.

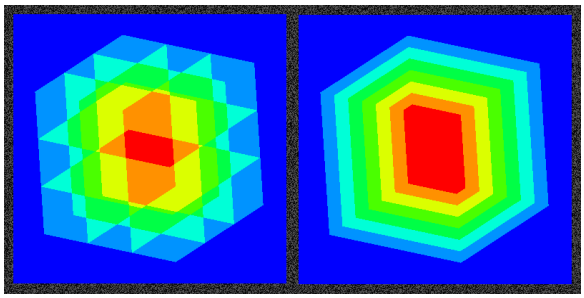
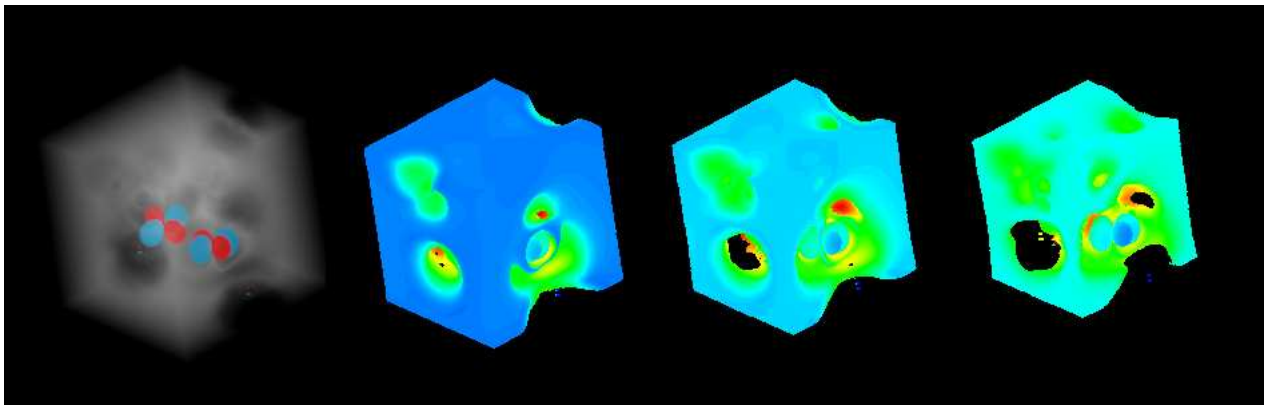


Figure 6.1: Example of data level comparison showing number of samples along each ray. The left image is from a ray tracing algorithm which only visits its cell faces, while the right image is from sampling the ray at regular intervals. Red indicates more samples, while blue shows less samples.

Aside from showing the behavior of an algorithm, the number of samples metric is also a good means of analyzing data content. Figure6.2 shows the colormap visualizations of the number of samples for the accumulated colors in each viewing direction to reach different target opacities. The rendering is done using regular sampling and tri-linear interpolation of data. It shows that our data level visualization provides an added dimension to looking at volume data that is different from viewing in stereo, isosurfaces, rendering from multiple viewing points, or adjusting transfer functions.



(a) image (b) $\alpha = 0.10$ (c) $\alpha = 0.15$ (d) $\alpha = 0.20$

Figure 6.2: Number of samples to reach the given accumulated opacity in volume rendering of Hipip data. (a) volume rendered image using regular sampling, and using the standard "rainbow" colormap, the number of samples when the ray accumulates up to (b) 0.10, (c) 0.15, (d) 0.20 opacity. Black regions indicate that rays entered and exited the volume without reaching the target opacity level.

6.2 Distance Metrics

Distance based metrics such as the eye distance and the volume distance are useful for determining whether the rays have penetrated deep enough into the volume data or not. In Figure 6.3, columns (b) and (c) show the level of penetration using the eye and volume distances respectively. The range of actual values used to colormap the data from these metrics are listed in Table 6.1. Column (b) shows that the eye distance is larger for farther corners and closer for the near corner of the data volume – this is not apparent in the volume rendered image alone. Column (c) shows the distance information measured starting from the entry point into the data volume. Regions that are black indicate that the ray penetrated through the entire volume without reaching the maximum opacity (0.11) for ray termination.

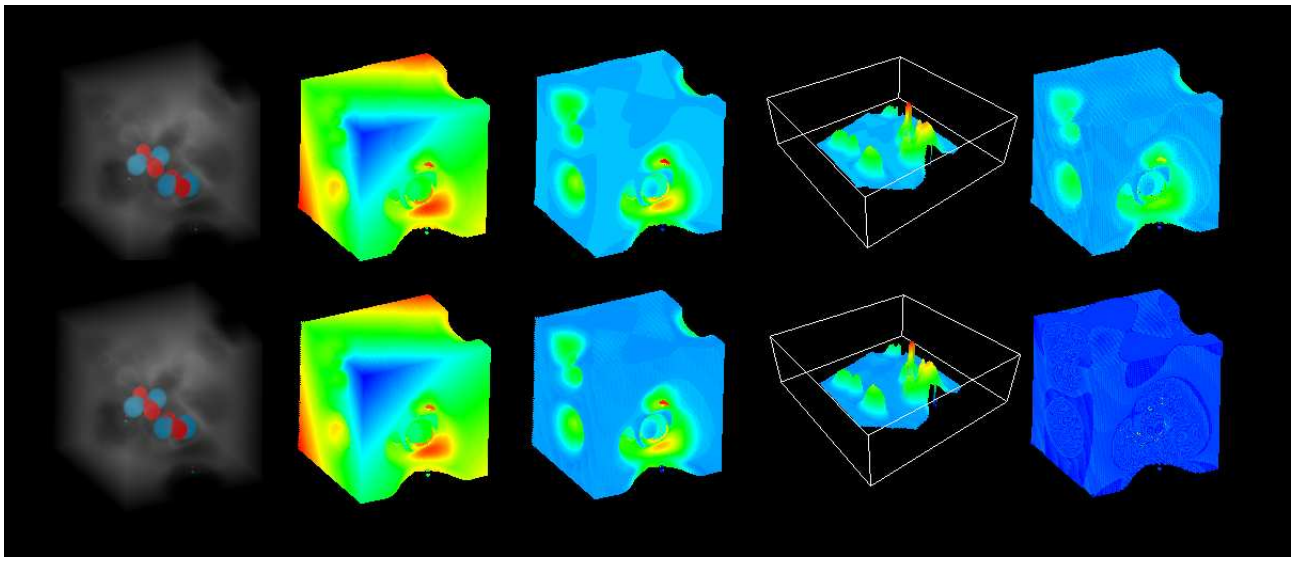
The figures shown so far used the standard "rainbow" colormap. The human's non-linear perceptual response to this colormap can give the false impressions of linear magnitude and undesirable false geometries and banding in the visualizations. Column (d) of Figure 6.3 shows an alternative method of presenting distance metric information by treating it as a height map.

6.3 Dot Product and Correlation

Where independent metrics like number of samples or volume and eye distance fail to explain differences, combined metrics like statistical correlations and the dot product of ray sample vectors provide another perspective into how the data sets and algorithms differ.

Both images in Figure 6.4 are generated by regular ray sampling with the same step size. The only difference is that the top image uses data interpolation while the bottom image uses color interpolation. Note that the black area indicates the ray passed though the entire volume without satisfying the given stop conditions that user sets.

The first column of Figure 6.4 shows that the blue and red blobs near the center of the image is present in the bottom row but not the top row. The images on the number of samples (column (b)) show that both algorithms reached the area of interest. The other four images in Figure 6.4 are from metrics that combine two sets into one. The dot product of the red color components and the correlation of the blue color components, using variation 1 of Figure 5.1, show that there are low correlations around the area in question. However, they also show a spot with higher similarity in the midst of the low correlation area.



(a) images (b) (c) (d) (e)

Figure 6.3: Top row from (a) to (d) contains images and metrics visualizations of ray casting with regular sampling and bottom row from (a) to (d) is for cell face intersection sampling. The distance between two samples in regular sampling method is same as the lateral size of a cell of 64^3 Hipip data. Each column shows (a) volume rendered image, (b) the distance from the eye point, (c) distance from the volume's bounding box in viewing direction when each ray accumulates an opacity of 0.11, and (d) the volume distance rendered as a height field. The top image of column (e) is the image of the difference between the top and bottom of column (b) and bottom image of column (e) is the image of the difference between the top and bottom of column (c).

(a) Algorithm	(b) Number of samples	(c) Eye point distance	(d) Volume distance	(e) top Number of samples difference	(e) bottom Distance difference
regular sampling	min : 6 max : 61	min : 11423.6 max : 11490.6	min : 5.99943 max : 61.0005	min : 1 max : 51	min : 0 max : 3.20
cell face intersection sampling	min : 8 max 93	min 11423 max : 11490.2	min : 5.45037 max : 60.7501		

Table 6.1: Minimum and maximum of metrics colormapped in Figure 6. The eye point distance is calculated in orthographic view and thus the size of numbers are not important. Note that the colormapping from column (b) to (d) is done in terms of each column's minimum and maximum, so comparisons can be made within each column.

That is the colors are similar at that spot but different around the spot. This may be attributed to the diffusing effects of color interpolation and is reinforced by the dot product metric, using variation 2, where most of the image is red (or high) except around the region where the two blobs are missing. The dot product of opacity values using variation 2 is also lower in the area of interest. Note that variation 2 uses extrapolated sample values to make the program collect data up to the same physical location. However, the correlation of data using variation 2 shows very high correlation in most of area including the area of interest. This implies that the data sampled are very similar but the colors are different except at that one spot and thus the source of the differences is most likely to be the difference in interpolation method that computed the color intensities at the sampling points. This is consistent with the fact that the correlation or dot product metrics using variation 1 showed a high spot surrounded by relatively low values.

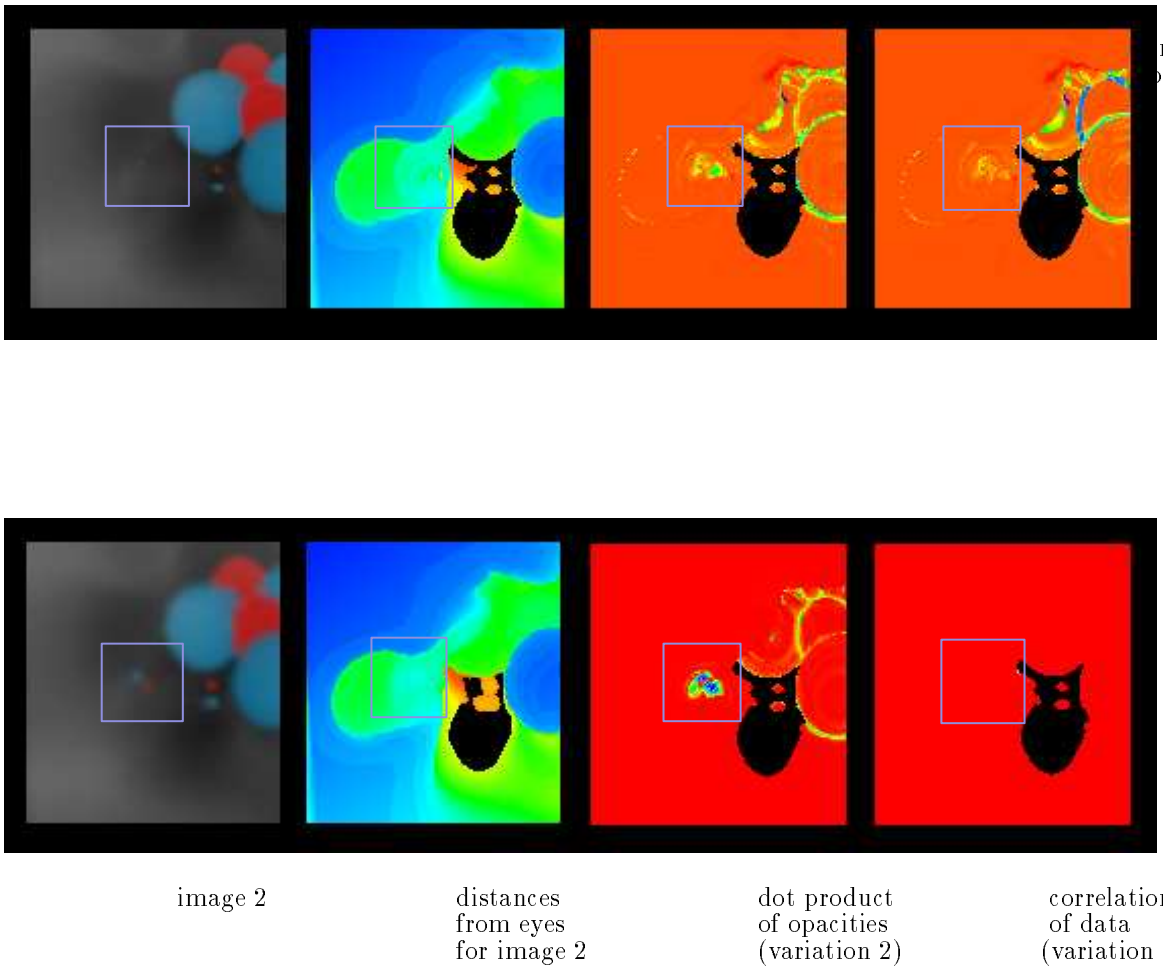


Figure 6.4: Dot products and correlation metrics used to explain the discrepancies between image 1 and 2. The difference arose from the fact that the top row interpolated data values while the bottom row interpolated color values – leading to different sample vectors.

7 CONCLUSIONS

We presented a framework for comparing different DVR algorithms and illustrated this by mapping different DVR algorithms to ray based

approach. We then presented several new data level comparison metrics that highlight different aspects of the volume data and the DVR algorithms. 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* the two images may be different.

We plan to investigate other bases for comparing DVR algorithms aside from ray tracing to test the generality of this comparison framework. Each new basis will require and can provide new metrics for data level comparison which we also plan to evaluate. Finally, we would like to extend these data level comparison methods to compare volumetric data sets such as those arising from experimental versus computational, forecasts versus observed, and nested multi-resolution models. These data level visualizations provide additional probing tools for scientists.

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 supported by NSF grant IRI-9423881, NASA Cooperative Agreement NCC2-5207, 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] Brian Cabral, Nancy Cam, and Jim Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In *Proceedings 1994 Symposium on Volume Visualization*, pages 91–98, Washington, D.C., Oct 1994. IEEE/ACM. Use for a conference paper.

[3] R. A. Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. In *Computer Graphics*, pages 65–74, August 1988.

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

[5] Philippe Lacroute and Marc Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. In *Proceedings of SIGGRAPH 94*, pages 451–458, Orlando, FL, July 1994.

[6] David Laur and Pat Hanrahan. Hierarchical splatting: A progressive refinement algorithm for volume rendering. In *Computer Graphics (ACM Siggraph Proceedings)*, volume 25, pages 285–288, July 1991.

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

[8] Marc Levoy. Efficient ray tracing of volume data. *ACM Transactions on Graphics*, 9(3):245–261, July 1990.

[9] R. Machiraju and R. Yagel. Reconstruction error characterization and control: A sampling theory approach. *IEEE Transactions on Visualization and Computer Graphics*, pages 364–378, December 1996.

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

[11] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995.

[12] 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.

[13] P. Sabella. A rendering algorithm for visualizing 3D scalar fields. In *Computer Graphics*, pages 51–58, August 1988.

[14] 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.

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

[16] Craig Upson and Michael Keeler. V-buffer: Visible volume rendering. In *Computer Graphics*, volume 22, pages 59–64, July 1988.

[17] 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.

[18] L. Westover. Footprint evaluation for volume rendering. In *Computer Graphics*, pages 367–376, August 1990.

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

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

[21] Craig M. Wittenbrink and Arun K. Somani. Permutation warping for data parallel volume rendering. In *Proceedings of the Parallel Rendering Symposium*, pages 57–60, color plate p. 110, San Jose, CA, October 1993.