Direct Volume Rendering of Multi-Valued Data Sets

Newton Der and Alex Pang
Computer Science Department
newtond@cse.ucsc.edu, pang@cse.ucsc.edu

July 2006
Technical Report No. UCSC-SOE-08-20
School of Engineering, University of California, Santa Cruz, CA 95064

Abstract

This paper deals with methods for direct volume rendering of 3D multi-valued scalar data sets. A multi-valued scalar data is one where there are multiple scalar values at each location. Hence, a 3D multi-valued scalar data contains multiple scalar values at each voxel. While there are a few techniques that deal with direct volume rendering of vector data (velocity components at each voxel), diffusion tensor data (eigenvalues at each voxel), multi-field data (multivariate vector at each voxel), or scalar uncertainty data (data plus scalar uncertainty at each voxel), the multi-values are about a single variable. As such, the emphasis of the visualization is not so much about the relationships among the different instances of the same variable, but rather about their collective pattern and behavior. We discuss two approaches for direct volume rendering of multi-valued scalars: (a) convert to scalar then volume render, or (b) volume render then convert to scalar. The latter offers a wide array of interesting possibilities which are described in the paper.

1 Introduction

In direct volume rendering, every voxel contributes to the rendered image. Using a raycasting approach, a ray is shot from the viewpoint through each pixel on the view plane into the 3D volume. Scalars that are encountered along this ray contribute to the final color of the respective pixel. However, what happens when a collection of scalars, rather than a single scalar, is found at each voxel? How do we determine the contribution of each group of scalars to the final pixel color? This paper will discuss two general approaches to extend current direct volume rendering methods to handle this new form of data, which we refer to as multi-valued data.

Visualizing multi-valued data was introduced in [Luo et al. 2003]. In that paper, it was referred to as a distribution data, but has since been renamed multi-valued data since not all multi-valued data are necessarily distribution data. A multi-valued data consist of a collection of values about a *single* variable. This collection can be denoted as $M(v_1, v_2, ..., v_n)$ where each v_i is an instance value of variable V. The collection may arise from a measurement process or a modeled process for example. In the latter case, it is particularly useful to consider probabilistic models where the collection of values describe the set of sample of possible outcomes of the modeled process. This makes multi-values an ideal representation for both data and its associated uncertainty. Where previously uncertainty has often been represented as a scalar quantity, using multi-values to represent uncertainty offers a much richer insight into the nature of uncertainty than could be captured by a single number.

Multi-valued data and multivariate data can both be represented as vectors, but are conceptually distinct. Multivariate data, also referred to as multi-field [Kniss et al. 2003], comprise multiple variables at each spatial location and time whereas multi-valued data describe values found for a single variable. While multivariate visualization techniques may be applied to multi-valued data, they are not necessarily appropriate. For example, a common objective with multivariate data is to find relationships among variables; finding relationships among different instances of the same variable is not usually relevant.

Multi-valued data can be defined for multiple dimensions. Spatial multi-valued data consist of multiple values at each physical location in the domain. One can have time-varying spatial multi-valued data as well. Furthermore, multiple values may exist for more than variable, leading to multidimensional, multivariate, multi-valued data sets. The three data descriptors: multi-valued, multivariate, and multidimensional are orthogonal concepts.

The multi-values at each location and/or each point in time, can be described by its probability density function (pdf). The pdf may be known or unknown, estimated from a sample, or approximated using a discrete function (a histogram) or a continuous function (a continuous pdf). Multi-valued data can also be order-invariant or order-sensitive. For example, multi-valued data from probabilistic models are not ordered. On the other hand, if the values represent a gene expression sequence, then a multi-valued set will only remain meaningful if the order of its components is preserved.

2 Related Work

Several papers deal with the question of how to visualize more than one variable at each 3D location. One of these works extended scalar direct volume rendering to direct volume rendering of vector fields [Fruhauf 1996] by encoding flow directions together with flow magnitudes to both color and opacity. An alternative approach was proposed in [Interrante and Grosch 1998] where 3D LIC is first applied to the volumetric flow data before it is rendered as a scalar texture field. Moving beyond vector fields, [Kindlmann et al. 2000] discussed different methods for directly volume rendering diffusion-weighted magnetic resonance imaging (MRI) tensor data. Among the proposed techniques are: a 2D transfer function defined over a barycentric mapping of tensors with linear anisotropy, planar anisotropy and isotropic regions; Lit-tensors to encode lighting and

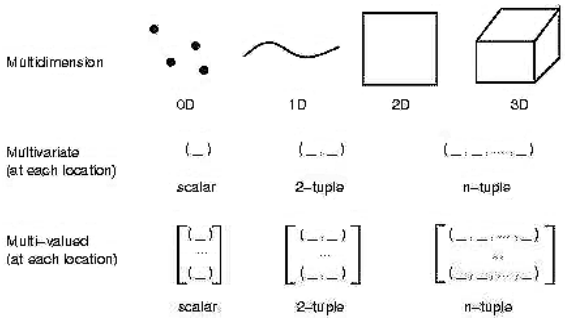


Figure 1: Illustration and comparison between multidimensional, multivariate and multi-valued data. The first row represents spatial domains of dimensionality 0D, 1D, 2D and 3D. For multivariate data, at each location in the spatial domain, there is a multivariate vector shown in the second row. For multi-valued data, (denoted by square brackets in the third row) at each location in the spatial domain, there can be scalar multi-values (1st column) all the way to multivariate multi-values (last column).

shading information to the extracted data; and reaction-diffusion textures as an alternative means to display the anisotropy in the tensors.

A general approach for dealing with multivariate volume rendering is the use of multidimensional transfer functions [Kniss et al.] and their efficient implementation [Kniss et al. 2003]. 2D, 3D or higher dimensional transfer functions can be used to extract features in multi-field data such as CT and MRI data sets. A possible mapping is to map the first dimension to the raw data, the second dimension to gradient magnitude which highlights areas of rapid change in the volume, and the third dimension to the second directional derivative along the gradient direction in order to better isolate material boundaries.

[Djurcilov et al. 2001] proposed both an inline approach using 2D transfer functions, as well as a post-processing pass of compositing separately volume rendered data sets. The former approach used 2D transfer functions that mapped one dimension to raw data and the other dimension to the scalar uncertainty field. The latter approach used directly volume rendered scalar uncertainty and used the result to introduce artifacts into the volume rendered image of the raw data.

In terms of visualizing multi-valued data sets, the previous work in this area are described in [Kao et al. 2001; Kao et al. 2002; Luo et al. 2003]. The results can be summarized into three approaches as follows: (a) Reduce the multi-values at each location into a small set of parametric statistics, then visualize the statistics; (b) Use non-parametric statistical descriptors to describe the shape of the distribution of values in a multi-value, then visualize those descriptors; and (c) Define an algebra that operates directly on multi-valued data types and use it to implement visualization algorithms. These methods were largely applied to 2D distribution data; this paper extends these methods to 3D multi-valued data by volume rendering.

3 Data

One of the data sets used in this paper is a 3D time varying output from ocean modeling. The model covers the Middle Atlantic Bight shelfbreak which is about 100 km wide and extends from Cape Hatteras to Canada. Both measurement data and ocean dynamics were combined to produce a 4D field that contains a time evolution of a 3D volume including variables such as temperature, salinity, and sound speed. To dynamically evolve the physical uncertainty, an Error Subspace Statistical Estimation (ESSE) scheme was employed [Lermusiaux 1999]. This scheme is based on a reduction of the evolving error statistics to their dominant components or subspace. To account for nonlinearities, they are represented by an ensemble of Monte-Carlo forecasts. Hence, numerous 4D forecasts are generated and collected into a 5D field for each physical variable. For each physical variable, the dimension of the data set is 65 x 72 x 42 voxels with multiple values at each point. We look at the sound speed field which has 80 values at each voxel.

The other data set is a 3D diffusion profile data set taken from a sequence of 55 magnetic resonance imaging (MRI) scans of a normal human brain. Diffusion refers to the movement of water molecules within biological tissue. It is an anisotropic activity, meaning the water can travel faster in some directions than others. The diffusion weighting was obtained along 55 different directions uniformly distributed in 3D space, thus creating 55 complete 3D brain volumes. Each volume measures 256 x 256 x 70 voxels, with a scalar at each voxel representing the MRI signal intensity. This signal intensity, which we will denote as S, can be summarized by the following equation:

$$S = S_0 * \exp -b * ADC \tag{1}$$

S_0 is the signal intensity when a diffusion gradient direction is not specified, b is the diffusion weighting factor measured in $\frac{sec}{mm^2}$ (equal to 1000 in this data set), and ADC is the apparent diffusion coefficient in $\frac{mm^2}{sec}$. Solving for ADC in the above equation, we obtain:

$$ADC = \frac{\ln \frac{S}{S_0}}{-b} \tag{2}$$

If we plot the ADC along each of the 55 diffusion weighting directions in 3D, we acqure the ADC profile of the particular voxel. The ADC profile's shape tells us the preferred direction of diffusion at the location in the brain. If the profile resembles a sphere, diffusion is isotropic. If the profile is peanut-shaped, there is one preferred diffusion direction. Knowing these directions is useful as it corresponds to the orientation of white matter tracts in the brain. The profile can also take on a cross shape, meaning there are two intersecting diffusion directions, or two fibers crossing each other. Abnormalities in the white matter can signify the existence of a disease such as Alzheimer's. In this paper, we concentrate our methods on the ADC profile multi-values.

4 Approach

We now discuss two different approaches for direct volume rendering of 3D multi-valued data. The first approach involves converting the multi-values to scalars and then volume rendering the scalar data. The second approach volume renders the multi-valued data directly and then converts the result to scalars.

4.1 Convert to Scalar, then Volume Render

In this section, we discuss several techniques for converting the multi-values to scalar first, before volume rendering. The first method computes parametric statistics on each multi-value which are then volume rendered. The second and third methods take a shape descriptor approach to characterize each multi-value. The fourth method calculates local gradients of multi-values at each voxel to determine opacity. The fifth method uses a 3D scatter plot to map multi-valued voxels to color and opacity.

4.1.1 Computing Aggregate Statistics

One simple way to summarize a multi-value would be to compute a parametric statistic. In order to do this, we assume that each multi-value can be described by a Gaussian distribution. The statistics that we then compute could be, for example, the mean, standard deviation, or variance. Calculating these values provides us with a scalar at each voxel. We can then map these scalars to color and opacity using a simple 1D transfer function. This is illustrated in Figure 2 and Figure 3.

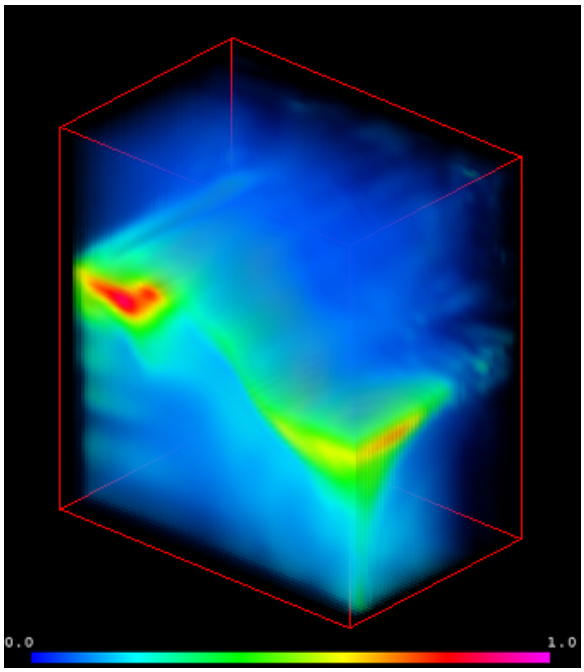


Figure 2: Volume rendered image of ocean data. Normalized standard deviation is linearly mapped to opacity and color. A rainbow color map is used. Red opaque regions have high standard deviation, while blue transparent regions have low standard deviation.

4.1.2 Kullback-Leibler Distance

The assumption that a multi-value can be described by a Gaussian is not always valid. For example, the distribution could be multi-modal. It is also possible to have two multi-values that have completely different distribution shapes, but possess similar means and standard deviations. It would therefore be difficult to discern the difference between these two multi-values when rendering the volume based on these statistics alone. A solution to this problem could be to use a different distance measure that can describe the similarity or difference between two distributions.

One such measure is the Kullback-Leibler (KL) distance [Kullback 1959]. This KL distance is used to describe the similarity (or lack thereof) between a source distribution $P(k)$ and target distribution $Q(k)$. In this paper, $P(k)$ and $Q(k)$ are evaluated at continuous locations from a kernel density estimate of the multi-value. We use the following formula to compute KL distance:

$$KL(P,Q) = \sum_k P(k) \log \frac{P(k)}{Q(k)} \tag{3}$$

If KL distance equals zero, $P(k)$ and $Q(k)$ are identical; as the KL distance increases, the similarity between the two distributions decreases. Note that KL distance can never be a negative value. However, since it is possible that $Q(k) = 0$, $\frac{P(k)}{Q(k)}$ will thus result in ∞. We can solve this issue by padding the appropriate $Q(k)$ with small nonzero values. Distortion of the actual KL distance can occur, but minimizing the padded values can reduce this.

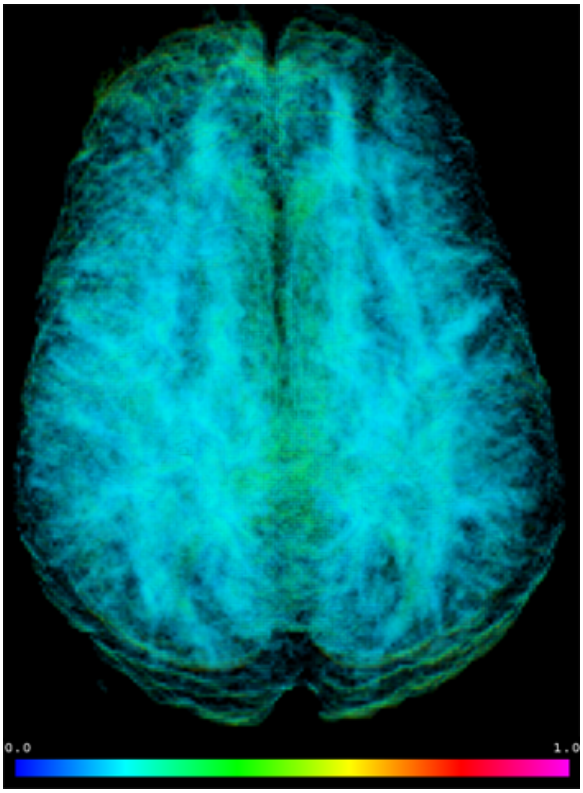
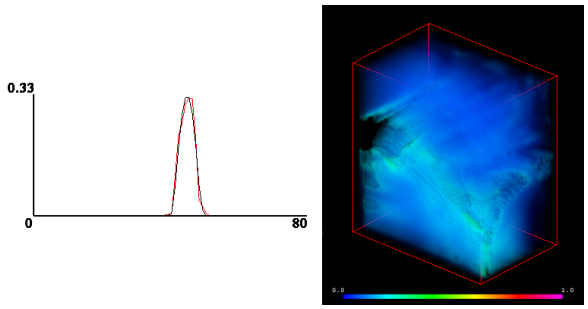


Figure 3: Volume rendered image of brain data. Normalized standard deviation is linearly mapped to opacity and color. A rainbow color map is used. Voxels with normalized standard deviation less than 0.15 were rendered transparent in order to allow a better view of regions higher in standard deviation.

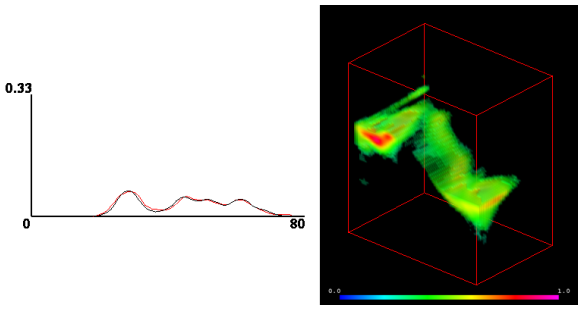
One way to use this distance is for the user to specify a target multi-value of interest. Each of the multi-value in the data can then be compared to the target using KL. This helps us to isolate voxels that share similar properties as the target without having to rely on, but still be able to use, parametric statistics. For example, we can still use standard deviation for assigning color to voxels, but map opacity to increasing similarity. We can also define a cut-off threshold where voxels containing a similarity value below the specified threshold are rendered transparently, while the remaining voxels' opacity is still mapped to a parametric statistic such as standard deviation. Figure 4 shows an example of how this is done.

4.1.3 Peak Hunting

We can further describe the shape and structure of a multi-value by employing a peak hunting algorithm. [Kao et al. 2002] offers a way to classify and count peaks with a user-adjustable threshold. We have used their algorithm in this research, allowing us to determine the following information about each peak: height, width, location and positional order. These values provide us with more information that we can use in building a transfer function. For example, we can isolate multi-values with a peak height above a certain threshold, or highlight multi-values with a certain number of peaks. The

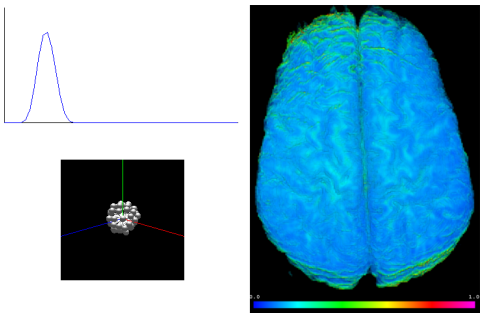


(a) KL distance with unimodal multi-value

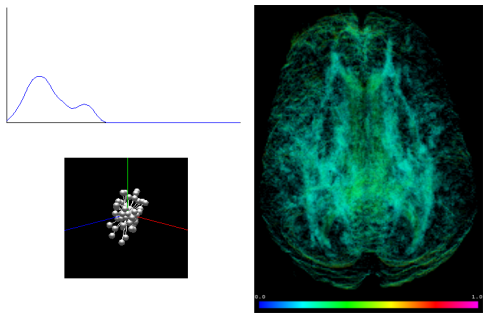


(b) KL distance with multimodal multi-value

Figure 4: Volume rendering using a similarity measure to filter out unwanted voxels. In the plots on the left, the red curves are the user-specified target multi-value, while the black curves are the density estimate of the multi-value at some voxel in the volume. KL distance is linearly mapped to opacity, where voxels falling below a user-specified threshold are rendered transparently. In (a), we wish to extract regions with unimodal distributions, while in (b), we wish to extract regions that are multimodal. Comparing these two images with Figure2, we now have the ability to extract only those portions of the volume that are more interesting.



(a) KL distance with isotropic ADC profile



(b) KL distance with anisotropic ADC profile

Figure 5: Volume rendering using a similarity measure to filter out unwanted voxels. In the plots on the top left, the curves are the density estimate of the ADC profile's magnitudes. In the plots on the bottom left, the ADC profile is displayed. In (a), we wish to extract regions that have isotropic ADC profiles, while in (b), we wish to extract regions that are anisotropic. Comparing these two images with Figure3, we now have the ability to extract only those portions of the volume that are more interesting.

ability to combine these characteristics with other associated scalar values when assigning color and opacity helps to make interesting regions of the volume more apparent. Figure 6 shows a rendering based on the number of peaks at each multi-value.

4.1.4 3D Transfer Functions

Once we have computed the various statistics and measures explained above, we can apply a transfer function to assign color and opacity to each voxel. In Figure 2 we used a basic 1D transfer function, giving increasing opacity to higher values, and applying a rainbow color map. The downside to this approach is that only one scalar is used in the transfer function, which does not exploit the extra information provided by other measures. Here, we experimented with 3D transfer functions similar to [Djurcilov et al. 2001; Kniss et al. 2003].

Instead of a 2D scatter plot, we use a 3D scatter plot with each axis representing a different scalar value. For example, in Figure 7, the X-axis represents modality, the Y-axis denotes standard deviation, and the Z-axis is arithmetic mean. We can interactively highlight different regions of the scatter plot and assign any color and opacity we desire. This method helps us to isolate voxels with constraints on three parameters.

4.2 Volume Render, then Convert to Scalar

In this section, we introduce a method of volume rendering the multi-value data first, and then converting to scalar for assigning color and opacity. One such approach involves performing a weighted binwise addition of the multi-values intersected along each ray cast into the 3D volume (see Figure 8). We then discuss three possible weights that can be used to isolate certain features in the data.

In traditional raycasting, a ray is "shot" into 3D space for each pixel. Voxels encountered along each ray contribute to the final pixel color. Each voxel is usually associated with a scalar, but in our research, a multi-value exists at each location. Previously, we first reduced each multi-value to a scalar, and then used these scalars to composite the final image. The result is that we are losing much of the information about each multi-value when we are compositing just a derived scalar. While similarity measures such as the Kullback-Leibler distance can tell us certain properties about the shape of a multi-value, the ideal situation would be to composite the actual multi-values so that their shape and structure contribute to the final pixel color.

The approach that we propose is to compute a weighted binwise addition of the density estimates of the multi-values met at each pixel. We now describe the combination process, followed by explanations of possible weights that would be useful in isolating features in the volume.

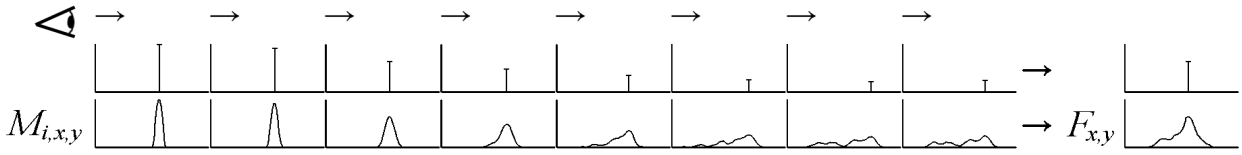


Figure 8: Illustration of multi-values encountered by a view ray. The top row represents a traditional scalar volume rendering. The bottom row shows what happens when we wish to volume render multi-valued data. For a ray cast into the volume at pixel coordinates (x, y), we encounter n multi-values. The multi-values $M_{i,x,y}$ are summed binwise, with a weight w_i corresponding to each multi-value. The result is a final multi-value $F_{x,y}$. Volume rendering the actual multi-values, rather than converting the multi-values to scalars first, helps retain more information about each multi-value's structure.

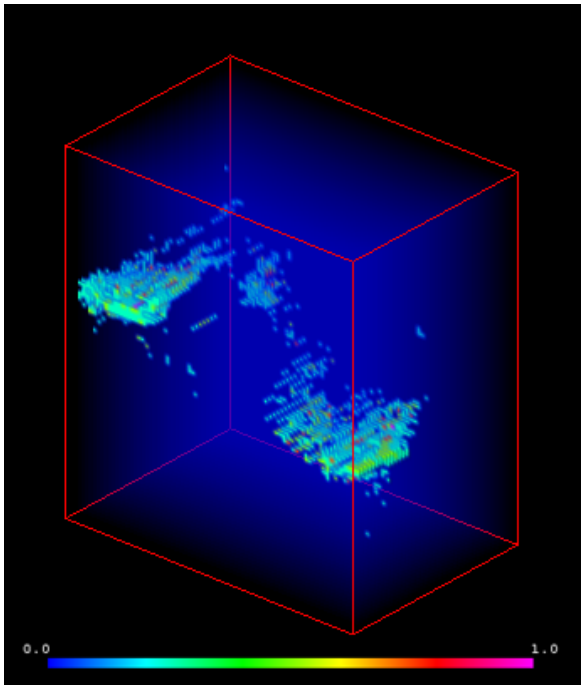


Figure 6: Volume rendering using a transfer function based on modality (number of peaks) of each distribution. Unimodal multi-values are characterized by the dark blue area. They were given lower opacity to allow a clearer view of the multimodal multi-values in the central portion of the volume.

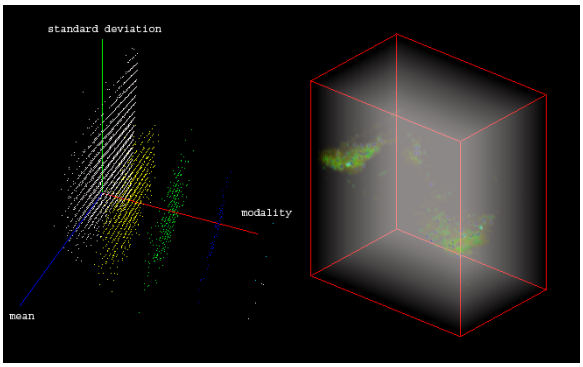


Figure 7: Volume rendering using a 3D scatter plot transfer function. The image on the left shows the 3D scatter plot with the red axis corresponding to the modality of each multi-value, the green axis denoting standard deviation, and the blue axis representing arithmetic mean. A lower opacity was given to the more abundant white region which corresponds to voxels with a lower modality. The green regions denote areas with higher modality.

4.2.1 Technique

It is important to note that when performing the binwise addition, we use a density estimate of the multi-value instead of the actual multi-value. Before we explain our combination algorithm, we first provide the following definitions:

1. Let (x, y) be the current pixel coordinates of the ray we are casting.

2. Let n be the number of voxels (and thus, the number of multi-values) encountered along the ray.

3. Let $M_{i,x,y}(v_{i,1}, v_{i,2}, v_{i,3}, ..., v_{i,k})$ be the i-th multi-value found at pixel coordinate (x, y). Let k be the number of samples in the density estimate, and let $v_{i,j}$ be the j-th sample associated with the i-th multi-value.

4. Let w_i be the weight associated with the i-th multi-value.

5. Let $F_{x,y}(r_1, r_2, r_3, ..., r_k)$ be the final combined multi-value at pixel coordinate (x, y). Let k be the number of samples in the multi-value, and let r_j be the j-th sample.

Our basic equation for the weighted binwise addition is then:

$$F_{x,y} = \sum_{i}^{n} w_i M_{i,x,y} \tag{4}$$

where for each r_j, $j = 1...k$

$$r_j = \sum_{i}^{n} w_i v_{i,j} \tag{5}$$

The end result is a 2D array of multi-values that we can then compute various metrics of, and apply a transfer function based on those computed values. The weight w_i can be a precalculated scalar associated with the multi-value, such as the standard deviation or modality mentioned in previous sections. We now look at different possible measures that we can use to weight the multi-values along each ray.

4.2.2 Standard Deviation

Standard deviation is a statistic used to measure the dispersion or variation in a distribution. In our ocean data, the highest fluctuation in sound speed values occurred along the shelf break, which is situated along the curved region running across the center of the image

in Figure 9. The surrounding water showed relatively consistent sound speeds by comparison.

Naturally, assigning greater weight to the multi-values with higher standard deviations gave greater prominence to voxels along the shelf break, denoted by the red-pinkish area. The resulting 2D array of multi-values can then be assigned color and opacity according to standard deviation as in Figure 9.

4.2.3 Similarity

Weighting by similarity to a target multi-value is an efficient means of isolating only certain multi-values of a desired shape. This method gives us even greater flexibility because instead of just weighting by a standard deviation or the number of peaks, we can specify an exact multi-value to compare against. For the similarity measure, we use the KL distance described earlier.

Figure 10 shows sample renderings of the ocean data when we compare each multi-value to a unimodal multi-value and use the corresponding similarity value as the weight. We see the advantage of our multi-value combination method, since we see a separation among unimodal multi-values that was not evident before. The red-pinkish region at the top of the volume represents a unimodal area with a high standard deviation. The yellow-greenish area at the bottom denotes similarity to a unimodal multi-value, but with a lower standard deviation. We also see a faint, almost transparent blue curved region running through the center of the volume, which is the shelf-break. This means the shelf-break differs in shape from a unimodal multi-value.

4.2.4 Peak Information

The modality, or number of peaks in a distribution, can also be used to weight the multi-values. A complication that we encountered was that the more interesting but rare multimodal multi-values were overwhelmed by an abundance of unimodal multi-values. During the computation of the final multi-value $F_{x,y}$, the contribution of unimodal voxels outweighed the multimodal, resulting in a unimodal $F_{x,y}$. To combat situations like this, it is necessary to scale down the weight of the overwhelming multi-values.

In this case, we scaled down the weight of unimodal multi-values by a factor of 100. The result can be seen in Figure 9. Not surprisingly, the multimodal multi-values are populated in the shelf break region, where standard deviation is higher.

As mentioned before, the peak hunting algorithm that we implemented also gives us additional information other than the number of peaks. For example, we can discern the location of the peak's apex within the multi-value. The location of the peak's apex denotes the most frequently occurring value in the multi-value. This criterion can also be used to weight the multi-values during the combination process. However, in the ocean data, the apex locations among unimodal multi-values are so consistent that practically the same image is rendered when weighting by modality. The same situation applies to other peak measures such as height and width.

4.2.5 Gradient

A popular technique in volume rendering is to compute the gradient at each voxel. The gradient is useful because it provides insight into regions with the highest change. Calculation of the gradient in the scalar case is straightforward. With multi-values, gradient calculations can be used to tell us where there are significant changes in the shape of the distributions.

For interior voxels, central differencing can be used to calculate gradients; while forward differencing cab be used for boundary voxels. (see Figure 11).

To calculate gradients of voxels with multi-values, we can use the KL distance mentioned earlier. Since t, b, l, r, n, and f are now

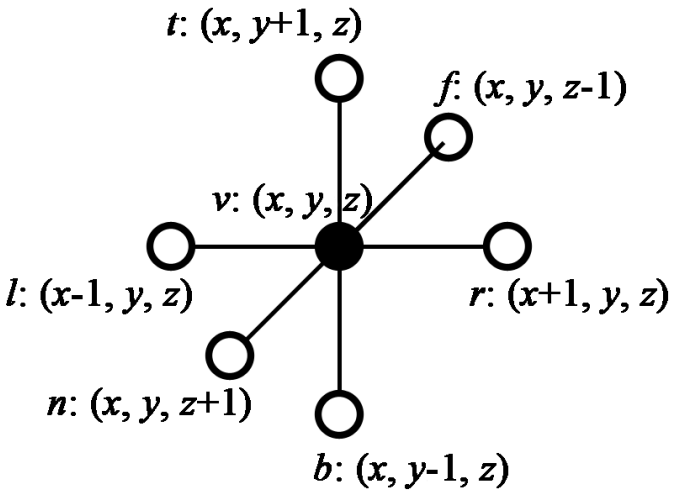


Figure 11: Illustration of the relative locations of the voxels for computing the gradient. Voxel v is the location we want to find the gradient of, and t, b, l, r, n, and f are its neighboring voxels.

multi-values instead of scalars, we calculate KL distance instead of differences. So, for interior voxels, Δx, Δy, and Δz become:

$$\Delta x = \frac{KL(l,r)}{2\Delta}$$

$$\Delta y = \frac{KL(t,b)}{2\Delta}$$

$$\Delta z = \frac{KL(n,f)}{2\Delta}$$

While for boundary voxels, we have:

$$\Delta x = \frac{KL(r,v)}{\Delta} + \frac{KL(l,v)}{\Delta}$$

$$\Delta y = \frac{KL(t,v)}{\Delta} + \frac{KL(b,v)}{\Delta}$$

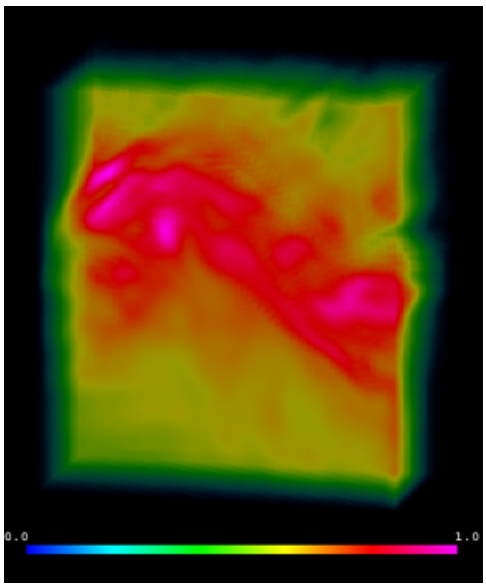
$$\Delta z = \frac{KL(n,v)}{\Delta} + \frac{KL(f,v)}{\Delta}$$

The gradient is then just the sum of δx, δy, and δz: We then use the gradient as a weight when computing our binwise summation of the multi-values.

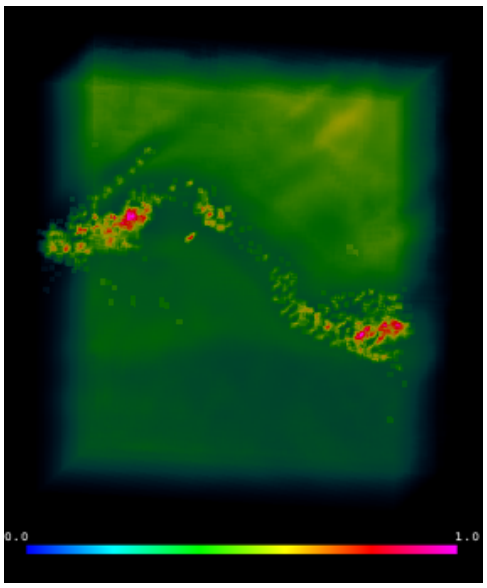
$$\nabla v = \Delta x + \Delta y + \Delta z \tag{6}$$

4.2.6 Area

One metric that describes the shape of a multi-value, like similarity and modality, is the area under the distribution representing the multi-value. For probability distributions, the area would be one. But for the combined, weighted multi-values along the view ray, the area can change. The area can be computed in many different ways using a number of approximation methods. We used a simple trapezoidal approximation for various implementation reasons. Figure 13 shows two volumes weighted by area.

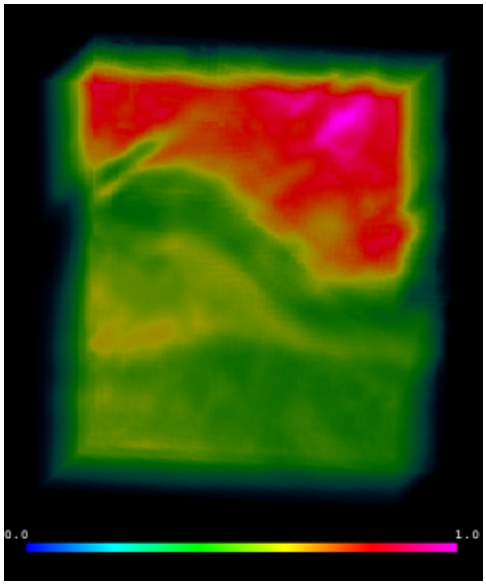


(a) Standard deviation-weighted volume

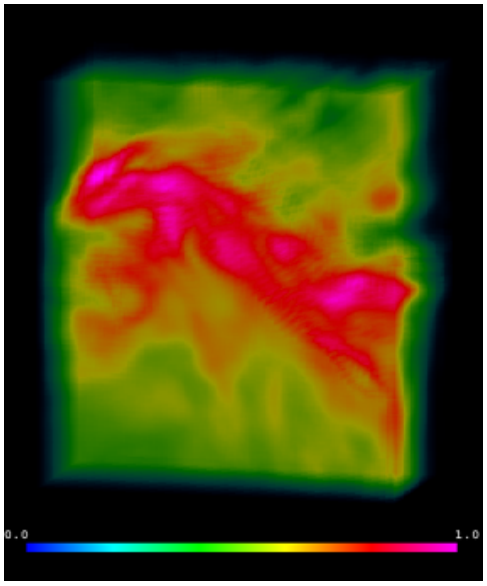


(b) Modality-weighted volume

Figure 9: Raycast renderings of a standard deviation-weighted volume on the left, and a modality-weighted volume on the right. Both are assigned color and opacity according to a linear map of standard deviation, illustrated at the bottom of each image. In (a), the shelf break is denoted by the reddish area, having high standard deviation. We also see the shelf break in (b), where the red and yellow regions have higher modality (number of peaks), and higher standard deviation as well.

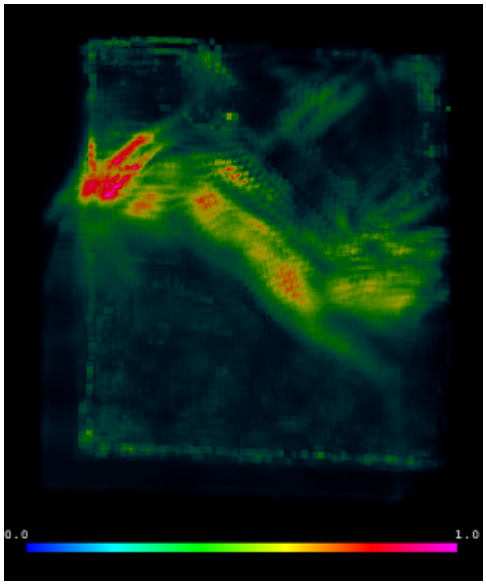


(a) Unimodal similarity-weighted volume

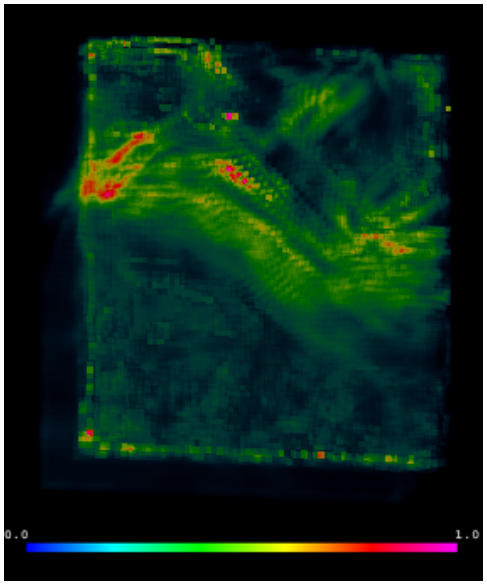


(b) Multimodal similarity-weighted volume

Figure 10: Raycast renderings of similarity-weighted volumes. The volume on the left was weighted by similarity to a unimodal multi-value, while the volume on the right was weighted by similarity to a multimodal multi-value. Both are assigned color and opacity according to a linear map of standard deviation, illustrated at the bottom of each image. In (a), we can see a transparent curve along the center, where the shelf-break is located. This means the shelf-break is least similar to a unimodal multi-value, while the surrounding areas are most similar. We note that the upper half has much higher standard deviation than the lower half, which is not evident in the other volume rendering methods we have discussed. In (b), we see a red curved region running through the center, meaning the shelf break is similar to a multimodal multi-value, and has high standard deviation. The image in (b) looks fairly close to the standard deviation-weighted volume, but provides sharper definition to the shelf break.

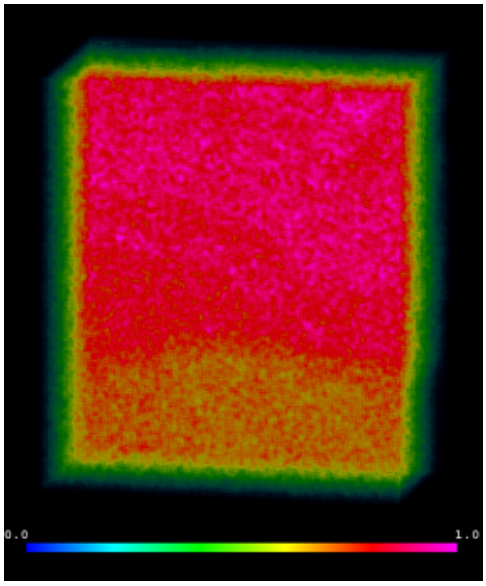


(a) Gradient-weighted volume #1

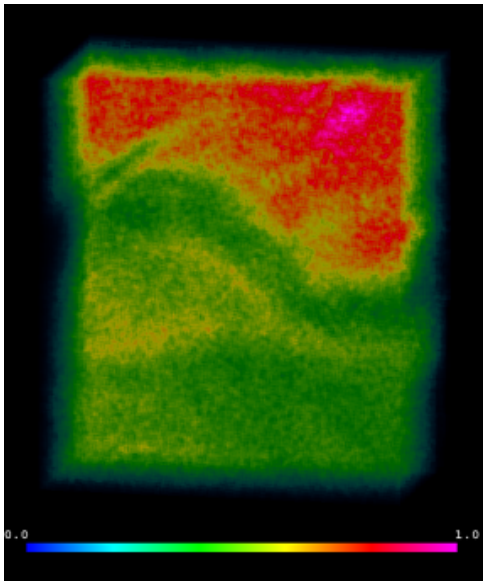


(b) Gradient-weighted volume #2

Figure 12: Raycast renderings of gradient-weighted volumes. The volume on the left was assigned color and opacity according to a linear map of area. The volume on the right was assigned color and opacity according to a linear map of standard deviation. In (a), we see better isolation of the shelf break from the rest of the volume. The red and yellow areas denote regions of rapid change and increased area (under the multi-value curve). In (b), we see that some areas along the shelf break, although having high area, have lower standard deviation.



(a) Area-weighted volume #1



(b) Area-weighted volume #2

Figure 13: Raycast renderings of area-weighted volumes. The volume on the left was assigned color and opacity according to a linear map of area. The volume on the right was assigned color and opacity according to a linear map of standard deviation. In (a), we see that the areas are high in the top portion, with a smaller region below with low area. In (b), we see better separation along the shelf break. The top half has high area and high standard deviation, while the lower half has high area but lower standard deviation.

5 Results

In Figures 2 and 3 we computed the standard deviation on each multi-value. In Figure 6 we found the number of peaks in each multi-value's density estimate. All three images are examples of extracting one scalar per multi-value, and then using those scalars to determine color and opacity. For the ocean dataset, these methods helped to define the curved shelf break in the center region of the volume. For the brain dataset, computing the standard deviation of each multi-value proved useful in highlighting anisotropic regions. Isotropic ADC profiles have a spherical shape, meaning all 55 directions share a consistently similar magnitude, and thus have low standard deviation. Since anisotropic ADC profiles have a mix of high and low magnitudes to comprise a peanut shape, standard deviation is higher. For these methods, only one parameter was used in each case.

In Figures 4 and 5 we combined another variable into the rendering. We computed the similarity between each multi-value and a target multi-value. Multi-values whose similarity fell below a certain threshold, i.e. not similar enough, were made transparent. In Figure 4, the image on the left consists of multi-values which are most similar to a unimodal multi-value. Note the absence of the center region. We see this appear in the right image, where we isolated multimodal multi-values. This suggests that multi-values along the shelf break are mostly multimodal, while the rest are unimodal. We can also infer that the unimodal multi-values have fairly consistent standard deviations, as all of them share the same dark blue color. The multimodal multi-values, on the other hand, show a wider distribution of standard deviation values, denoted by the red, yellow, and green areas.

In Figure 5, we wished to isolate regions of isotropy and anisotropy. Through probing various points in the volume, we discovered that isotropic regions were denoted by a unimodal multi-value whose peak was relatively high. The image on the left shows the multi-value, the corresponding spherical ADC profile below, and the resulting volume with isotropic regions given high opacity, while anisotropic voxels are made transparent. The signature of anisotropic regions, on the other hand, was illustrated by a bimodal multi-value. The taller peak represents the higher frequency of smaller magnitudes (located near the center of an anisotropic voxel's peanut shaped profile), while the smaller peak represents the lower frequency of higher magnitudes (the elongated ends of the peanut). While we were able to see a basic structure of the white matter tracts in Figure 3, using a similarity measure provides us with a more accurate and defined image. We were unable to successfully separate voxels with two fibers intersecting from voxels with a single fiber running through it. This is because the density estimates for these two cases were too similar to each other, and KL distance was not sensitive enough to pick up on their slight differences.

In Figure 7, we illustrate a way to directly assign color and opacity to the computed scalars using a 3D scatter plot. The user can draw a 3D box around a group of data points and assign any color or opacity to only those points. The red axis corresponds to modality, the green axis denotes standard deviation, and the blue axis represents arithmetic mean. The greenish areas in the rendered volume show the shelf break having higher modality and standard deviation, while the mean is fairly consistent regardless of spatial location.

In Figures 9, 10, 12, and 13, we provide example renderings from our multi-value combination method. In Figure 9, we see standard deviation-weighted and modality-weighted volumes, both with color and opacity linearly increasing with standard deviation. The shelf break predictably has higher standard deviation, illustrated by the red curved region. The modality-weighted volume also isolates the shelf break multi-values and tells us that they are generally multimodal.

Figure 10a shows an image produced from a volume weighted by similarity to a multimodal multi-value, with color and opacity determined by standard deviation. It appears similar to the standard deviation-weighted volume in Figure 9a, but provides better definition of the reddish shelf break. The volume in Figure 10b is weighted by similarity to a unimodal multi-value, with color and opacity also increasing linearly with standard deviation. The shelf break is of course more transparent than the rest of the volume, but what is most interesting is that the top half has much higher standard deviation than the lower half. This separation among unimodal multi-values was not evident before in Figure 2 or 4, where we converted the multi-value data to scalar prior to volume rendering. We now see the advantage of our multi-value combination method, as we can render the volume of multi-values before converting the data to a scalar form. Our technique enables each multi-value itself, rather than a scalar representative, to contribute to the final rendered image. Moreover, we can combine different weights and transfer functions to render our volume with a rich amount of information.

Figure 12 shows two volumes weighted by gradient magnitude. The shelf break shows a region of higher change among multi-values. The color and opacity in the left image is determined by the area under each multi-value density estimate, while the right image is colored according to standard deviation. We see that multi-values along the shelf break have high area denoted by the red-yellowish blobs in the left volume, but not necessarily as high standard deviation, since those same areas are more green in the right image.

Figure 13 shows two volumes weighted by area, with the left one colored by area, and the right one colored by standard deviation. We see that although the area is fairly high for a majority of the volume, standard deviation is high mostly for the top half above the shelf break. Note the grainy texture of both volumes, and note the similar appearance of Figure 13b to Figure 10b. This suggests that although unimodal multi-values have smooth transitions among standard deviation values, the area varies much more from voxel to voxel. We can also conclude that the upper half above the shelf break is a region of high area and standard deviation, but exhibit a unimodal shape. The bottom half is unimodal as well, but has lower standard deviation and area. This example again illustrates the power of rendering a volume using our weighted binwise addition of multi-values, prior to converting the multi-values to scalars for assigning color and opacity. We are able to combine various weights and measures to draw the most conclusions about the data.

6 Implementation and Performance

The images in Figures 2 to 7 were rendered using a 2D texture mapping approach. Texture stacks are axis-aligned, and so the rendered stack depends on the current viewpoint. Alpha blending is used to composite the textures. Because we are using equipment with hardware-accelerated texture mapping, the volume is fully interactive in that it can be rotated and zoomed with no perceived delay. Pre-processing was done to reduce the multi-values first into scalar statistical measures; this step took less than 5 seconds.

Renderings involving the combination of different multi-values were drawn using a software-based raycasting approach. As a result, performance drops significantly, with render times dependent on viewpoint and the number of voxels intersected along each ray. Average render times hovered around five minutes on a Pentium 4 3.0 Ghz machine with an NVidia GeForce4 MX440 graphics card. With the recent attainment of newer graphics hardware, we now have the ability to considerably reduce rendering times by using hardware-supported 3D texture maps and fragment shaders. Unfortunately time did not allow for this newer implementation, but we hope to achieve this in a future revision of this paper.

7 Conclusions and Future Work

We have presented several methods of direct volume rendering 3D multi-valued data. First, we used basic parametric statistics such as arithmetic mean and standard deviation to summarize each multi-value. Since these measures cannot adequately distinguish two differently shaped multi-values from each other, we employed a shape descriptor approach. Kullback-Leibler distance helped to define the similarity between the volume's multi-values and a target multi-value, and a peak hunting algorithm gave us information on the multi-value's modality.

We also explored a way to delay the conversion from multi-value to scalar as late as possible in the volume rendering pipeline, so that each multi-value – rather than a scalar associated with it – could contribute to the final image. The idea involved performing a weighted binwise addition, and results helped to highlight the interesting features of the volume.

In some of the renderings, a voxel's spatial location relative to other voxels is difficult to discern due to the nature of direct volume rendering. In the future we plan to implement shading techniques to enhance the visual structure of the volume. We plan to improve on our current binwise addition method by applying the transfer function after each ray-cast step, rather than applying it after the entire ray-casting process. This should alleviate the problem of unimodal multi-values polluting the final image, and reduce the chance of inconsistent results. The recent acquiring of a newer graphics board will also allow us to use 3D textures to manage the brain data more easily, since previous memory limitations have prevented us from performing the binwise addition process on the data. More importantly, now that we have methods for volume rendering multi-value data, we will share the results and obtain feedback on their utility and improvements from our science colleagues.

8 Acknowledgements

We would like to thank Pierre Lermusiaux for the ocean data set, and Alison Love for help in getting the project going. This work is supported in part by NSF grant ACI-9908881 and NASA grant NCC2-5281.

References

DJURCILOV, S., KIM, K., LERMUSIAUX, P., AND PANG, A. 2001. Volume rendering data with uncertainty information. In *Data Visualization 2001*, Springer, D. Ebert, J. M. Favre, and R. Peikert, Eds., 243–252, 355–356. `www.cse.ucsc.edu/research/avis/uvolren.html`.

FRUHAUF, T. 1996. Raycasting vector fields. In *Proceedings Visualization'96*, 115–120, 475.

INTERRANTE, V., AND GROSCH, C. 1998. Visualizing 3d flow. *IEEE Computer Graphics and Applications 18*, 4 (July-August), 49–53.

KAO, D., DUNGAN, J., AND PANG, A. 2001. Visualizing 2D probability distributions from EOS satellite image-derived data sets : A case study. In *Proceedings Visualization 2001*, IEEE, 457–60.

KAO, D., LUO, A., DUNGAN, J., AND PANG, A. 2002. Visualizing spatially varying distribution data. In *Proceedings of the 6th International Conference on Information Visualization '02*, IEEE Computer Society, 219–225.

KINDLMANN, G., WEINSTEIN, D., AND HART, D. 2000. Strategies for direct volume rendering of diffusion tensor fields. *IEEE Transactions on Visualization and Computer Graphics 6*, 2, 124–138.

KNISS, J., KINDLMANN, G., AND HANSEN, C. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proceedings Visualization 2001*, 255–63.

KNISS, J., PREMOZE, S., IKITS, M., LEFOHN, A., HANSEN, C., AND PRAUN, E. 2003. Gaussian transfer functions for multi-field volume visualization. In *IEEE Visualization*, 497–504.

KULLBACK, S. 1959. *Information Theory and statistics*. Wiley.

LERMUSIAUX, P. 1999. Data assimilation via error subspace statistical estimation, Part II: Middle Atlantic Bight shelfbreak front simulations and ESSE validation. *Monthly Weather Review 127*, 7, 1408–1432.

LUO, A., KAO, D., AND PANG, A. 2003. Visualizing spatial distribution data sets. In *VisSym'03*, 29–38, 238.