Transfer Functions in Direct Volume Rendering: Design, Interface, Interaction

Gordon Kindlmann

Scientific Computing and Imaging Institute
School of Computing
University of Utah
gk@cs.utah.edu

A principle of direct volume rendering is that visualizations can be created without creating intermediate geometric structure, such as polygons comprising an isosurface, but simply by a "direct" mapping from volume data points to composited image elements. Together with traditional computer graphics elements such as camera, lighting, and shading, the central ingredient in that direct mapping is the assignment of optical properties (opacity, color, *etc.*) to the values comprising the volume dataset. This is the role of the *transfer function* [6, 26, 28]. As simple and direct as that mapping is, it is also extremely flexible, because of the immense variety of possible transfer functions. Because that flexibility is generally unconstrained, the most important parameter in producing a meaningful and intelligible volume rendering is also one of the hardest parameters to set appropriately. This motivates the study of transfer function design, the development of new interfaces for transfer function specification, and a consideration of how interaction techniques in visualization systems can simplify transfer function creation. These course notes attempt to describe the current state of the art of transfer functions in direct volume rendering. Existing approaches can be approximately located in a continuum between *data-driven* and *image-driven* methods, which determine transfer functions based on information extracted from the volume dataset, or from direct volume rendered images, respectively.

Keywords: Volume visualization, Direct volume rendering, Transfer functions, feature detection, data exploration

1 Introduction

Before describing the various approaches to finding and specifying transfer functions, the basic types of transfer function should be outlined. Because they are functions in the strict mathematical sense, when talking about transfer functions it is important to identify the domain and range of transfer function in question. This is also a natural way to categorize the different types of transfer functions.

In the simplest type of transfer function, the domain is the scalar data value (assuming the volume dataset itself is scalar), and the range is opacity. Since the direct volume rendered image is generally composed through repeated applications of the over operator [39], the extent to which a data value is visible in the final image is determined by how much opacity it contributes. Only important features should receive high opacity, so as to not be obscured by opacity from uninteresting regions. Because of the fundamental role opacity plays in creating an intelligible volume visualization, this particular type of transfer function can be given the more specific term *opacity function*. On the other hand, the process of assigning opacity to a volume data point is often called *classification* [28]. Figure 1 illustrates the application of opacity functions to some different datasets, showing slices of the dataset before and after classification, as well as a shaded volume rendering. This figure also demonstrates a very common class of transfer function, based on linear ramps between user-specified control points.

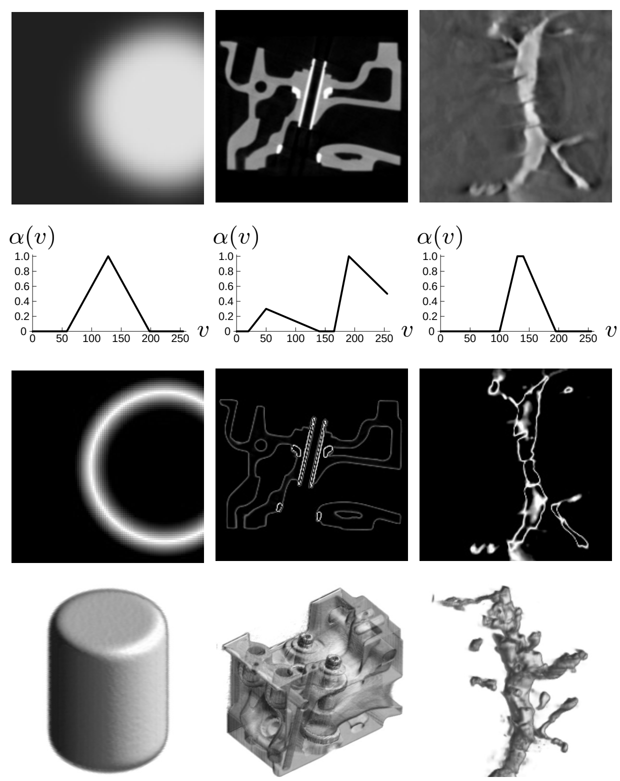

Figure 1: Opacity function demonstration for (from left to right) synthetic cylinder, CT engine block, and electron microscopy dendrite. Going from top to bottom: a slice of the dataset, a plot of the opacity function which assigns opacity α according to data value v, the result of applying the opacity function to the slice, and an image rendered using the shown opacity function.

Opacity functions can be generalized to different types of transfer function by augmenting the function's range. The range often includes color, because color is a simple and natural way to visually distinguish between structures [28]. In general, any optical property that can be represented and composited by computer graphics can be in the range of a transfer function. This includes opacity, color, emittance, scattering by phase functions [23], shading parameters, texturing [17], and even index of refraction [42]. These elements can be represented in varying degrees of sophistication; for instance, opacity can vary according to color, instead of being a single scalar [34].

Transfer functions can also be generalized by increasing the dimension of the function's domain. These can be termed *multi-dimensional* transfer functions. In scalar volume datasets, a useful second dimension is that of gradient magnitude [16, 19, 26]. Because the gradient magnitude characterizes how quickly values are changing in a given neighborhood, its inclusion in the domain of the transfer function allows the distinction between homogeneous regions, and transition regions. This is especially effective in medical contexts, where the feature of interest is often the boundary between two materials. Just as edge detectors in computer vision sometimes employ a second derivative measure [32], further differentiation in structure can be achieved in volume rendering by adding a second derivative to the domain of the transfer function [19]. In fact, the domain of the transfer function need not include the data value. Revealing renderings of internal structure are possible with transfer functions based on the gradient magnitude alone [2], which can be enhanced by modulating opacity according to how orthogonal the gradient vector is to the view vector [3, 41]. Transfer functions based solely on the two-dimensional space of surface principal curvature magnitudes have also be explored [12].

Another type of multi-dimensional transfer function arises from the visualization of volume data which is not a single scalar field. This includes "multi-variate" data, which logically consists of a set of overlapping and related scalar values. Magnetic resonance imaging, for example, can measure many different physical quantities in living tissue (proton density, T1 and T2 relaxation times) [27], and any combination of these can serve as axes in the domain of a multi-dimensional transfer function [25]. Volumetric color data such as produced by the Visible Human Project [35], and multivariate data produced by computational fluid dynamics numerical simulation, are other types of data which can benefit from multidimensional transfer functions [7, 18, 21]. Various types of derivatives can be measured in multi-variate data, including a generalizations of gradient magnitude [4, 5, 43]; these can also play a role in the transfer function [18, 21]. Finally, vector and tensor [17] volume datasets can also be visualized with appropriate applications of multi-dimensional transfer functions.

The enormous flexibility of transfer functions is both a benefit and a drawback. The benefit is that direct volume rendering can be an extremely expressive form of volume visualization because the image can represent such a variety of aspects of the data. The problem is that finding a good transfer function is extremely difficult, and the most common method is a frustrating trial-and-error process. There are at least three reasons for this:

1. Transfer functions have an enormous number of degrees of freedom in which the user can get lost. Even using simple linear ramps, every control point adds two degrees of freedom.

2. The usual interfaces for setting transfer functions (based on moving control points defining a set of linear ramps) are generally not constrained or guided by the dataset in question, or by the problem domain from which the dataset came. The lack of guidance is what forces the user into a trial-and-error mode of interaction, in which the transfer function domain is explored only by observing changes in the volume rendering as a result of incremental adjustments.

3. Transfer functions are inherently non-spatial, in the sense that their assignment of color and opacity does not includes spatial position as a variable in their domain. This can be frustrating if the user is interested in isolating one feature of the volume which is *spatially* localized, but not distinguishable, in terms of data value (or other transfer function domain variables), from the other regions in the volume.

The goal of volume visualization is to produce a clear and informative picture of the important structures in a dataset. Setting a transfer function by trial-and-error is frustrating because it is tantamount to performing feature detection by trial-and-error. Fortunately, recent research in volume visualization has explored the specific problem of transfer function design, and the development of new interfaces for transfer function specification. Existing methods can be roughly categorized with a distinction between *data-driven* and *image-driven* [15, 37, 38]. Data-driven methods extract information from the volume dataset itself, which can be used to constrain the space of transfer functions, or to guide the user towards good transfer function settings. Image-driven methods are instead based on volume rendered images, so that the space of transfer functions is navigated indirectly, by exploring (with varying degrees of automation and interaction) the space of renderings. The following section considers volume visualization methods which are similar to direct volume rendering, but which successfully side-step the difficulties of transfer function specification. Then, the remaining sections consider current methods for transfer function specification in terms of the data-driven and image-driven distinction.

2 Volume Visualization by Spatial Feature Detection

There is a small but growing class of volume rendering techniques which do not employ transfer functions in the traditional sense of the word. Generally, transfer functions do not have positional dependence. That is, color and opacity are assigned on the basis of locally measured properties such as value and gradient magnitude. These quantities obviously vary with position in the volume, but position itself does not appear in the domain of the transfer function, nor does any other quantity which can't be directly expressed in terms of local properties. The variables in the transfer function domain are generally so simple that it is practical to compute them from the dataset "on the fly" during the rendering process, instead of as a seperate pre-process.

However, it is hard to draw an essential distinction between quantities which tend to be computed by a pre-process (such as segmentation based on region growing or clustering), resulting in position-dependent information incompatible with transfer functions, versus those quantities which are so readily computed or approximated (such as gradient magnitude) that it makes perfect sense for them to be in the transfer function domain. Sometimes, for instance, the gradient magnitude is precomputed for reasons of efficiency [24], but one can also imagine ray-tracing approaches that do sophisticated filtered derivative estimation and eigensystem computation at every sample in order to compute coordinates in a curvature based transfer function [12].

Some research in volume rendering avoids traditional transfer functions while maintaining the goal of creating an informative image of the structure of the dataset. This is possible by transforming the problem of transfer function specification to one of feature detection in the spatial domain. The associated parameter space is likely simpler than the space of transfer functions, but there is still useful flexibility. This work is relevant to a consideration of transfer functions because it reminds us of the limitations of volume visualization based on traditional transfer functions, and it may inspire new transfer function domains and applications.

Lürig and Ertl [29] use three-dimensional morphological operations such as erosion and dilation to characterize feature size at a range of scales, then use the results of these operations to generate color-coded visualizations of the volume structure. Fang *et al.* [9] created volume visualizations by applying a sequence of image processing operations such as histogram equalization, smoothing, sharpening, and edge detection, and then using a simple linear ramp opacity function. Hladůvka *et al.* [13] identify regions of high salience in the volume dataset, to facilitate progressive transmission and visualization, by analyzing the smallest eigenvalue of the Hessian of data values, as well as the magnitude of the gradient. Rheingans and Ebert [8, 41] generalize the classification and shading steps of traditional volume rendering to include a series of opacity and color modifications to acheive boundary enhancement, silhouettes, halos, and depth cueing, creating improved visualizations of volume structure.

3 Data-driven Transfer Function Specification

The principle of data-driven transfer function specification is that information derived from the dataset can either guide the user towards appropriate transfer function settings, or can actually constrain the space of transfer functions available in the user interface to a more promising subset. Because the transfer function is position-independent, the information used to generation a transfer function is usually also position-independent. This explains why many data-driven methods are based on histograms and their analysis, since histograms by their nature accumulate information in a position-independent fashion.

Many data-driven methods useful for transfer function specification are actually targeted towards the determination of isovalue in the context of volume visualization by isosurface extraction. Isovalue determination is a similar problem to transfer function specification, in that the effectiveness of the resulting visualization depends entirely on a parameter setting which is applied uniformly throughout the volume. Because the parameter space of isovalues is one-dimensional, it is simpler to navigate, but finding the right isovalue can be taxing, especially with visualization methods or applications which do not support varying the isovalue at interactive rates. Any method which helps determine isovalues is immediately useful for setting opacity functions, because it gives information about which values are more important for showing the structure of a dataset. Rather than giving full opacity to exactly one value, so as to depict an isosurface, we can give opacity to a broader range of values, depending on their relative importance.

The Contour Spectrum is an important technique for isovalue determination [1] in unstructured meshes. By exploiting the mathematical properties of the mesh, important measures of an isosurface, such as surface area, volume, mean gradient magnitude, and topological characteristics, can be computed efficiently. More importantly, the results of these measurements are integrated into the same interface which is used to set the isovalue. By providing a compact visual representation of the metrics evaluated over the range of possible isovalues, the user can readily decide, based on their rendering goals, which isolevel to use. The idea of enriching a user interface for parameter setting with dataset-specific information is a powerful one, which has widespread relevance.

The idea of tracking how the topology of the isosurface changes as the isovalue increases has been used by others as a way of determining important isovalues and transfer function settings. Fujishiro *et al.* [10] use a "Hyper Reeb graph" to depict the isosurface topology at any given isovalue, as well as the isovalues corresponding to critical points where the topology changes. Given some assumptions about what topological characteristics signify important isosurfaces (such as considering important isosurfaces as those with a small number of connected components), this information can be used as guidance to set isovalues and transfer functions. Or, if isosurfaces at the critical values are assumed to be important, setting isovalues and transfer functions can be largely automated.

Pekar *et al.* [36] exploit the divergence theorem of vector calculus [33] to determine which isovalues tend to correspond with strong boundaries in a very efficient manner. Because the gradient vector is always perpendicular to an isosurface, the gradient magnitude is expressible as a dot product with the isosurface normal. Since the surface integral of the gradient magnitude, computed over an isosurface, generally indicates how closely the isosurface corresponds to a significant boundary, the divergence theorem can be employed to replace the surface integral with a volume integral of the gradient divergence, which is the Laplacian. Fortunately, volume integrals can be easily approximated by simple histograms, so this method requires only a single pass through the dataset to create a Laplacian-weighted histogram of data values. Other weighting schemes for emphasizing different surface and curvature characteristics are also possible. The resulting histograms can be used as guidance on either isovalue selection or opacity function creation.

Tenginakai *et al.* [45] use statistical methods to extract information about salient isosurfaces. This approach is based on histograms of values in a local window. Histograms can be described in terms of their moments, which are model-independent statistical estimators of central tendency. The variance is the second moment, and skew and kurtosis are based on the third and fourth moments, respectively. The local window can be centered on any voxel, and the statistical measures of its histogram can be associated with the center voxel. A two-dimensional scatterplot of the relationship between the center voxel's data value, and the associated histogram's statistics, can serve as a "statistical signature" which summarizes the behavior of a statistical measure across the space of data values. By assuming a model of the boundary as a discontinuous junction between two fixed values, one can determine parametric curves which, if present in the statistical signatures, indicate the presence and characteristics of a boundary.

Tenginakai's approach, like Pekar's, produces information which is interpreted in terms of some minimal assumptions about what constitutes importance, in order to produce guidance in setting isovalues or opacity functions. If one uses a different domain for the transfer function, other than data value, then minimal assumptions can be sufficient to vastly simplify the task of transfer function specification. For example, Csébfalvi *et al.* [2] avoid dependence on data value by creating opacity functions based on gradient magnitude, which are easy to set given the assumption that high gradient magnitudes are interesting. Furthermore, by reducing opacity when the gradient direction is aligned with the view vector, the only portion of a surface which contribute opacity is its silhouette. Hladůvka*et al.* [12] use the two-dimensional space of principle surface curvature magnitudes (κ_1, κ_2) to form a transfer function domain, in which is trivial to enhance and color different structures in the volume according to their surface shape. Sato *et al.* [44] use weighting functions of eigenvalues of the Hessian matrix to measure the shape of local structure in terms of edge, sheet, line, and blob. Two and/or three of these measures can be used as axes of a transfer function emphasize different structures in the volume according to their shape, which tends to have biological significance in the context of medical imaging.

The semi-automatic method of Kindlmann and Durkin [15, 16] is based on analysis of a three-dimensional histogram which records the correlation, throughout the given dataset, between data value, gradient magnitude, and the second directional derivative along the gradient direction. The approach can be thought of as having two components; the first is similar to the Contour Spectrum, in that functions computed from the 3D histogram can be used as guidance in setting isovalues and opacity functions. The second component is for semi-automatically generating opacity functions. Based on dataset-specific information and some user-specified controls, it calculates opacity functions which aim to make opaque only those positions in the transfer function domain which reliably correspond to the boundary between two relatively homogeneous regions. One benefit of this method is the ability to generate two-dimensional opacity functions (of data value and gradient magnitude) as well as standard one-dimensional opacity functions, using the same interface. The drawback in the second component is additional conceptual complexity, and the assumption of a particular boundary model, namely a Gaussian-blurred step function. An assessment of how significantly this practically limits the second component's utility has not been performed. Other work has used results from the first component to automate the adjustment of opacity functions for magnetic resonance angiography visualization [40].

4 Image-driven Transfer Function Specification

Image-driven methods use direct volume rendered images to navigate the space of transfer functions. Anyone who has spent time using trial-and-error to find a transfer function can attest to how unintuitive the relationship between the transfer function and the rendered can be. Specifically, small changes in the transfer function can lead to either very small or very large changes in the rendered image. The power of image-based methods is in their ability to create a more uniform and a more intuitive interface, allowing the user to directly find the best rendering, instead of having to manipulate the transfer function itself.

The first work to address the problem of transfer function specification for volume rendering by He *el at.* [11] uses genetic algorithms to "breed" a good transfer function for the dataset in question. The system randomly generates a set of transfer functions, and renders small images for each one. Presented with this set of renderings, the user then picks the few renderings that seem to best display the volume data, and a new population of transfer functions is stochastically generated based on those that the user picked. This process iterates until the user feels that the best transfer function has been found. Alternately, a image-processing metric like entropy, variance, or energy is used as an objective fitness function to evaluate the rendered images without human guidance, and the process eventually converges on a transfer function which maximizes the fitness function. The method succeeds in generating good renderings, and frees the user from having to edit the transfer function manually.

The Design Gallery method presented by Marks *et al.* [31] addresses the problem of "parameter tweaking" in computer graphics in general, with applications to light placement for rendering, motion control for articulated figure animation, as well as transfer functions in direct volume rendering. Here, the goal is not to find the one best parameter setting, but to find as wide a variety of parameter settings as possible, relying on a user-specified metric to determine similarity between rendered images. The system generates a very large set of transfer functions, and then formats small thumbnail renderings resulting from these transfer functions into a two dimensional arrangement called a "design gallery". The user peruses these thumbnail images, selecting the most appealing rendering. Because the method performs organized sampling of the parameter space, it is likely that if a good transfer function exists for a given dataset, the Design Gallery will find it or a close approximation to it. The spatial coherence of renderings with similar appearance facilitates finding the best possible setting.

Other recent work has used different approaches to organize the rendered images in a way which facilitates finding good transfer functions. König and Gröller [22] use an approach which could be termed *thumbnail parameterization*, whereby families of transfer functions are displayed by linear arrangement of thumbnail renderings, for exploring one-parameter and two-parameter families of transfer functions. For example, this facilitates finding the right data value at which to center a peak in the opacity function, or finding the right combination of opacities by which to scale two separate peaks in an opacity function. Ma [30] developed a graph-based interface which organizes visualization results according to the history of parameter changes. Each node in the graph is a thumbnail rendering; the relationship between a change in rendering parameter and the resulting visualization is depicted by an edge connecting renderings before and after the change. Jankun-Kelly and Ma [14] have developed a spreadsheet-like interface for recording and representing the parameter space exploration processes. Illustrating the relationship between rendering parameters and the resulting visualization in organized and systematic fashion helps users find and share effective visualization results.

5 Discussion

The distinction between image-driven and data-driven methods for setting transfer functions is not an essential one. It is possible, for instance, to use rendered images to guide transfer function specification, but do so in a way which is also constrained by the underlying data. Kniss *et al.* [20] proposed an method of *dual-domain interaction* which straddles the image and data domains. As points in the volume are queried with a 3D data probe, or by clicking on the image of a cutting plane, the transfer function is updated to assign opacity in a region around the transfer function domain coordinates of the query location. As with image-driven methods, interaction is in the context of the rendered image, and direct interaction in the transfer function domain is avoided (though not disallowed), but the transfer function generated is still constrained by the properties of the given dataset, similar to data-driven methods.

Comparisons of transfer function specification methods implicitly involve a more fundamental question: what kind of tools benefit the different kinds of visualization needs? More specifically, how much knowledge (on the part of the user) can be assumed when creating interfaces for handling complicated parameter setting? When is it better to simply create guidance through a parameter space, versus enforce constraint to some heuristically determined subset of it? What is the appropriate amount of automation? What is the best context to present new information extracted from a given dataset, but beyond what is used in any particular parameter setting task? The Design Galleries method of setting transfer functions is a very effective way for a novice user to find a good visualization, without any knowledge of the associated space of transfer functions. However, it doesn't generate or directly express information about inherent structure in the dataset, which the Contour Spectrum does by graphing different isosurface metrics evaluated over the space of isovalues. On the other, this interface is useful only after we understand what the different metrics represent, and have formed an idea of what constitutes an interesting isovalue in terms of those metrics. Like most other data-driven tools, the Contour Spectrum is useful because it provides *guidance*, while Kindlmann's semi-automatic method, used in its entirety, actually constrains the user to a significantly smaller subset of the space of opacity functions. This may seem unnecessary for one-dimensional opacity functions, but such constraint is more readily appreciated when creating two-dimensional opacity functions, which have a vastly greater number of degrees of freedom.

Because the transfer function is the central ingredient in creating a direct volume rendering, it is also defines a fundamental limit on what kinds of visualization can be done by direct volume rendering. Suppose there are two structures which can be readily separated in the spatial domain of the volume dataset, and which are known to correspond to different features in the underlying object. Suppose also that the transfer function domain coordinates for the voxels in one structure overlap with those for the other structure. It is impossible to visualize these structures in isolation using direct volume rendering. As long as the transfer function is the sole source of opacity assignment, structures can be distinguished in the rendering only to the extent that they can be separated in the transfer function domain. As mentioned in Section 1, this limitation, and the relationship it imposes between the transfer function and the rendered image, is part of the reason that setting transfer functions is unintuitive and time-consuming. The methods described here represent different strategies for dealing with the problem of the transfer function: either by side-stepping the problem entirely and doing spatial feature detection (Section 2), by using something as intuitive as rendered images to indirectly navigate the unintuitive relationship between transfer function and rendering (Section 4), or by combining information extracted from the dataset with heuristics about feature importance to create guidance or constraint in the transfer function space. Future work in volume rendering, transfer function design, and visualization interaction techniques, will surely produce novel ways of extending transfer functions and enriching user interfaces to make direct volume rendering a more powerful visualization tool.

References

[1] Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. The Contour Spectrum. In *Proceedings Visualization '97*, pages 167–173, 1997.

[2] Balázs Csébfalvi and Eduard Gröller. Interactive Volume Rendering based on a "Bubble Model". In *Proceedings Graphics Interface 2001*, pages 209–216, June 2001.

[3] Balázs Csébfalvi, Lukas Mroz, Helwig Hauser, Andreas König, and Eduard Gröller. Fast Visualization of Object Contours by Non-Photorealistic Volume Rendering. In *Proceedings EUROGRAPHICS 2001*, volume 20(3), pages 452–460, 2001.

[4] A. Cumani, P. Grattoni, and A. Guiducci. An edge-based description of color images. *GMIP*, 53(4):313–323, 1991.

[5] Silvano Di Zenzo. A Note on the Gradient of a Multi-Image. *Computer Vision, Graphics, and Image Processing*, 33(1):116–125, Jan 1986.

[6] R. A. Drebin, L. Carpenter, and P. Hanrahan. Volume Rendering. *ACM Computer Graphics (SIGGRAPH '88 Proceedings)*, pages 65–74, August 1988.

[7] David Ebert, Christopher Morris, Penny Rheingans, and Terry Yoo. Designing Effective Transfer Functions for Volume Rendering from Photographic Volumes. *IEEE Transactions on Visualization and Computer Graphics*, (to appear) 2002.

[8] David Ebert and Penny Rheingans. Volume Illustration: Non-Photorealistic Rendering of Volume Models. In *Proceedings Visualization 2000*, pages 195–202. IEEE, October 2000.

[9] Shiaofen Fang, Tom Biddlecome, and Mihran Tuceryan. Image-Based Transfer Function Design for Data Exploration in Volume Visualization. In *Proceedings IEEE Visualization 1998*, pages 319–326,546. IEEE, October 1998.

[10] Issei Fujishiro, Taeko Azuma, and Yuriko Takeshima. Automating Transfer Function Design for Comprehensible Volume Rendering Based on 3D Field Topology Analysis. In *Proceedings IEEE Visualization*, pages 467–470,563, October 1999.

[11] Taosong He, Lichan Hong, Arie Kaufman, and Hanspeter Pfister. Generation of Transfer Functions with Stochastic Search Techniques. In *Proceedings Visualization '96*, pages 227–234, 1996.

[12] Jiří Hladůvka, Andreas König, and Eduard Gröller. Curvature-Based Transfer Functions for Direct Volume Rendering. In Bianca Falcidieno, editor, *Spring Conference on Computer Graphics 2000*, volume 16, pages 58–65, May 2000.

[13] Jiří Hladůvka, Andreas König, and Eduard Gröller. Salient Representation of Volume Data. In D. Ebert, J. M. Favre, and R. Peikert, editors, *Data Visualization 2001, Proceedings of the Joint Eurographics – IEEE TVCG Symposium on Visualization*, pages 203–211,351, 2001.

[14] T J Jankun-Kelly and Kwan-Liu Ma. Visualization Exploration and Encapsulation via a Spreadsheet-Like Interface. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):275–287, July-September 2001.

[15] Gordon Kindlmann. Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering. Master's thesis, Cornell University, Ithaca, NY, January 1999.

[16] Gordon Kindlmann and James W. Durkin. Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering. In *IEEE Symposium On Volume Visualization*, pages 79–86, 1998.

[17] Gordon Kindlmann and David Weinstein. Hue-Balls and Lit-Tensors for Direct Volume Rendering of Diffusion Tensor Fields. In *Proceedings Visualization 1999*, pages 183–189, October 1999.

[18] Joe Kniss, Charles Hansen, Michel Grenier, and Tom Robinson. Volume Rendering Multivariate Data to Visualize Meteorological Simulations: A Case Study. In *VisSym 2002, Joint EUROGRAPHICS - IEEE TCCG Symposium on Visualization*, (to appear) 2002.

[19] Joe Kniss, Gordon Kindlmann, and Charles Hansen. Interactive Volume Rendering Using Multi-Dimensional Transfer Functions and Direct Manipulation Widget. In *Proceedings Visualization 2001*, pages 255–262, October 2001.

[20] Joe Kniss, Gordon Kindlmann, and Charles Hansen. Interactive Volume Rendering Using Multi-Dimensional Transfer Functions and Direct Manipulation Widgets. In *Proceedings IEEE Visualization 2001*, pages 255–262,562, October 2001.

[21] Joe Kniss, Gordon Kindlmann, and Charles Hansen. Multi-Dimensional Transfer Functions for Interactive Volume Rendering . *IEEE Transactions on Visualization and Computer Graphics*, (to appear) 2002.

[22] Andreas König and Eduard Gröller. Mastering Transfer Function Specification by Using VolumePro Technology. In Tosiyasu L. Kunii, editor, *Spring Conference on Computer Graphics 2001*, volume 17, pages 279–286, April 2001.

[23] Wolgang Krueger. The application of transport theory to visualization of 3-D scalar data fields. *Computers in Physics*, pages 397–406, July-August 1991.

[24] Philip Lacroute and Marc Levoy. Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transform. In *ACM Computer Graphics (SIGGRAPH '94 Proceedings)*, pages 451–458, July 1994.

[25] David H. Laidlaw. *Geometric Model Extraction from Magnetic Resonance Volume Data*. PhD thesis, Department of Computer Science, California Institute of Technology, Pasadena, CA, May 1995.

[26] Marc Levoy. Display of Surfaces from Volume Data. *IEEE Computer Graphics & Applications*, 8(5):29–37, 1988.

[27] Zhi-Pei Liang and Paul C Lauterbur. *Principles of Magnetic Resonance Imaging*. IEEE Press, New York, 2000.

[28] Barthold Lichtenbelt, Randy Crane, and Shaz Naqvi. *Introduction to Volume Rendering*, chapter 4. Prentice-Hall, New Jersey, 1998.

[29] Christoph Lürig and Thomas Ertl. Hierarchical Volume Analysis and Visualization Based on Morphological Operators. In *Proceedings IEEE Visualization 1998*, pages 335–341,548. IEEE, October 1998.

[30] Kwan-Liu Ma. Image Graphs - A Novel Approach to Visual Data Exploration. In *Proceedings IEEE Visualization*, pages 81–88,513, October 1999.

[31] J. Marks, B. Andalman, P.A. Beardsley, and H. Pfister et al. Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation. In *ACM Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 389–400, August 1997.

[32] D. Marr and E. C. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London*, B 207:187–217, 1980.

[33] Jerrold E. Marsden and Anthony J. Tromba. *Vector Calculus*, chapter 2.6. W.H. Freeman and Company, New York, 1996.

[34] Herke Jan Noordmans, Hans T M van der Voort, and Arnold W M Smeulders. Spectral Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 6(3):196–207, July-September 2000.

[35] National Library of Medicine (U.S.) Board of Regents. Electronic Imaging: Report of the Board of Regents. U.S. Department of Health and Human Services, Public Health Service, National Institutes of Health. NIH Publication 90-2197, 1990.

[36] Vladimir Pekar, Rafael Wiemker, and Daniel Hempel. Fast Detection of Meaningful Isosurfaces for Volume Data Visualization. In *Proceedings IEEE Visualization 2001*, pages 223–230, 2001.

[37] Hanspeter Pfister, Bill Lorensen, Chandrajit Bajaj, Gordon Kindlmann, Will Schroeder, Lisa Sobeierajski Avila, Ken Martin, Raghu Machiraju, and Jinho Lee. The Transfer Function Bake-Off. *IEEE Computer Graphics and Applications*, 21(3):16–22, May/June 2001.

[38] Hanspeter Pfister, Bill Lorensen, Chandrajit Bajaj, Gordon Kindlmann, Will Schroeder, and Raghu Machiraju. The Transfer Function Bake-Off. In *Proceedings Visualization 2000*, pages 523–526, October 2000.

[39] Thomas Porter and Tom Duff. Compositing Digital Images. In *ACM Computer Graphics (SIGGRAPH '84 Proceedings)*, pages 253–259, July 1984.

[40] Christof Rezk-Salama, Peter Hastreiter, Jörg Scherer, and Günther Greiner. Automatic Adjustment of Transfer Functions for 3D Volume Visualization. In *Proceedings Vision, Modeling, and Visualization 2000*, pages 357–364, November 200.

[41] Penny Rheingans and David Ebert. Volume Illustration: Nonphotorealistic Rendering of Volume Models. *IEEE Transactions on Visualization and Computer Graphics*, 7(3):253–264, July-September 2001.

[42] David Rodgman and Min Chen. Refraction in Discrete Ray Tracing. In *Proceedings Volume Graphics 2001*, pages 3–17,403, June 2001.

[43] Guillermo Sapiro. Color Snakes. *CVIU 68(2)*, pages 247–253, 1997.

[44] Yoshinobu Sato, Carl-Fredrik Westin, Abhir Bhalerao, Shin Nakajima, Nobuyuki Shiraga, Shinichi Tamura, and Ron Kikinis. Tissue Classification Based on 3D Local Intensity Structures for Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):160–180, April-June 2000.

[45] Shivaraj Tenginakai, Jinho Lee, and Raghu Machiraju. Salient Iso-Surface Detection with Model-Independent Statistical Signatures. In *Proceedings IEEE Visualization 2001*, pages 231–238, 2001.