Transfer Functions on a Logarithmic Scale for Volume Rendering

Simeon Potts Torsten Möller

Graphics, Usability and Visualization (GrUVi) Lab
School of Computing Science
Simon Fraser University

Abstract

Manual opacity transfer function editing for volume rendering can be a difficult and counter-intuitive process. This paper proposes a logarithmically scaled editor, and argues that such a scale relates the height of the transfer function to the rendered intensity of a region of particular density in the volume almost directly, resulting in much improved, simpler manual transfer function editing.

Key words: Volume Graphics, Transfer Functions, User Interfaces

1 Introduction

In volumetric rendering, an opacity transfer function is used to control what parts of the data are visible (and their relative rendered opacities). In the simplest case, classification of the data is based on the scalar values of the data, usually recorded as an 8 or 16-bit integer for each discrete sample point on a grid. When a transfer function is applied, these values are mapped to a real number typically between 0 and 1 that represents the opacity (per unit length) associated with the data point, with 1 being opaque and 0 being transparent. More sophisticated classification of the data, such as classification that depends on the gradient magnitude as well as the data value, can be specified using multi-dimensional transfer functions [11, 9].

Transfer functions are often manually created using an editable graph, relating the original data values on the horizontal axes to associated opacities on the vertical axis. This paper addresses the vertical scale on this editable graph, and presents the advantages of using a logarithmic scale instead of the usual linear one.

After a review of current transfer function approaches in Section 2, Section 3 provides the motivation behind using a logarithmic scale to specify transfer functions. In Section 4 we compare images produced using transfer functions specified on a linear scale with images produced using transfer functions specified on a logarithmic scale. Section 5 summarizes our results and looks at possible future work.

2 Previous Work

Transfer functions are an essential part of the volume rendering pipeline. Drebin et al. [3] have been using them in order to assign material percentages to the given data values. Levoy [11] used the data value and its derivative magnitude in order to visualize boundaries of equal thickness or strength. Since these methods (also classified as *data-centric*) are not intuitive to use, several researchers have tried to enhance the user interfaces in order to make data exploration more effective.

Recently the concept of material percentage volume has been extended by Bergner et al. [2] using spectral transfer functions and a user interface based on a light-dial in order to explore various parameter combinations.

Bajaj et al. [1] compute several isosurface metrics of the data, such as the enclosed area or volume as well as the gradient surface integral for the selected iso-value. These metrics serve as a guide for selecting iso-values or more complex transfer functions. The hope is that the minima or maxima of these curves represent interesting places in the data that are worth visualizing. This is also known as the contour spectrum. Pekar et al. [13] suggest Laplacian-weighted histograms that compute the gradient magnitude over the isosurface, indicating strong transition regions at its maxima. They use the divergence theorem in order to compute the histograms efficiently.

Kindlmann et al. [7] suggest the use of 2D histograms of the gradient magnitude as well as the second derivative of the data in the direction of the gradient. They argue that within transition regions the shape of these histograms takes on a particular arch-like structure, which can be used to guide the user to find interesting transition regions in the data. They further suggest the semi-automatic creation of transfer functions, where the user simply inputs a surface distance to opacity map in order to visualize the data. This idea has been extended by Tenginakai et al. [15] using a more general statistical approach using localized k-order central moments.

Kniss et al. [9] suggest a local probing of the underlying data which uses the 2D histograms of Kindlmann et al. [7] in order to find boundary regions within the data

interactively. They describe a user interface for the assignment of color and opacity values. However, the effectiveness of this user interface still needs to be investigated.

Other data measures have been investigated for use in transfer function design. Curvature as a transfer function criterion has been introduced by Hladůvka et al. [5]. A robust computation of curvature and its use for transfer functions and non-photorealistic rendering has been suggested by Kindlmann et al. [8].

Opacity transfer functions have also been efficiently adapted to very noisy ultrasound data by Hönigmann et al. [6].

All these methods are known as *data-centric* methods and tend to have fairly counter-intuitive user interfaces. A second class of methods known as *image-centric* approaches try to rectify this by steering the transfer function design using a set of rendered images. This tends to be rather computationally challenging, since volume rendering tends to be a computationally demanding task.

The space of all transfer functions is rather large to explore. He et al. [4] use genetic algorithms in order to create a population of transfer functions starting with a user-given or random set of transfer functions. This requires, however, a fitness function. Marks et al. [12] attach a user interface to this process which groups the population into visually similar renderings and lets the user zoom in, select and refine this population.

König and Gröller [10] create a visual interface allowing the creation of linear combinations of small opacity regions. It also has an interactive color picker. This interface was realized through the real-time rendering capabilities of a VolumePro graphics card [14].

The goal of this paper is to improve typical data-centric user interfaces and hence the data exploration process. We will show that a linear mapping of the data value to the opacity will lead to unintuitive results, which can be rectified by a logarithmic mapping.

3 Logarithmically Scaled Transfer Functions

We began to study the graph that a transfer function editor employs out of frustration with the counter-intuitive results obtained when rendering volumetric data. A typical opacity transfer widget can be seen in Figure 1, while the resulting rendering is depicted in Figure 9 (see colour section).

We observed that when rendering data sets with relatively large regions of roughly uniform density, the resulting images were most sensitive to detailed changes in the transfer function when the opacity was set nearly to zero. Editing the transfer function within the lowest five to ten percent of the range often resulted in the most visible differences in the rendered images, allowing us to differentiate finer structures in the data, while larger values corresponded to renderings that appeared nearly opaque, obscuring large portions of the finer structures present. However, for smaller, thinner regions of some density in the data, it was sometimes necessary to use higher opacity values to get the region to appear in the rendering at all.

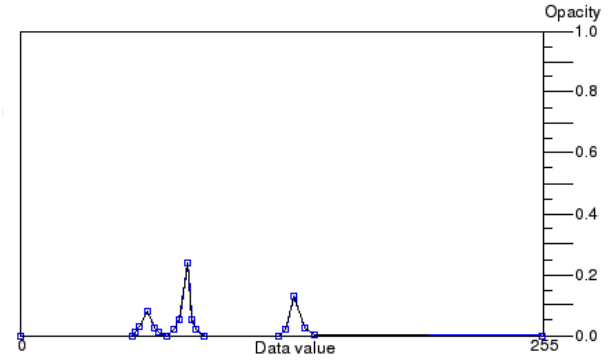

Figure 1: A typical transfer function widget. The x-axis denotes the data value while the y-axis denotes the opacity for the respective data value. The top bar denotes the colours assigned to the respective data values.

These observations about volume renderings are due to *compositing*, an iterative process that closely approximates the occlusion along a viewing ray in a volume. It turns out that when even a small opacity value is assigned to a region in the data of any significant size, the final rendered intensity will quickly increase to near-opaque, and it's color will be due primarily to the sampled voxels closest to the viewer. Each sample is obscured in part by every sample in front of it, resulting in a contribution to the intensity that is exponentially less as distance through the volume increases. We illustrate the situation by graphing the intensity of a rendered pixel as a function of both the distance the viewing ray travels through the volume (thickness) of the transfer function value (density) in Figure 10 (see colour section).

To put Figure 10 in the context of transfer function editing, consider that the horizontal axis corresponds to the thickness of a homogeneous region in the volume that we wish to visualize (as a percentage of the total volume size) and the vertical axis corresponds to the transfer function values that the user could set for the region. The graph displays the intensity that will result from compositing the region by rendering it, as a set of ranges or contours. Ideally, if we estimate the size of a particular homogeneous region relative to that of the volume, we can set the intensity of the region such that it corre-

sponds to any one of the intensities on the graph that we desire. The apparent difficulty is that most of the bands of intensity on the graph are very narrow and are highly dependent on the size of the region in question.

The fact that the bands span an extremely narrow region for depths above 30 percent of the volume means that for such regions, differences in transfer function values as small as 0.001 (in the region near zero) will visibly change the image. Even if an entire screen is used for a transfer function editor, a user must be able to specify transfer functions to the accuracy of a single pixel to make such changes. At the same time, values above 0.05 all map to nearly opaque, resulting in 95 percent of the screen space being wasted.

We propose a better approach by observing that we can scale the vertical axis of the graph in any way we wish, though the horizontal axis scale is fixed by the rendering method (alpha compositing). If we scale the transfer function opacities logarithmically, the result is a graph as in Figure 11 (see colour section). In this graph, the regions of intensity are better distributed, without "pushing" any of them off of the graph. Consequently, it would be easier, both conceptually and physically, for a user to precisely control the intensity of a region in the volume. This should result in more efficient transfer function editing, which is significant since transfer function editing is one of the most time-consuming aspects of creating clear and informative renderings of medical, mathematical or generally scientific data sets.

Naturally, one can convert a linear transfer function view into a logarithmically one quite easily. If we denote with d the dimension of the data set, then let ρ^d represent the "optical density" (transparency) of the data set, while $d\alpha$ would represent the corresponding "physical thickness" of the data set. Adjusting for a bijective mapping from the interval $[0, 1]$ onto and into itself, we can derive the following relationship between the linearly scaled opacity transfer function α of Figure 10 and the logarithmically scaled opacity transfer function α' of Figure 11 (see colour section):

$$(\rho^{\alpha'})^d = d^{\alpha'} = d\alpha + (1.0 - \alpha) \tag{1}$$

This assumes an assignment of an "optical density" for a particular material of $\rho = (\sqrt[d]{d})^{\alpha'}$, which can be considered a user controlled parameter. The above equation can be expressed as

$$\alpha' = 1.0 - \frac{1}{-\mathrm{a}} \ln((1 - e^{-\mathrm{a}})\alpha + e^{-\mathrm{a}}) \tag{2}$$

Where α' is the scaled transfer function value obtained from α, the transfer function value, and

$$\mathrm{a} = \max(\ln d, 1) \tag{3}$$

To map a point on the logarithmic graph back to a transfer function value (which is necessary during interactive editing) the inverse transformation is used:

$$\alpha = \frac{e^{-\mathrm{a}(1-\alpha')} - e^{-\mathrm{a}}}{1 - e^{-\mathrm{a}}} \tag{4}$$

where α is the transfer function value obtained from α', the logarithmically scaled value from the graph. The main logarithmic mapping is $\alpha = e^{-\mathrm{a}(1-\alpha')}$, where a is a user controlled parameter. Equation 4 results by ensuring a bijective mapping from the interval $[0, 1]$ onto and into itself.

4 Results

Here we present renderings of a data sets using the traditional (linear) opacity transfer functions in comparison to our new logarithmically scaled opacity transfer functions.

We start with a synthetic data set which illustrates the problem nicely. Our data set consists of concentric spheres generated by the following function

$$f(x, y, z) = \max(0, 1 - \lceil 2\frac{n}{d}\sqrt{x^2 + y^2 + z^2}\rceil/n) \tag{5}$$

where $f(x, y, z)$ is the density of the data on a scale from 0 to 1, d is the largest dimension of the volume (in our example $d = 128$), n is the number of spheres (in Figure 2 and 3 we used $n = 10$) and the x, y, and z coordinates are in voxel units with the centre of the volume as the origin. Figure 2 is a volume rendering of the dataset with a linear transfer function, and Figure 3 is a rendering of the same dataset with a transfer function that is linear when graphed on the logarithmic scale (resulting in an exponential function).

We can clearly see that the exponential transfer function shows almost all layers of the data, while the linear transfer function shows roughly half of the layers. We expect that in this particular case a linear transfer function should generate an image where all of the layers are visible, increasing uniformly in intensity, a result which is only achieved if the linear function is drawn on a logarithmic scale.

The applicability of logarithmically-scaled transfer functions is not limited to data sets where there are large regions of uniform density. Figure 6 and Figure 7 demonstrate the effect of an exponential transfer function on a volume consisting of thin concentric spherical shells only one voxel in width. The linear transfer function is shown in Figure 4 and the logarithmic transfer function is shown

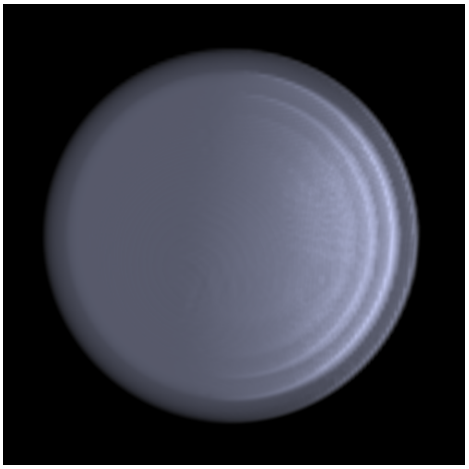

Figure 2: Rendering using a linear transfer function

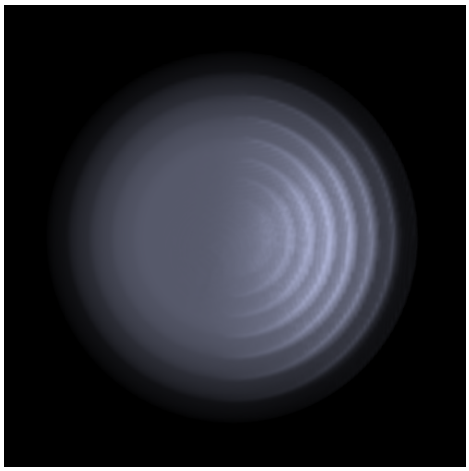

Figure 3: rendering using an exponential transfer function

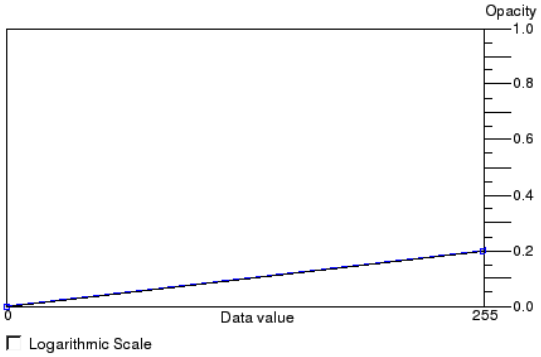

Figure 4: A linear transfer function

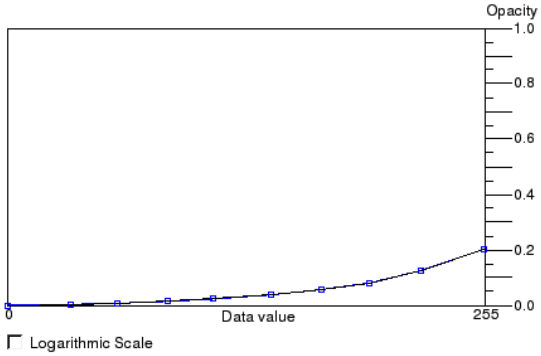

Figure 5: An exponential transfer function

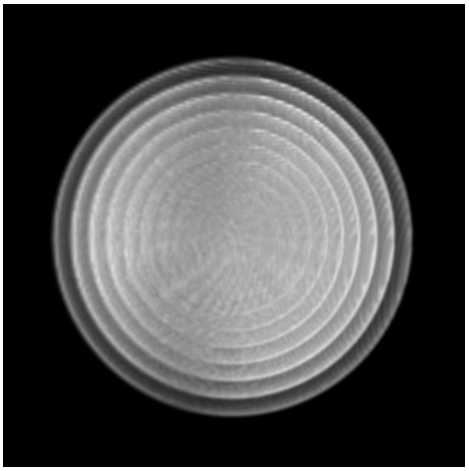

Figure 6: Thin concentric shells rendered with a linear transfer function

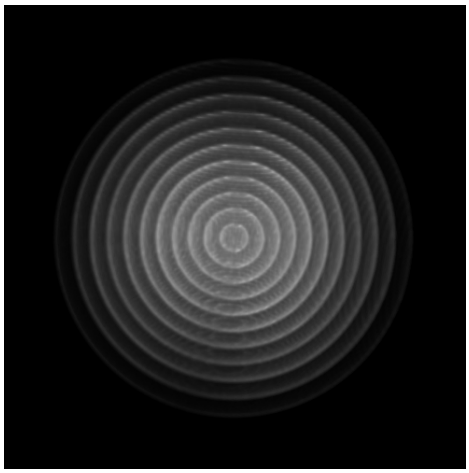

Figure 7: Thin concentric shells rendered with an exponential transfer function

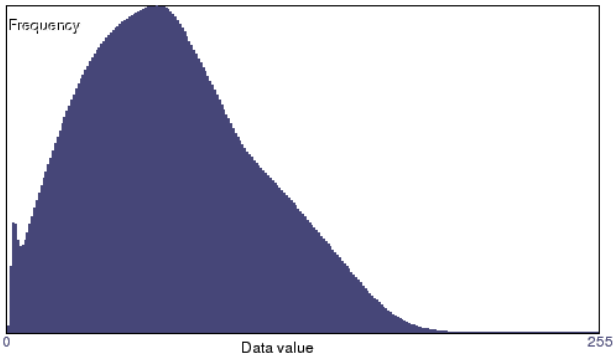

Figure 8: Laplacian-weighted histogram of the head data set rendered in Figure 12

in Figure 5. The depiction of fine structures is clearly superior in the logarithmic transfer function case.

Perhaps the difference is best shown on a real dataset. In Figure 8, 12-17 we display a Laplacian-weighted histogram [13] of a human head dataset (MRI of dimensions 256x256x129), followed by two transfer function assignments and renderings of the data.

The rendered head images and the corresponding transfer functions, shown both on a linear and logarithmic scale, illustrate the situation where a small change in the transfer function can have a very visible effect on the rendered image (here the blue region is visible only in Figure 12). Scaling the transfer function logarithmically makes the change in the transfer function far more perceptible to the observer and easier to execute for the user, since less pixel accuracy with a pointing device is required on the logarithmic scale.

5 Conclusion and Future Research

The principles behind volume rendering are derived from radiation theory, a realistic and physically-based model. However, the primary purpose of volume rendering is typically not to create fully realistic renderings, but to aid in the understanding of the data set in question, revealing as much detail of the data as is desired. Transforming the transfer function opacity scale to a logarithmic one takes the physically-based model into full consideration, and without altering it gives the user a model that they can treat as linear, correlating the magnitude of their changes to the transfer function to the magnitude of the changes they see in the renderings.

We conclude that transfer function editors with a logarithmic scale (taking into account the size of the volume in voxels) would ease the cognitive overhead of designing transfer functions by producing a more intuitive and direct response. We demonstrate this by studying the influence of data points on the weight of the rendering integral in Section 3 and through our own use of the rendering software.

Besides opacity transfer functions, the colour component is just as unintuitive in typical transfer function designs. However, this can not be fixed by a logarithmic scale. We are currently investigating new user interfaces that allow a more traditional data-centric transfer function approach, without loosing the intuition typically present in image-centric transfer function design approaches.

Acknowledgements

Rendered images were produced with vuVolume, a rendering suite developed at Simon Fraser University. We would like to thank the members of the GrUVi lab for their discussions. Head dataset is courtesy of Jeff Orchard. Tooth data courtesy of B. Lorensen, General Electric. This work was partially funded by the Advanced Systems Institute (ASI) of British Columbia as well as the Natural Science and Engineering Research Council of Canada.

References

[1] Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. The contour spectrum. In *Proceedings of the 8th IEEE Conference on Visualization '97*, pages 167–175. IEEE Computer Society Press, 1997.

[2] Steven Bergner, Torsten Möller, Mark S. Drew, and Graham D. Finlayson. Interactive spectral volume rendering. In *Proceedings of the conference on Visualization '02*, pages 101–108. IEEE Computer Society, 2002.

[3] Robert A. Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 65–74. ACM Press, 1988.

[4] Taosong He, Lichan Hong, Arie Kaufman, and Hanspeter Pfister. Generation of transfer functions with stochastic search techniques. In *Proceedings of the 7th conference on Visualization '96*, pages 227–234. IEEE Computer Society Press, 1996.

[5] 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 (SCCG 2000)*, volume 16, pages 58–65, May 2000.

[6] Dieter Hönigmann, Johannes Ruisz, and Christoph Haider. Adaptive design of a global opacity transfer function for direct volume rendering of ultrasound

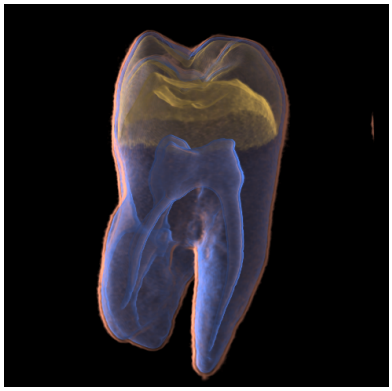

Figure 9: A rendering of a tooth after applying the transfer function in Figure 1.

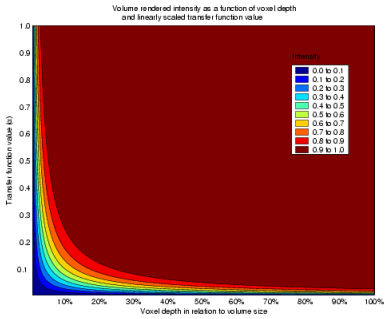

Figure 10: Data contribution depending on the opacity assigned and depth from the eye-point (opacities on a linear scale). We assume a homogenous block of data.

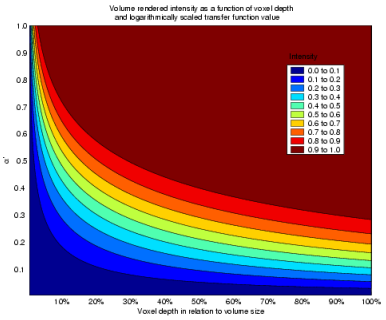

Figure 11: Data contribution depending on the opacity assigned and depth from the eye-point (opacities scaled logarithmically). We assume a homogenous block of data.

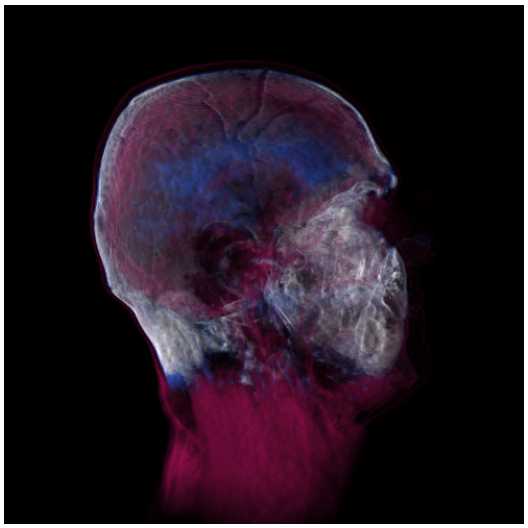

Figure 12: Rendering of the head data set

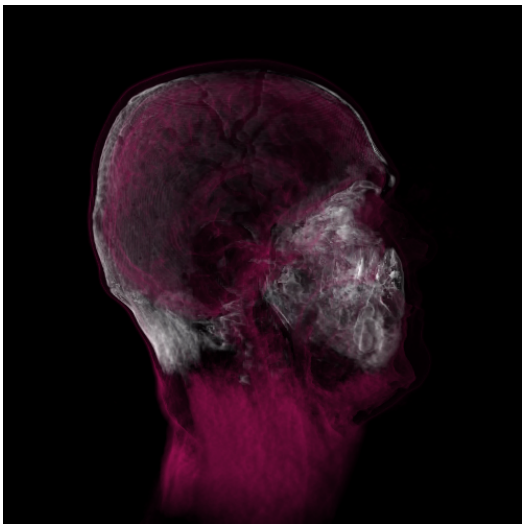

Figure 13: A second rendering of the head data set

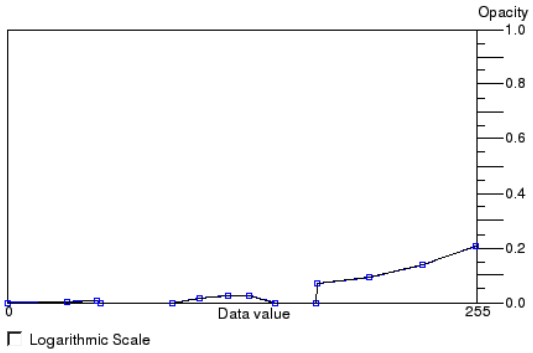

Figure 14: A linear transfer function for the MRI head, resulting in the image in Figure 12

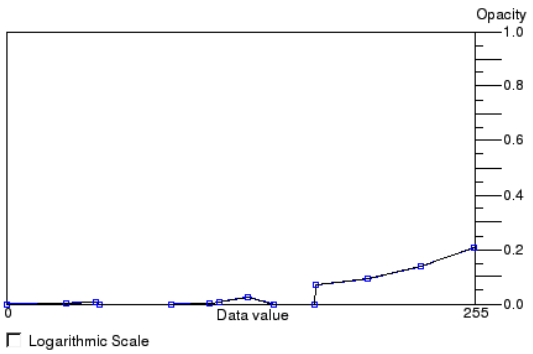

Figure 15: A second transfer function on a linear scale resulting in the rendering in Figure 13.

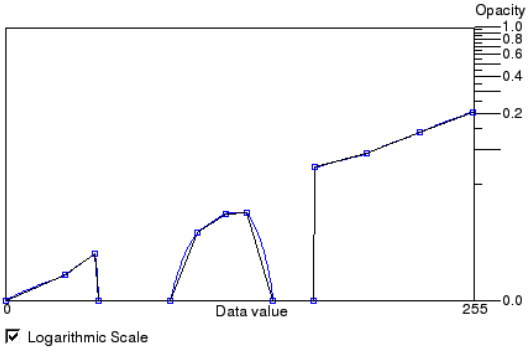

Figure 16: The same transfer function as in Figure 14 on a logarithmic scale. Both transfer functions result in the identical rendering depicted in Figure 12.

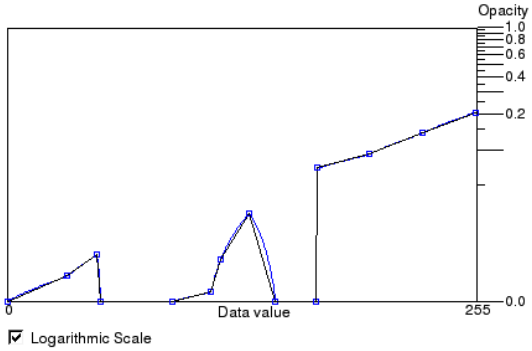

Figure 17: The same transfer function as in Figure 15 on a logarithmic scale. Both transfer functions result in the identical rendering depicted in Figure 13.

data. In *Proceedings of the IEEE Conference on Visualization '03*. IEEE Computer Society, 2003.

[7] Gordon Kindlmann and James W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of the 1998 IEEE Symposium on Volume visualization*, pages 79–86. ACM Press, 1998.

[8] Gordon Kindlmann, Ross Whitaker, Tolga Tasdizen, and Torsten Möller. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *Proceedings of the IEEE Conference on Visualization '03*. IEEE Computer Society, 2003.

[9] Joe Kniss, Gordon Kindlmann, and Charles Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002.

[10] Andreas König and Eduard Gröller. Mastering transfer function specification by using volumepro technology. In *Spring Conference on Computer Graphics 2000 (SCCG 2001)*, volume 17, pages 279–286, April 2001.

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

[12] J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. Design galleries: a general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 389–400. ACM Press/Addison-Wesley Publishing Co., 1997.

[13] Vladimir Pekar, Rafael Wiemker, and Daniel Hempel. Fast detection of meaningful isosurfaces for volume data visualization. In *Proceedings of the IEEE Conference on Visualization '01*, pages 223–230. IEEE Computer Society, 2001.

[14] Hanspeter Pfister, Jan Hardenbergh, Jim Knittel, Hugh Lauer, and Larry Seiler. The volumepro real-time ray-casting system. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 251–260. ACM Press/Addison-Wesley Publishing Co., 1999.

[15] Shivaraj Tenginakai, Jinho Lee, and Raghu Machiraju. Salient iso-surface detection with model-independent statistical signatures. In *Proceedings of the IEEE Conference on Visualization '01*. IEEE Computer Society, 2001.