Size-based Transfer Functions: A New Volume Exploration Technique

Carlos D. Correa, *Member, IEEE*, and Kwan-Liu Ma, *Senior Member, IEEE*

Abstract— The visualization of complex 3D images remains a challenge, a fact that is magnified by the difficulty to classify or segment volume data. In this paper, we introduce size-based transfer functions, which map the local scale of features to color and opacity. Features in a data set with similar or identical scalar values can be classified based on their relative size. We achieve this with the use of scale fields, which are 3D fields that represent the relative size of the local feature at each voxel. We present a mechanism for obtaining these scale fields at interactive rates, through a continuous scale-space analysis and a set of detection filters. Through a number of examples, we show that size-based transfer functions can improve classification and enhance volume rendering techniques, such as maximum intensity projection. The ability to classify objects based on local size at interactive rates proves to be a powerful method for complex data exploration.

Index Terms—Transfer Functions, Interactive Visualization, Volume Rendering, Scale Space, GPU Techniques.

1 Introduction

One of the challenges visualizing volumetric data sets is the rendering of features of interest so that they stand out from the inherent noise and other less interesting features, which may occlude or add clutter. A number of approaches have been proposed to deal with this problem, including rendering operations, segmentation and manipulation techniques. Despite the proliferation of rendering and manipulation techniques, the use of transfer functions is still the predominant method. Transfer functions map scalar data values at each sample point to color and opacity. In many cases, first and second order derivatives of the scalar field are used to improve classification or add lighting to volume rendered data [14, 8]. In other cases, it is possible to incorporate geometric or semantic information when applying transfer functions, such as spatial coordinates, curvature, spatial frequency or user-defined tags. Some of these result in a multi-dimensional transfer function space, and require efficient manipulation mechanisms to be properly deployed in visualization systems. In this paper, we propose a new dimension to define transfer functions: the relative *size* of features. Size, understood as the magnitude of the spatial extents of a given part of a volume, is an intuitive concept that can be manipulated more easily than high-dimensional values.

With size-based transfer functions (SBTF), it is now possible to map color and opacities based on the relative size of features. For example, detection of aneurysms requires the visualization of convoluted vascular structures. Large and small features alike often appear with similar or identical density in MR imaging or angiography. If we incorporate size to derive a transfer function, large parts, such as an aneurysm, can be mapped to different color and opacities than the normal vessels. An example is shown in Fig. 1. Note how the aneurysm is clearly marked in a different way than small vessels. In other cases, small features are also important. For example, the detection of cancer cells requires the analysis of small features on a mammograph, including veins, arteries and cancer tumors. The ability to visualize the relative size of features enhances the presence of these cells over other larger structures. A similar need exists for detecting pores and cracks for non-destructive testing in industrial CT. With a size-based transfer function, it is now possible to highlight sizes of interest.

One of the desired properties of a size-based transfer function is that size, analogous to density in a volume data set, should be continuous. This is a natural assumption, since features usually exhibit spatial coherence. For example, the color mapping used to classify the aneurysm in Fig.1(middle) shows subtle changes in the width of the vessels, which otherwise would be classified identically. With our approach, we can now see these small variations in size, while still visualizing global differences between large and small features.

This continuous representation of size is achieved with *scale fields*, which are scalar fields where every voxel represents the local scale or size of the feature containing that voxel. This idea contrasts with previous approaches to multi-scale analysis, which enhance classification or segmentation using a pyramid representation of a volume. Pyramids often result in discrete – and usually disperse – representation of scale, and they seldom offer the possibility to detect small variations in size. Instead, we derive a methodology for computing scale fields based on continuous scale-space theory and a set of scale detection filters. Scale-space theory was introduced by the computer vision community to analyze and process 2D images [15, 33], and has been widely used for diffusion-based smoothing of 3D volumes. In this paper, we show how diffusion mechanisms can be derived to improve scale selection, and how this information can be used to build a scale field. Unlike previous attempts for scale-space analysis of a volume, which were mostly applied as off-line operations due to its computational cost, our methodology allows the user to manipulate the parameters interactively, which proves to be a powerful mechanism for visualizing and exploring the complex structure of data volumes.

2 Related Work

Despite the fact that volume rendering has become a commodity, transfer function design is still a challenge. Many methods have been proposed to that end, which can be broadly classified as image-centric and data-centric [22]. Our work follows a data-centric approach, where a transfer function is derived by analyzing the volume data. The predominant approach derives a 1D transfer function based on scalar data values. Some have proposed higher-dimensional transfer functions based on first and second order derivatives of a volume, i.e., gradient information [14, 8] and curvature [6, 9]. These ideas have been extended to include rendering parameters, leading to lighting transfer functions [17] and illustration-inspired operators [23, 1]. Manual design of these high-dimensional transfer functions, however, remains a difficult task. To alleviate this problem, researchers have proposed better interaction techniques, e.g., widgets [10] and user painting [28], and semi-automatic methods that exploit additional properties of the data. For example, Roettger et al. proposed spatialized transfer functions [24], improving 2D histograms with spatial information. Fujishiro et al. [4] and Takahashi et al. [27] extract topological struc-

Carlos D. Correa and Kwan-Liu Ma are with the University of California, Davis, E-mail: {correac,ma}@cs.ucdavis.edu.

Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

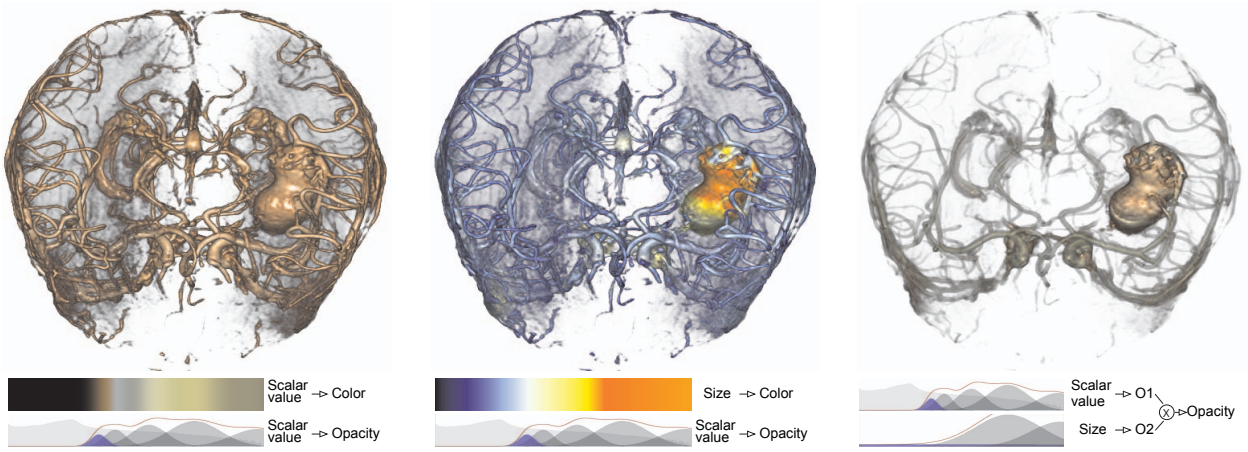


Fig. 1. Size-based classification of an aneurysm ($250 \times 250 \times 125$). (Left) 1D transfer function based on scalar value. (Middle) Size-based classification, where size maps to color and scalar value to opacity. (Right) Size-based classification, where opacity is the product of opacity mappings from both scalar value and size, in this case emphasizing larger features. Color mapping is same as (left).

tures to derive transfer functions, while Correa and Silver use curve-skeletons to define transfer functions along features [3]. Huang and Ma use partial region growing to facilitate the generation of transfer functions [7]. Enhancing transfer functions with computed or acquired values often results in an even more complex feature space. For this reason, we propose a new definition of transfer functions that maps the local size of features to color and opacity. Unlike other complex spaces, size can be defined in a single dimension and it complements easily traditional transfer functions.

Our methodology for extracting size is based on scale-space analysis. Scale-space theory is a framework for multi-scale analysis of images, developed by the image processing community [11, 15, 33]. At the core of this framework is the realization of a multi-scale representation of images that exhibits certain axiomatic principles about its uniqueness and invariant properties. The most common is the linear scale space, obtained with progressive Gaussian smoothing. Applications abound, including diffusion [30, 21], feature detection and automatic scale selection [16]. The study of scale-space in visualization has been limited, partly due to the increased computational complexity of 3D volumes. Previous alternatives to represent the scales of a volume used Laplacian pyramids [5] or Wavelet transforms [20, 32]. Westermann and Ertl [32] use a hierarchical multiscale representation to enhance structures such as edges and improve volume rendering. Vincken et al. [29] and Lum et al. [18] use pyramid representations to improve volume classification. These techniques refine the classification of a given voxel based on its behavior across the scales in the pyramid. Pyramid approaches, however, subsample the data in both space and scale. Here, we use continuous scale-space to derive high-quality *scale maps*, which are used to classify along a single dimension, *size*, orthogonal to scalar value. This property not only allows to define transfer functions based on size, but also improves classification. Continuous scale-space analysis has been used to detect and enhance vascular structures in medical images [19, 12], in segmentation [26], and shape detection [25]. In the latter, multiscale filters help detect features of varying shape, such as sheets, lines and blobs, and allows them to design transfer functions based on shape. In a similar fashion, we derive filters to detect size and use it as an extra dimension in transfer function design. In that sense, our approach is complementary to that of Sato et al [25]. In this paper, we propose a novel formulation of nonlinear diffusion that improves scale detection, and a mechanism to create a continuous representation of scale based on scattered data interpolation.

3 Continuous Scale Space

Continuous scale space is a framework developed by the computer vision community to analyze the multi-scale nature of data. Given a continuous N-dimensional signal $f : R^N \mapsto R$, its linear continuous scale space representation is a family of signals $L : R^N \times R_+ \mapsto R$ that satisfy the diffusion equation:

$$\partial_t L = \frac{1}{2}\nabla^2 L \tag{1}$$

with initial condition $L(\mathbf{x};0) = f(\mathbf{x})$. It has been shown that the solution to this equation is given by Gaussian smoothing of the signal with kernels of size t:

$$g(\mathbf{x};t) = \frac{1}{2\pi t} e^{\frac{-||\mathbf{x}||^2}{2t}} \tag{2}$$

The parameter t is therefore referred to as the *scale parameter*, and the linear scale-space representation of the signal can be computed via convolution:

$$L(\mathbf{x};t) = g(\mathbf{x},t) * f(\mathbf{x}) \tag{3}$$

This result comes from the assumption that Gaussian smoothing is the only filter that satisfies "reasonable" axioms for scale space. These axioms include linearity, isotropy, shift and scale invariance, and non-creation and non-enhancement of local extrema [15]. Some of these axioms have been relaxed, which has led to nonlinear and anisotropic scale spaces [30]. One of the most important axioms, which is the basis of scale detection, is that local extrema in scale space should not be created or enhanced as the scale parameter grows. In other words, new level surfaces are not created as we increase the scale parameter. In the following section, we describe how some of these axioms translate in desirable properties of our methodology for extracting local size.

3.1 Feature Detectors in Scale Space

Lindeberg noted that feature detectors can be obtained by looking at the behavior of the derivatives of the scale space along the scale dimension [16]. This has been widely exploited to extract edges, ridges, blobs and corners from 2D images. In our approach, we are interested in detecting blobs as Laplacian maxima in both scale and space. The scale at which the blob is detected, i.e., the t parameter, is the representative scale of the blob. This means that a Gaussian blob of width t, and centered around the detected point, can be used to "describe" both the size and location of the structure of the image at that location.

Scale-space representations are intrinsically different than pyramid representations. Pyramids are constructed by subsampling in both scale and space. Due to the the space subsampling, it is difficult to accurately locate Laplacian maxima for higher scales. Due to scale subsampling, maxima points can only detected at discrete scales, which are usually small compared to a uniform discretization of the scale-space. Fig.2 plots the response of the scale detection filter for two different points. On the left, the maximum is detected at scale 40.

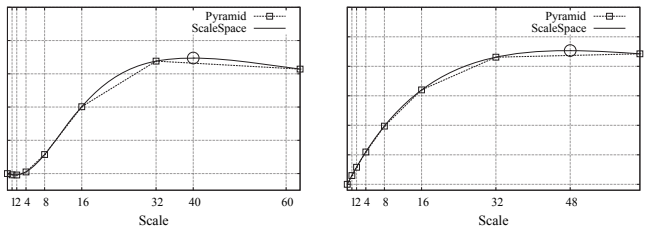


Fig. 2. Comparison of continuous scale-space vs. pyramids for scale selection. We plot the response of a scale detection filter for two points along multiple scales. Maximum points (circled) represent the intrinsic scale of the voxel. In the left, a pyramid representation (dashed line) incorrectly detects a maximum at scale 32, where it should be at scale 40. In the right, the pyramid results in a monotonically increasing curve and no maximum is detected, where it should be detected at scale 48.

However, if we use a pyramid (dashed line), the maximum appears to be at scale 32, due to the poor sampling of higher scales. On the right, the maximum detected at scale 48 is not detected at all using a pyramid, which seems to grow monotonically. A continuous approximation can be built from pyramids, assuming a reconstruction model, at the cost of computational complexity.

Scale-space representations, on the contrary, can be tuned to detect scales at varying degrees of granularity. This gives us the possibility to create scale fields with varying degrees of precision, as described in the following sections. Later on, we compare the result of our approach based on continuous scale-space vs. a pyramid representation.

4 Scale Field

The first step towards applying size-based transfer functions is the estimation of a 3D scalar field to encode size information, called the *scale field*. The scale field is a mapping $S : R^N \mapsto R$, such that $S(\mathbf{x}) = t$ represents the local scale of the feature containing the point $\mathbf{x}$. We can define this field more formally as the fitting of a continuous function S to a set of points $\{\mathbf{x}_i;t_i\}$, such that $S(\mathbf{x}_i) = t_i$, and it exhibits certain degree of continuity, where $\{\mathbf{x}_i;t_i\} \subseteq L$ represent the most salient points in scale-space, according to a given detection filter. The process of computing scale fields can then be decomposed into several steps, as depicted in Fig.3:

Scale-space computation, which generates a multi-scale representation of the 3D image, given an interval of "interesting scales" $[t_{min}, t_{max}]$.

Scale detection, which outputs a set of points $\{\mathbf{x}_i;t_i\}$, which we refer to as the *scale abstraction* of the 3D image, and finally

Backprojection, which fits a continuous function – the scale field – to the scale abstraction.

These steps are described in the following section. Because we are interested in generating scale fields for the visualization of discrete 3D images, we derive its computation for the discrete case. Therefore, we use the terms *scale field* and *discrete scale field* (which is a discretization of a *continuous* scale field) interchangeably, unless specifically noted. We assume that the discretization occurs in a regular grid.

4.1 Scale-space Computation

Instead of costly convolution, we compute the scale space of a data set iteratively, using forward Euler integration of the diffusion equation (Eq. 1). Using a discretization stencil around a given point (typically a 6-, 18- or 26-point neighborhood), we compute the representation at a scale $t + \Delta t$ iteratively,

$$L(\mathbf{x};t+\Delta t) = L(\mathbf{x};t) + \Delta t \sum_{ijk \in N} \alpha_{ijk} \Phi(L(\mathbf{x}_{ijk};t) - L(\mathbf{x};t)) \quad (4)$$

where N is a set of indices that point to the neighbors of a given point $\mathbf{x}$, α_{ijk} are normalizing coefficients, and $L(\mathbf{x};0)$ is the original image. Φ is a flow function,

$$\Phi(x) = x \cdot C(x) \quad (5)$$

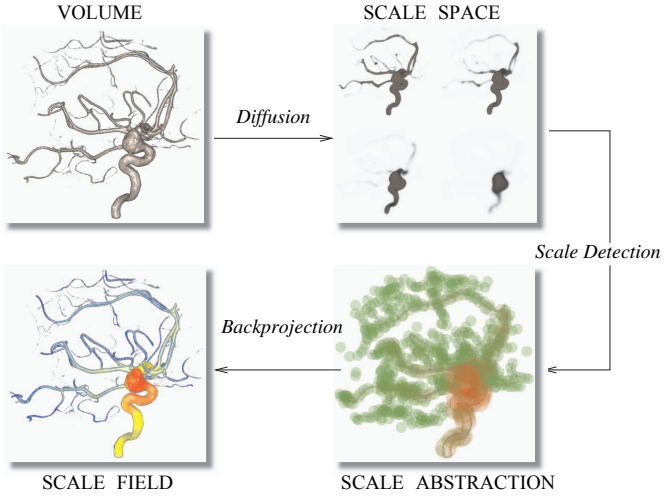


Fig. 3. Overview of the process of obtaining scale fields.

where C is a conductivity function. For the linear case, the conductivity is constant and it is equal to 1. Perona and Malik introduced a non-linear conductivity function based on gradient magnitude, which preserves edges [21],

$$C(||\nabla L||) = \frac{\lambda^2}{\lambda^2 + ||\nabla L||^2} \quad (6)$$

where λ is a sensitivity parameter. As λ grows, C approaches 1 and the result is linear diffusion. This function favors large regions with certain degree of homogeneity and preserves edges.

When we consider variable conductivity for the diffusion problem, it becomes equivalent to solving the non-linear diffusion equation,

$$\partial_t L = \frac{1}{2} \nabla \cdot C(\mathbf{x}) \nabla L \quad (7)$$

This problem, however, is known to be ill-posed in the continuous case and requires regularization factors [2]. Weickert also noted that, in the discrete case, a finite difference discretization acts as a regularizer and stable results can be obtained for certain values of Δt (typically ≤ 1). Alternatively, Weickert suggests to define conductivity as a tensor instead of a scalar [30]. We derive our own variation of non-linear conductivity that improves scale detection, as described in the following section.

The user can control the diffusion process with two parameters: the number of iterations n of Eq.4 and the scale sampling distance Δt. These parameters indicate the minimum and maximum sizes that can be detected. One can automate the selection of the parameters, assuming the user wants to detect all possible sizes, by setting $n = Dim/\Delta t$, where Dim is the dimensions of the volume. Setting $\Delta t = 1$ defines the minimum detectable size as the size of a voxel. Sub-voxel sizes can be detected by setting $\Delta t < 1$.

4.2 Scale Detection

A number of detection filters can be obtained from the scale-space of a volume by looking at the first and second derivatives. In particular, blobs can be detected at multiple scales whenever the normalized second derivative $\nabla^2 L$ assumes a local maxima. This is obtained as the normalized Laplacian,

$$t\nabla^2 L = t(L_{xx} + L_{yy} + L_{zz}) = tTr(H(\mathbf{x})) \quad (8)$$

where $H(\mathbf{x})$ is the Hessian matrix at a point $\mathbf{x}$. The set of these points which are maxima in both space and scale – here denoted as Laplacian of Gaussian (LoG) extrema – together with the scale parameter t at which they were found, constitute the *scale abstraction* of the data set. Fig.3 shows the scale abstraction for an aneurysm data set where spheres are used to indicate the location and size of the detected blobs.

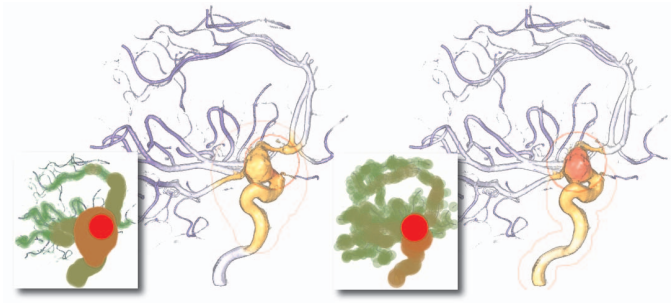


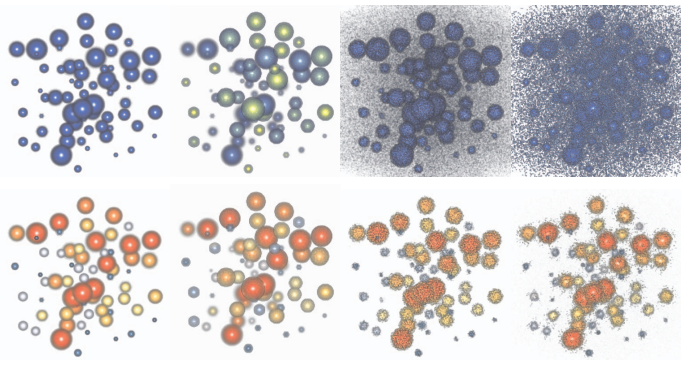
Fig. 4. Pyramid vs. Scale-space. On the left, scale maps constructed from a pyramid exhibit poor granularity of the scales and poor localization. Note the discrepancy between the largest scale (red circle) and the aneurism. On the right, scale map based on our approach leads to a continuous representation of scale with better localization. The aneurism is correctly detected as the largest feature.

Using linear diffusion to detect scales may result in two or more nearby objects merging in a single blob before they are detected as individual features. In such cases, the scale maxima are not localized accurately. For this reason, we seek to develop a filter that favors diffusion in regions of a given homogeneity, but prevents diffusion over edges. The original Perona-Malik filter does not seem of interest since diffusion stops at the edges. This generates a lot of false maxima responses that need to be pruned. Instead, we want to control the direction of diffusion, so that it only crosses boundaries in the direction towards the medial axis of a feature. In our discretization stencil, this can be achieved by considering the signed forward differences as an approximation of the gradient, instead of using the unsigned gradient magnitude. Assuming the discretization in Eq.(4), we construct a different conductivity function,

$$C(L(\mathbf{x}_{ijk})-L(\mathbf{x})) = \begin{cases} \frac{\lambda^2}{\lambda^2+(L(\mathbf{x}_{ijk})-L(\mathbf{x}))^2} & L(\mathbf{x}_{ijk})-L(\mathbf{x})>0 \\ 1 & \text{otherwise} \end{cases} \quad (9)$$

where λ is a sensitivity parameter controlled by the user to define possible boundaries between features. Small values ($\lambda \in (0.01, 0.1)$) lead to better localized scale maxima (useful for convoluted structures), while large values ($\lambda > 1$) makes diffusion linear. Although it results in slight deviations of the scale maxima, it is in general more robust to noise. The effect of our filter is similar to that of progressively applying smoothing and *erosion* filters.

Previous approaches based on pyramids follow the linear diffusion formulation, with the additional subsampling in both scale and space. Subsampling in scale results in a limited number of detected sizes, which underestimate or overestimate the representative scales of features. Subsampling in space compounds the localization problem. Figure 4 shows a comparison of an SBTF applied to an aneurism data set. On the left, the scale abstraction is computed via a Laplacian pyramid. The scale abstraction is depicted on the lower left corner as a series of spheres. When we superimpose the largest spheres onto the original image, we see that this method greatly mislocates the largest feature. In addition, the granularity of the detected scales does not capture the subtle variations of the different features. This result is representative of previous multi-scale classification methods, such as the one by Lum et al. [18]. On the right, we show the results of our approach. Note that the distribution of scales is smoother and they are correctly localized in space. Now the aneurism, highlighted in red, is clearly extracted from the nearby structures. One can improve the results of the pyramid by adding interpolation between different levels. To avoid the problems described in Fig.2, one can approximate the scale-space with higher order interpolation, as suggested by Kothe [13]. This will improve the granularity of the scales. Localization, however, may not be improved, since it is the product of our nonlinear diffusion filter. In that case, one could construct an anisotropic diffusion pyramid, and this is the focus of our future research.



(a) Original (b) Intensity (c) Gaussian (d) S&P

Fig. 5. Robustness to intensity variation and noise. (a) Original data set consisting of spheres of random size, (b) intensity variation (notice the change in opacity and color), (c) Gaussian and (d) Salt-and-pepper noise. Bottom: Size-based classification. Color is mapped to size so that blue and white indicate smallest features, while yellow and red larger sizes. Classification is robust to both intensity variation and noise. Modulating the opacity based on size, so that smallest features are more transparent, results in noise suppression.

4.3 Backprojection

The previous stage outputs a set of discrete points that represent the most salient scales of the volume. For a continuous and smooth transfer function, we need to fit a continuous representation. One option is to define a local descriptor as the maximum response of the normalized LoG at every single voxel. This, however, does not result in an intuitive representation of size, since this function decreases for voxels away from the LoG extrema. Instead, it is reasonable to assume that all voxels within a radius t of a LoG extrema $(\mathbf{x};t)$, can be described with a size t.

Because these blobs may intersect, we must fit a scale so that it exhibits certain degree of continuity. This can be accomplished with scattered data interpolation. A fast method can be obtained with Shepard's interpolation. Given a scattered set of LoG extrema $N = \{(\mathbf{x}_i;t_i)\}$, the scale field S can be obtained as

$$S(\mathbf{x}) = \sum_{\mathbf{x}_i \in N} \Theta(||\mathbf{x}-\mathbf{x}_\mathbf{i}||)t_i \quad (10)$$

where $\Theta(d)$ is a basis function. Common basis functions are Gaussian kernels and Wendland polynomials [31]. In our approach, we use the fourth degree Wendland polynomial, due to its compact support:

$$\Theta(d) = \left([1-d/h]_0^1\right)^4 (4d/h+1) \quad (11)$$

where $[\cdot]_0^1$ is a clamping function between 0 and 1, and h is a parameter that controls the local support of the basis function. As h increases, more overlap among kernels is obtained, which increases the smoothness of the field, at the cost of less distinction between the different scales. One can set $h = kt_i$, i.e., proportional to the representative scale, where k is usually 1. The user may adjust above and below this value to explore scale fields of varying smoothness. Alternatively, one can blend the scale blobs with the *maximum* operator, instead of a sum, in which case the resulting scale field is given by $S(\mathbf{x}) = max_{\mathbf{x}_i \in N}\{\Theta(||\mathbf{x}-\mathbf{x}_i)t_i\}$. This is useful when we must give priority to larger features and avoid smoothing away the scale variation. Our results in Fig.7 and 8 use this operator. In Section 6, we show that this method can be implemented in the GPU, using hardware-supported blending methods. Because this interpolation is based entirely on the scale abstraction, bleeding of a scale into a small feature may happen. This occurs when a small feature gets embedded in the blob surrounding a large feature. In this case, we can improve the construction of the scale field with a visibility test. The visibility test is used to modulate the weight of the LoG extrema at a given point. If a sample point is not "visible" from the LoG extrema, then its weight Θ

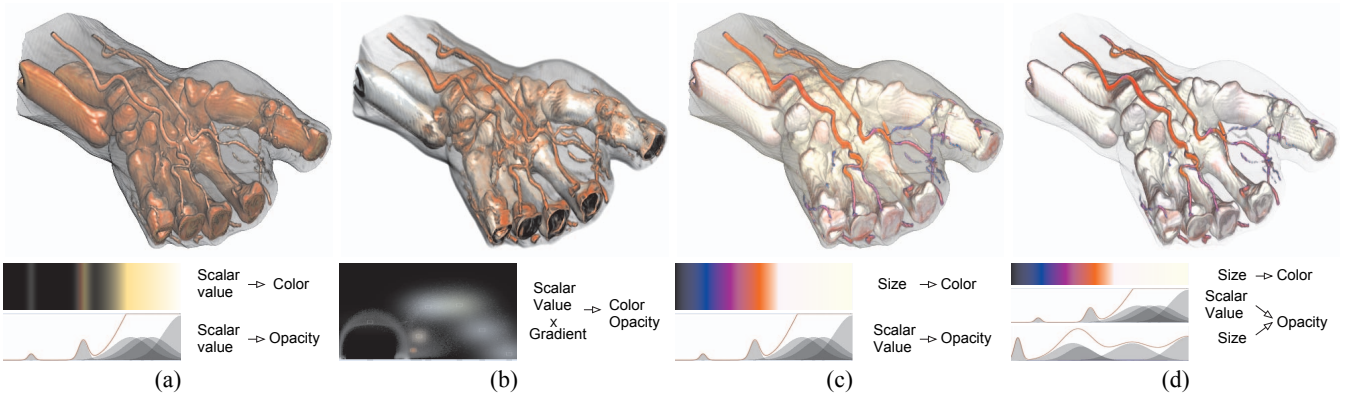


Fig. 6. Size-based classification of an unsegmented hand data set. (a) Traditional transfer functions make it difficult to emphasize the vessels as opposed to bones, where the difference in density is little (as is the case with many contrast-enhanced data sets). (b) 2D transfer functions only improve classification marginally (c) With a size-based transfer functions, the user can now assign colors based on size. In this case, the opacity mapping is maintained from (a). (d) By using size to map color and also opacity, it is now possible to highlight certain parts, e.g., veins and small vessels, while hiding other parts, e.g., skin. Note that results are comparable to those of rendering a segmented data set.

becomes 0. A visibility parameter τ is used to control the sensitivity of this test. This parameter is essentially a thresholding value, and can be based on scalar value or gradient magnitude. A voxel is not visible from the LoG extrema if at some point in the ray connecting the two positions the intensity is below a threshold. Alternatively, one can threshold based on gradient magnitude. An example is shown in Fig.7. The scale field allows us to classify the brain tissue together with the neighboring vessels. With a visibility threshold, we prevent the scale to be backprojected to those voxels that correspond to vessels, giving a better classification of the brain. It must be noted that this is a fast approximation that works well for scale fields, and that more accurate results can be obtained with a region growing approach. This can be added into our methodology as a post-process based on our initial size-based classification.

4.4 Properties

The use of diffusion-based scale-space has interesting properties, which prove important for visualization.

4.4.1 Robustness to Noise

Because this method is based on diffusion, it is inherently robust to noise. To illustrate this, Fig.5(c-d) shows a series of synthetic data sets consisting of randomly sized spheres in a volumetric domain with Gaussian and salt-and-pepper (S&P) noise. In the bottom, we show the classification of the spheres based on their size (red is the largest). We can see that the classification does not suffer from the levels of noise (compare to Fig.5(a)). Furthermore, noise is represented in the smallest scales of the scale field for both Gaussian and S&P noise. A size-based transfer function that vanishes for small values can be used to suppress the noisy voxels, as shown in Fig.5.

4.4.2 Invariance to Intensity

Another interesting property is its invariance to intensity changes. Detection of extrema in the Laplacian is independent of the local intensity contrast. This means that features of different density but similar size will be represented similarly in the scale field, as shown in Fig. 5(b). This is important for size-based classification, as it provides orthogonality. Now, it is possible to create 2D transfer functions with scalar-value on one dimension and scale in the other.

4.4.3 Progressive Construction

This is a consequence of two of the scale-space axioms, scale invariance and non-enhancement of local extrema. These state that the operations that compute scale space are used in the same way for any given scale, and also, that the transformation from the original image to any scale is the same from a finer scale to any coarser scale. This implies that the scale-space can be constructed progressively, and intermediate results are valid approximations. That is, the scale field at a given scale is the same whether it is obtained from the original volume, or from an already smoothed version of the data set. This also has an important implication for interactivity, since large data sets ($> 512^3$) do not need to be analyzed at their full resolution. The scale field can be of a lower resolution, and the resulting scales (except for the granularity of which they are obtained) *are the same*.

5 Size-based Classification

With a scale field, we have obtained the necessary information to apply a size-based transfer function. An SBTF can be defined as a mapping $S \mapsto O$, where S is the scale field, and O is a set of optical properties, typically a four dimensional tuple containing color and opacity attributes. In general, we want to combine the scale field with the original scalar data values to improve classification. The resulting transfer function is then obtained from a Cartesian product of fields, $S \times F_1 \times ... \times F_n \mapsto O$, where $F_1, \ldots, F_n$ are scalar fields. In most of our examples, the opacity is both the product of size S and the original scalar values F.

Fig.1 shows a visualization of an aneurysm. Fig.1(left) and (middle) use the same opacity values, but in Fig.1 (middle), color values are obtained from an SBTF. The global differences in size become immediately apparent. On further inspection of the image, subtle variations on the width of the vessels also become apparent. In Fig.1(right), the SBTF is used also to define opacity, enabling the user to highlight automatically large features.

Fig.6 shows the classification of an unsegmented CT scan of a human hand. With 1D transfer functions, it becomes difficult to separate voxels belonging to vessels and those in bone tissue. A 2D transfer function, although improves classification, cannot properly separate the two. The difference, however, can be resolved via an SBTF. In Fig.6(c), we map size to colors, while scalar value is mapped to opacity (the same transfer function as in Fig.6(a). We can see a considerable improvement in classification. Further, small vessels are also classified based on their width. In Fig.6(d), opacity is mapped as the product of two opacity functions, based on scalar value and size, respectively. Now the skin tissue can be de-emphasized, while vessels and bone are emphasized.

Fig.9(a) shows an MR angiogram of a human head. Fig.9(b) shows the size-based classification of the same data set. When analyzing vascular data sets, maximum intensity projection (MIP) (Fig.9(c)) is often more effective than direct volume rendering. One of the problems is the difficulty to emphasize structures of interest, due to the accumulation of intensity. With an SBTF, we can map opacity directly to the rendered samples, to obtain a size-adapted MIP rendering of the an-

giogram. Figures 9(d) and (e) show a rendering where we highlight and suppress the large features, respectively.

5.1 Applications to Volume Rendering and Exploration

Similar to previous multiscale approaches, an SBTF helps classify and segment complex data sets. Fig.7 shows an MRI of a brain. Classifying the brain is difficult, since other occluding tissue, such as skin, is represented in the same density interval. However, when applying an SBTF, skin and skull tissue appear as small features, since they are relatively thin in comparison to brain tissue. By setting high opacity to the large features, now we can clearly separate the brain tissue, as shown in Fig.7(b). Note that some vessels and surrounding tissue is part of this large feature. In Fig.7(c), we set the visibility parameter to prune voxels that do not correspond to brain tissue, which results in a clear view of the brain. The rendering results are comparable to lengthy segmentation, but at higher speeds. Furthermore, the scale abstraction already provides a set of points that can be used as seeds for segmentation. Fig.8 shows a classification of an MRI knee, a noisy complex data set. Transfer functions based on scalar value cannot separate muscle, skin and bone properly. With an SBTF, we can now clearly classify the different tissues. Note that even bones of different size are assigned slightly different colors. The spatial relationship between the different overlapping structures is now clear.

Fig.11 shows a CT scan of multiple objects in a backpack. The difficulty to isolate features is lessened with an SBTF. This data set contains a number of objects of varying size, which can be identified now based on size. However, as we explore the entire scale-space, smaller sizes cannot be distinguished. For this reason, we can apply SBTF selectively, by considering sub-regions of the data. Fig.11 shows three stages of zooming into the data. At each stage, a new SBTF reveals a higher granularity of sizes. Towards the right, we can detect and highlight the differences in size of the small parts of an object, impossible to detect at the full resolution. Compare to the image on the right, where classification based solely on scalar value does not help isolate each part.

6 GPU Implementation

Our GPU implementation uses widely available features such as programmable shaders and framebuffer objects. The different stages described in section 4 are implemented using pixel shaders over multiple passes. In previous multi-scale approaches, a Laplacian or Gaussian pyramid is stored in GPU memory. However, storing a full scale-space in GPU memory is prohibitive. For example, a 256^3 volume would need 1GB of memory to store up to 64 scales at 8-bit resolution, or up to 16 at 32-bit resolution. For this reason, we compute the scale fields on a per-iteration basis, which only requires one temporary volume at a time. Therefore, the stages of scale-space computation and scale detection are merged into a single pass:

Scale detection: We solve the diffusion equation by implementing Eq. 4 in a pixel shader. The shader uses as an input a slice of the 3D texture and the output is written also to a 3D texture slice, which is subsequently used in the next iteration. To avoid simultaneous reads and writes, we use a ping-pong approach. To speed up the process of scale detection, we compute the second derivatives in the same pass. We also keep the derivatives of the previous two steps in order to find scale-space maxima. We encode this so that we use the four channels of a pixel $\left[R : L(\mathbf{x};t+1), G : \nabla^2 L(\mathbf{x};t+1), B : \nabla^2 L(\mathbf{x};t), A : \nabla^2 L(\mathbf{x};t-1)\right]$, where t is increased at each iteration. The scale maxima is then found by writing out a pixel when the value of the Laplacian at a given iteration (here stored in the blue channel) is the maximum of the Laplacian of the neighbors in iterations $t-1$, t and $t+1$ (here stored in the GBA channels, respectively, for each iteration). As we increment our iterations, those pixels denoting scale maxima are read back to the CPU and stored as the scale abstraction.

Backprojection: To implement backprojection, we generate a series of small slices for each scale-space blob, with sides proportional to their radii. Each pixel generated by these slices compute the weighted contribution of the blob. We then use blending to implement the sum. In another pass, we normalize the result by dividing by the sum of weights, which is stored in a different channel . To evaluate our approach, we implemented a comparable CPU version and obtained timing results. We used an Intel Core 2 Duo 2.4 GHz with 2GB of RAM and an nVidia GeForce 8800 GTX with 768 MB of texture memory. Figs. 10(a) and 10(b) show the timing for the scale-space computation and backprojection stages, respectively, in seconds, for three data set sizes (64^3, 128^3 and 256^3). We can see an improvement of about two orders of magnitude in both cases. A CPU implementation then proves to be impractical for interactive visualization. In Fig.10(c) a stacked line diagram compares the timing between the two stages in the GPU. Since most of the cost is required to compute the scale abstraction, we can decouple this stage from backprojection. Scale-space computation is done once, and re-computed only as the user changes the diffusion parameters, while backprojection can be done more frequently, as the user changes the smoothness and visibility parameters.

6.1 Limitations and Future Work

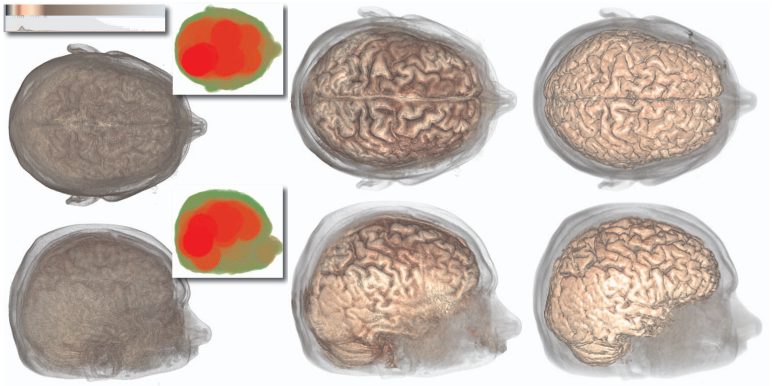
One of the limitations of our approach is the reliance of a temporary volume to compute the scale field. As GPUs improve, we believe that parts of our methodology will be implemented in real-time and embedded in the volume rendering process. However, since interesting sizes can be detected with a lower resolution scale field, we believe that our approach is still valid for larger data sets. Although a 512^3 full precision scale field cannot be accommodated in current GPU hardware, we can still create an interesting scale field at a lower resolution, restricting the detectable sizes to the resolution of the scale field. If we subsample by a factor of 4, features of sizes from 1 to 4 voxels wide cannot be discerned (they will be collectively detected as the smallest size). However, detection of sub-voxel sizes can be obtained by analyzing sub-regions at a time (so that each of them can fully fit in texture memory), as shown in Fig.11. We believe that this is a practical solution to limited texture memory, and encourages multi-scale exploration of large data sets.

One of the aspects of our approach, being based in diffusion, is figure/ground separation. Our approach works best when features can be clearly separated from the background. In other cases, such as in flow simulation, density may vary smoothly across the entire domain. In this case, similar to other classification approaches, a thresholding is needed to clearly define the features of interest. Similarly, detecting holes and cracks in industrial CT implies the classification of background as opposed to figure voxels. Since diffusion tends to move from high intensity to low intensity values, holes will not be detected. In that case, a figure/ground reversal solves the problem.

Another aspect of our approach is that our diffusion filter is still isotropic. This means that features are *tagged* according to the smallest size of their local structure. For large narrow structures, the inherent scale is the width rather than the length. Although this has proved to be a very useful descriptor, this can be extended to multiple types of structures to reveal different sizes according to the shape of a feature, e.g., planar, tubular or blob-like. This can be accomplished with locally adapted filters and it is currently ongoing research. This may become a complement to shape-based classification approaches such as the one introduced by Sato et al. [25].

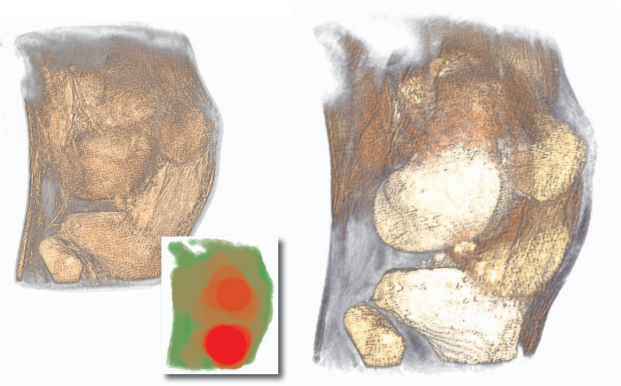
7 Conclusion

We have introduced the concept of *size-based transfer functions*, which maps the relative size of local features in a volume to color and opacity. Now it is possible to visualize complex data sets such that features can be classified based on their relative size. Therefore, small features can be clearly separated from large ones, especially important when these features have similar scalar value. Size-based transfer functions can be achieved with the use of *scale fields*, which are 3D fields where every voxel represents the representative scale at that point. We compute these scale fields via scale-space analysis and a set of detection filters. While being prohibitive in the CPU, our GPU implementation enables interactive exploration and can be easily deployed in visualization systems. Through a number of examples,



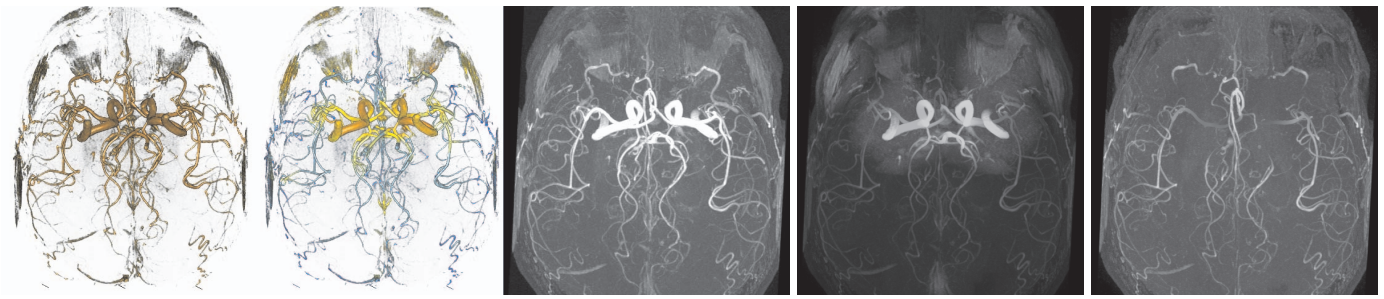
(a) (b) (c)

Fig. 7. Volume classification of the MRI brain data set ($256 \times 256 \times 156$) (a) Direct volume rendering (DVR) using a 1D transfer function, and detected scales (b) Size-based classification. Only large regions are highlighted, which correspond to brain tissue ($\tau = 0$, i.e. no visibility test is performed). Note how the brain is clearly seen, together with some small vessels. (c) With the visibility test ($\tau = 0.3$), we obtain a better classification of the brain tissue. Note that small veins have been removed.



(a) (b)

Fig. 8. Volume classification of an MRI knee data set ($512 \times 512 \times 87$) (a) Original data set with a 1D transfer function, and detected scales. (b) Size-based classification. Note that bones, muscle and skin can now be separated. Furthermore, individual bones van be classified differently due to their difference in size.



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

Fig. 9. Size-adapted visualization of angiograms ($512 \times 512 \times 128$). (a) Original DVR of angiogram (b) DVR of angiogram with a size-based transfer function (c) MIP rendering of angiogram (d) Size-adapted MIP, where only the large features are highlighted (e) Size-adapted MIP, where the large features are suppressed.

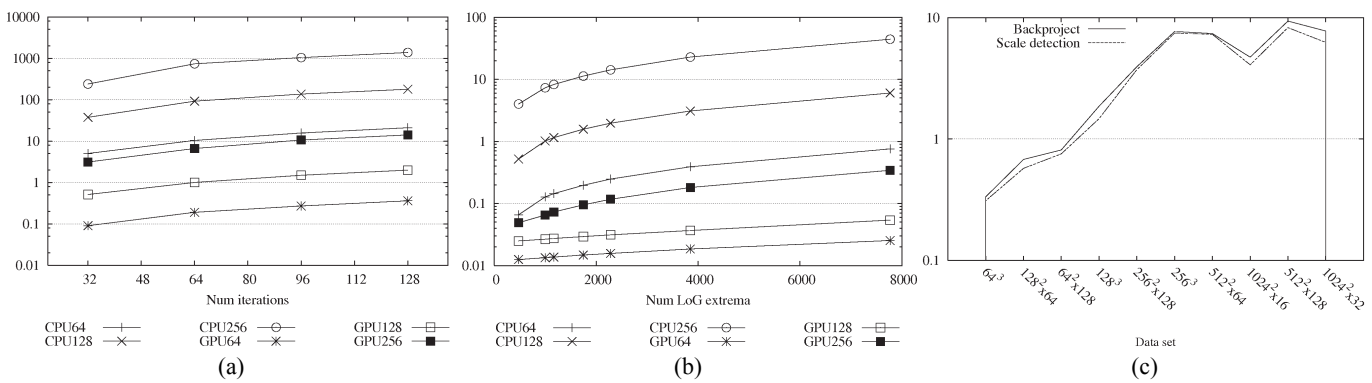


Fig. 10. (a) Timing of scale-space computation and detection for CPU vs. GPU implementations, for three data sets (64^3, 128^3 and 256^3) in seconds (log scale). (b) Timing of the backprojection stage. Note that the GPU implementation is two orders of magnitude faster. The CPU implementation is prohibitive for interactive exploration, while the GPU implementation approaches real-time for the smallest data sets. (c) Timing comparison for scale-space computation and backprojection using a stacked line diagram. Backprojection is considerably faster compared to scale-space computation, which enables us to decouple the stages of our methodology.

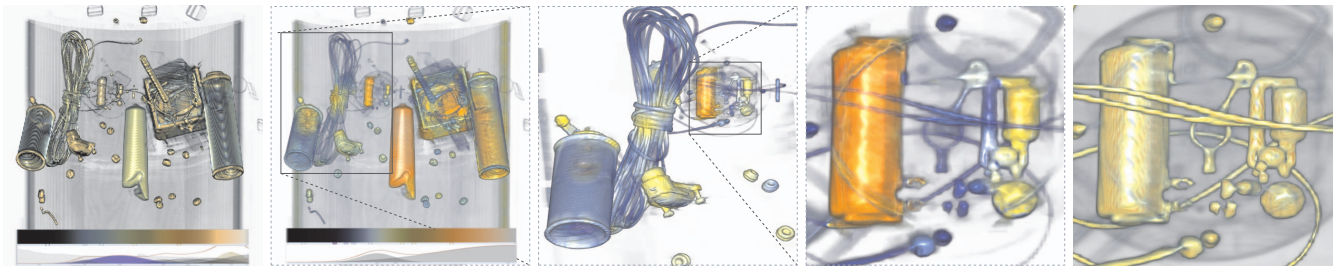


Fig. 11. Volume rendering of the backpack data set ($512 \times 512 \times 373$). From left to right: (1) Traditional transfer function based on data values (2) SBTF on the entire data set. Note that differences in size become immediately apparent. (3) Selective SBTF on a small region. At this scale, we can now visualize more sizes of interest. (4) In a smaller region, we can find differences in size undetectable from the entire dataset. In this case, we are able to decompose the parts of an object based on size. Compare to classification based on scalar value on the right.

we have shown that classification of complex data sets is made easier. Rendering of vascular data sets, such as MRIs of aneurysms, can now present to the user the subtle variations between different vessel sizes. Size-based transfer functions also improve the way occlusion is handled in volume rendering, and we have shown a number of examples where our approach achieves results comparable to those of time-consuming segmentation. Furthermore, our approach readily provides an abstraction, which can be used as a seed to more sophisticated segmentation algorithms. Size-based transfer functions provide a novel exploration technique that can be extended in a number of ways towards a more intuitive visualization of complex data sets.

Acknowledgements

This research was supported in part by the National Science Foundation through grants CCF-0634913, CNS- 0551727, OCI-0325934, OCI-0749227, and OCI-0749217, and the Department of Energy through the SciDAC program with Agreement No. DE-FC02-06ER25777, DE-FG02-08ER54956, and DE-FG02-05ER54817. Data sets are courtesy of the University of Utah, Viatronix Inc., Tiani Medgraph, Philips Research and The Institute for Neuroradiology, Frankfurt.

References

[1] S. Bruckner and M. E. Gröller. Style transfer functions for illustrative volume rendering. *Computer Graphics Forum*, 26(3):715–724, 2007.

[2] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J. Numer. Anal.*, 29(1):182–193, 1992.

[3] C. D. Correa and D. Silver. Dataset traversal with motion-controlled transfer functions. In *IEEE Visualization 2005*, pages 359 – 366.

[4] I. Fujishiro, T. Azuma, and Y. Takeshima. Automating transfer function design for comprehensible volume rendering based on 3d field topology analysis. In *IEEE Visualization*, pages 467–470, 1999.

[5] M. H. Ghavamnia and X. D. Yang. Direct rendering of laplacian pyramid compressed volume data. In *Proc. IEEE Visualization 1995*, page 192, 1995.

[6] J. Hladůvka, A. König, and E. Gröller. Curvature-based transfer functions for direct volume rendering. In *Spring Conference on Computer Graphics 2000 (SCCG 2000)*, volume 16, pages 58–65, 2000.

[7] R. Huang and K.-L. Ma. Rgvis: Region growing based techniques for volume visualization. In *PG '03: Proc. Pacific Conference on Computer Graphics and Applications*, page 355, 2003.

[8] G. Kindlmann and J. W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *VVS '98: Proceedings of the 1998 IEEE symposium on Volume visualization*, pages 79–86, 1998.

[9] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Moller. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *Proc. IEEE Visualization 2003*, 2003.

[10] J. Kniss, G. Kindlmann, and C. Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proc. IEEE Visualization 2001*, pages 255–262, 2001.

[11] J. J. Koenderink. The structure of images. *Biological Cybernetics*, 50(5):363–370, 1984.

[12] T. M. Koller, G. Gerig, G. Szekely, and D. Dettwiler. Multiscale detection of curvilinear structures in 2-d and 3-d image data. In *ICCV '95: Proc. of the Fifth International Conference on Computer Vision*, page 864, 1995.

[13] U. Kothe. Accurate and efficient approximation of the continuous gaussian scale-space.

[14] M. Levoy. Display of surfaces from volume data. *IEEE Comput. Graph. Appl.*, 8(3):29–37, 1988.

[15] T. Lindeberg. Scale-space for discrete signals. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(3):234–254, 1990.

[16] T. Lindeberg. Feature detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):79–116, 1998.

[17] E. B. Lum and K.-L. Ma. Lighting transfer functions using gradient aligned sampling. In *Proc. IEEE Visualization '04*, pages 289–296, 2004.

[18] E. B. Lum, J. Shearer, and K.-L. Ma. Interactive multi-scale exploration for volume classification. *The Visual Computer*, 22(9-11):622–630, 2006.

[19] R. Manniesing, M. A. Viergever, and W. Niessen. Vessel enhancing diffusion: A scale space representation of vessel structures. *Medical Image Analysis*, 10(6):815–825, 2006.

[20] S. Muraki. Volume data and wavelet transforms. *IEEE Comput. Graph. Appl.*, 13(4):50–56, 1993.

[21] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(7):629–639, 1990.

[22] H. Pfister, B. Lorensen, C. Bajaj, G. Kindlmann, W. Schroeder, L. S. Avila, K. Martin, R. Machiraju, and J. Lee. The transfer function bake-off. *IEEE Comput. Graph. Appl.*, 21(3):16–22, 2001.

[23] P. Rheingans and D. Ebert. Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Trans. on Visualization and Computer Graphics*, 7(3):253–264, 2001.

[24] S. Roettger, M. Bauer, and M. Stamminger. Spatialized transfer functions. In *EuroVis*, pages 271–278, 2005.

[25] Y. Sato, C.-F. Westin, A. Bhalerao, S. Nakajima, N. Shiraga, S. Tamura, and R. Kikinis. Tissue classification based on 3d local intensity structure for volume rendering. *IEEE Trans on Visualization and Computer Graphics*, 6(2):160–180, 2000.

[26] A. Sherbondy, M. Houston, and S. Napel. Fast volume segmentation with simultaneous visualization using programmable graphics hardware. In *Proc. IEEE Visualization 2003*, page 23, 2003.

[27] S. Takahashi, Y. Takeshima, and I. Fujishiro. Topological volume skeletonization and its application to transfer function design. *Graph. Models*, 66(1):24–49, 2004.

[28] F.-Y. Tzeng, E. B. Lum, and K.-L. Ma. A novel interface for higher-dimensional classification of volume data. In *Proc. IEEE Visualization 2003*, page 66, 2003.

[29] K. L. Vincken, A. S. E. Koster, and M. A. Viergewer. Probabilistic hyperstack segmentation of mr brain data. In *CVRMed '95: Computer Vision, Virtual Reality and Robotics in Medicine*, pages 351–357, 1995.

[30] J. Weickert. A review of nonlinear diffusion filtering. In *SCALE-SPACE '97: Scale-Space Theory in Computer Vision*, pages 3–28, 1997.

[31] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial basis functions of minimal degree. *Advances in Computational Mathematics*, 4:389–396, 1995.

[32] R. Westermann and T. Ertl. A multiscale approach to integrated volume segmentation and rendering. *Comput. Graph. Forum*, 16(3):117–128, 1997.

[33] A. P. Witkin. Scale-space filtering. In *IJCAI*, pages 1019–1022, 1983.