Illustration-Inspired Depth Enhanced Volumetric Medical Visualization

Nikolai A. Svakhine, David S. Ebert, and William M. Andrews

Abstract—Volume illustration can be used to provide insight into source data from CT/MRI scanners in much the same way as medical illustration depicts the important details of anatomical structures. As such, proven techniques used in medical illustration should be transferable to volume illustration, providing scientists with new tools to visualize their data. In recent years, a number of techniques have been developed to enhance the rendering pipeline and create illustrative effects similar to the ones found in medical textbooks and surgery manuals. Such effects usually highlight important features of the subject while subjugating its context and providing depth cues for correct perception. Inspired by traditional visual and line-drawing techniques found in medical illustration, we have developed a collection of fast algorithms for more effective emphasis/deemphasis of data, as well as conveyance of spatial relationships. Our techniques utilize effective outlining techniques and selective depth enhancement to provide perceptual cues of object importance, as well as spatial relationships in volumetric data sets. Moreover, we have used illustration principles to effectively combine and adapt basic techniques so that they work together to provide consistent visual information and a uniform style.

Index Terms—Interactive volume illustration, volume rendering, nonphotorealistic rendering, illustrative techniques.

1 Introduction

Recently, the fusion of volume rendering with nonphotorealistic rendering (NPR) has become an active research area called volume illustration. The goal of volume illustration research is to harness the power of traditional illustration techniques to more effectively convey information from volume data sets.

Historically, illustration has employed a wide range of drawing techniques and media to create effective representations of nature. Contrary to the use of image acquisition devices (e.g., a camera), an illustrator makes choices, whether conscious or not, about both *where and what* to draw and *how* best to draw it. These choices allow emphasis/deemphasis and directed focus and form the foundation for the contextual communication that is the ultimate purpose of an illustration.

In examining traditional illustrations, a few basic principles emerge. First, for creating believable representations of three-dimensional forms, the lighting/shading of the gross form or volume of an object is more important than the depiction of subordinate forms; second, the lighting/shading of subordinate forms is more important than the depiction of surface details or texture. For example, in Fig. 1c, the gross form of the head/brain is more important than depicting the subordinate forms of the eyes and depicting the forms of the subordinate eyes is in turn more important than depicting the fine detail of the eyelashes or the skin texture. Working within this fundamental framework, the illustrator may emphasize or subjugate features to better direct the viewer's focus.

We have utilized these illustration principles for conveying relationships to extend previous work in illustrative visualization and provide new techniques to more effectively support traditional illustrative outlining and depth perception techniques within illustrative visualization. We have also harnessed the ability to recognize and manipulate image gradients as an essential tool within traditional illustrations. Gradients of contrast, line weight, periodicity, density, and detail are used in illustrations to provide the viewer with a sense of looking into a volume of space.

We have also developed globally aware parameterizations of the techniques to enable the creation of effective illustrations and to avoid the pitfalls of isolatory application of separate techniques, which often leads parts of the image to compete visually and confuse the user by providing misleading information. As such, our parameterizations are based on the illustrators' use of a holistic approach in their renderings, choosing the styles and techniques, so they not only provide useful insight but also visually "agree" in the image as a whole.

For the context of this work, we define a *volume feature* as any part of volume that may be of interest to an illustrator or an observer and can be separated by a volume transfer function, gradient transfer function, or volume segmentation map.

We begin the paper by reviewing related work in Section 2. Section 3 presents the underlying algorithms that we use to outline features and fine details in the volumes, while Section 4 focuses on these depth perception effects, describing the color changes and the line styles adjustments, based on techniques used by illustrators.

- *N.A. Svakhine is with Adobe Systems Inc., WT-10 226, 345 Park Ave, San Jose, CA 95110. E-mail: svakhine@adobe.com, svakhine@gmail.com.*
- *D.S. Ebert is with School of Electrical and Computer Engineering, Purdue University, 465 Northwestern Ave., West Lafayette, IN 47907. E-mail: ebertd@purdue.edu.*
- *W.M. Andrews is with Bill Andrews & Associates, Inc., 19 Rockbrook Road, Augusta, GA 30909. E-mail: Bill@Andrewsmedart.com, bandrews@mcg.edu.*

Manuscript received 14 Sept. 2007; revised 21 Nov. 2007; accepted 5 Feb. 2008; published online 24 Mar. 2008.
Recommended for acceptance by M. Chen, C. Hansen, and A. Pang.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCGSI-2007-09-0139.
Digital Object Identifier no. 10.1109/TVCG.2008.56.

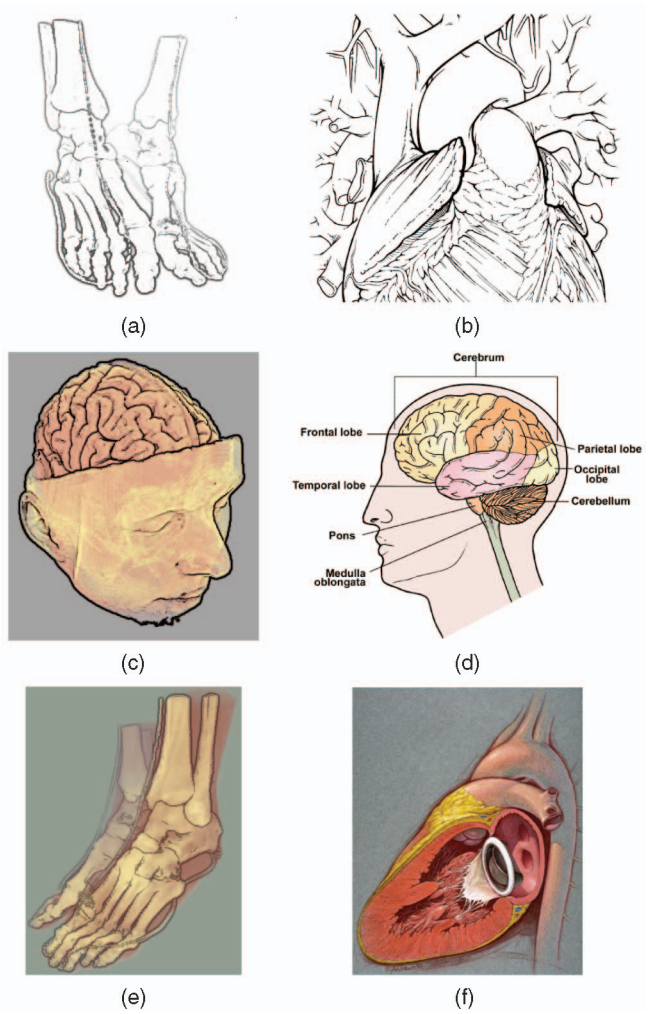

Fig. 1. Visible Woman Feet CT data set rendered using a (a) line-drawing style is inspired by (b) medical illustration techniques. (c) Enhanced rendering of MRI Head with depth darkening and detail emphasis produces rendering similar to (d) traditional educational medical illustration. (e) Feet CT with depth darkening of the bone area and depth cueing; (f) similar techniques are used by medical illustrators to emphasize the structure and convey depth information.

Section 5 describes heuristics for effect parameterization to effectively replicate various illustrative styles.

2 Previous Work

Since the introduction of volume illustration as a stand-alone research area, there has been a number of research efforts to implement different aspects of the illustrative techniques: nonphotorealistic shading, focus + context, and using lines for illustration.

Nonphotorealistic shading. By nonphotorealistic shading techniques, we consider the techniques that produce rendering styles that are drastically different from standard Phong-shaded volume rendering. Ebert and Rheingans [7] highlighted several key methods in adding such enhancements to a volume rendering pipeline. Lu et al. presented a stipple-based illustration technique for volumes [12] and had also shown an example-based volume rendering system [11], which allows the generation of illustrations by example textures. Treavett et al. [21] presented pen-and-ink methods for volume shading, and Nagy et al. [15] used hatching techniques for stylized rendering of the volume isosurfaces.

Focus + context techniques. A number of techniques have been developed in order to render illustrations using the traditional principles of "focus + context" with various levels of representation. Viola et al. [22] proposed importance-based volume rendering, where some features inside a volume are declared "important" by the user, and the rendering provides maximum detail and attentive focus for these features. Svakhine et al. [18] used a spherical area within the volume for the "focus" and described a set of illustration motifs that control rendering styles for both focus and context, creating different representations based on the purpose of the illustration. Bruckner and Gröller [2], [3] incorporated many similar illustrative techniques, adding exploded views and labels to the illustrative technique toolbox. Wang et al. [23] presented magnification lens rendering techniques to magnify the features of interest in focus while compressing the context volume regions. Taerum et al. [20] demonstrated the illustration-inspired system for depicting contextual close-up views of selected regions of interest within volumes. Cole et al. [6] presented an interactive system for placing emphasis in stylized renderings of 3D models, utilizing focus + context techniques for illustrative surface model rendering.

Lines/halos illustration techniques. For line illustration approximations, there are many systems that highlight volume features by using contours or silhouettes. The work by Lum and Ma [14] and Svakhine and Ebert [17] provided an interactive approach for both silhouettes and feature outlines/halos; however, these methods have a problem of nonuniform thickness in the outlines. The works by Interrante et al. [9] and Kindlmann et al. [10] described ways to emphasize ridges and valleys on the isosurfaces, as well as improved silhouettes; however, these approaches require a calculation of curvature information for the volume data, which can be computationally expensive. Burns et al. [5] proposed a line illustration framework in which the lines are extracted directly from the isosurfaces and are used as the fundamental rendering primitives; however, the interactive speeds for that algorithm can only be achieved without the hidden-line removal step. Fischer et al. [8] also used isosurface-based illustrative line drawing with silhouette emphasis. Recently, Bruckner and Gröller [4] extended these halo approaches to volumetric halos around edges of features that are used to indicate occlusion and enhance depth perception.

Surface-based illustration. The illustration approaches that use surface models are a very well developed area. Several of these recent results are extended to volumes in our current work. The main technical basis for this work is the technique for depth buffer unsharp masking by Luft et al. [13]: a fast hardware accelerated algorithm where the difference between the original depth buffer content and a low-pass filtered copy is utilized to determine information about the outlines of the objects. In addition, part of this work is dedicated to outlining the "plane breaks," or fine-scale details on the volume features' surfaces. The recent work by Rusinkiewicz et al. [16] addresses this problem with the "exaggerated shading" approach: a shading model for surfaces and volumes that uses principles of hand-shaded terrain illustration to depict fine-scale details.

Our approach differs from these previous approaches in several aspects. First, we only use the given volume data (raw volume values and segmentation (if available)), without computing any extra volume information such as curvature or isosurfaces (precomputation of volume gradients is usually avoided by an in-shader gradient evaluation). Second, within the texture-based volume-rendering pipeline, our per-slice shading modification is minimal, only adding a binary condition that determines if a given sample belongs to a feature. The final algorithm is run on a set of 2D images, and this combination allows us to avoid any significant performance penalties. Finally, our approach is unifies several enhancements: it provides a basis for both volume feature outlining and depth cueing.

3 Outlining Techniques

One main contribution of this work is the design of volume rendering enhancements that enable fast and effective outlining techniques and provide strong perceptual cues of object importance and spatial relationships within volumes. We adapted the original work by Luft et al. [13], which is an image-based approach that can operate on any rendered or otherwise acquired image with a corresponding depth buffer D.

The main principle of the work is constructing the "unsharp-mask" (ΔD) of the image. This is done by using a Gaussian filter G_σ, where σ is the size of the filtering kernel, to construct a blurred depth buffer $G_\sigma * D$. Subsequently, the unsharp mask is constructed as a difference field between the original depth buffer and the blurred depth buffer:

$$\Delta D = G_\sigma * D - D. \tag{1}$$

The next step is to modify the original image based on ΔD. Luft et al. [13] demonstrated several methods for such modifications, producing various effects. Particularly important for the current work is their "darkening effect," where the image brightness is reduced proportional to the function:

$$I = \lambda \cdot \Delta D^{-}, \tag{2}$$

where ΔD^{-} is the negative component of ΔD. This modification darkens the areas of depth discontinuities, effectively creating an outline (or "halo") for the image, and the parameter λ controls the impact of the effect.

In order to apply this technique to volume visualization, the following issues must be considered. First, unlike surface rendering, the depth information for the volume rendered image is not readily available. Second, in volumes, multiple features may occlude and intersect each other, and thus, the depth information for one feature can invalidate the other feature's depth. In order to overcome these problems, we use extra rendering buffers and transfer functions in our volume-rendering pipeline. The details are described in the following sections.

3.1 Depth Information for Volume Rendered Images

For the basis of this system, we are using a multitransfer function hardware-accelerated rendering approach described in [18]. Since we are using texture-based back-to-front slice-by-slice volume rendering, by default, the entire slice polygon fills the depth buffer with the slice's current depth, which does not give any meaningful information about the volume features. We need the ability to generate depth information based on volume contents, and thus, we need the criteria to determine whether a current volume sample contributes to the depth buffer or not. To "highlight" a distinct volume feature in this way, we utilize the familiar concept of a transfer function. We introduce an extra *feature transfer function* τ, which is a binary function, set by the user to 1 for the sample values that belong to the feature of interest, and 0 otherwise. Note that the τ function is completely independent of other standard transfer functions in our rendering pipeline and can use the volume segmentation, as well as the transfer function classification.

To create a depth buffer D inside a slice-rendering fragment program, we use the condition in (3), where i is the current slice's number, P is the current point in the volume, and $\alpha()$ is the main opacity function for the volume rendering. The initial depth buffer D_0 is initialized with the background depth (usually the corresponding value is 1):

$$D_i = \begin{cases} d_{slice} & \text{if } \tau(P) = 1 \text{ and } \alpha(P) > 0, \\ D_{i-1} & \text{otherwise.} \end{cases} \tag{3}$$

Note that instead of writing the current slice's depth d_{slice}, we can modify the algorithm to just write the constant depth value into the buffer, effectively creating a "mask" for the feature. In this case, since there is no depth variation, we lose the outline effect within the feature, but the overall outline becomes uniform, and both of these may be desirable effects for an illustrator.

Fig. 2 shows the comparison between a regularly rendered volume (Fig. 2a) and various effects aimed to enhance the bone outline. Fig. 2b shows the traditional silhouette criteria from that in [17] used to darken the color around the bone edges. While it is a viable shading model that enhances perception of the 3D shape, the feature is not clearly outlined as was intended. Fig. 2c shows the application of the new effect of image darkening through unsharp-masking the depth buffer. Fig. 2d demonstrates the same effect while using a constant depth value for the entire feature, removing most of the fine details (including the metallic wire on the cadaver).

3.2 Visualizing Fine-Scale Details

For surface models, sometimes the "plane breaks," or fine-scale details, are contextually important. These details are sometimes represented by small bumps or cavities and are characterized by very little depth variance (Fig. 3a). Because of this, the original depth darkening approach fails to detect these details.

For a simple solution to this problem, we can add a power term in the original importance function's construction:

$$I = \lambda \cdot (\Delta D^{-})^{p}. \tag{4}$$

This modification, with value $p < 1$, can bring some of these details out, but for most fine-scale details, the depth difference is too low to be used in the calculations, and even the precision limit of the depth buffer can be a serious obstacle. The darkening masks in Figs. 4a and 4b illustrate the results of this simple approach, clearly failing to highlight the separate small bones of the foot.

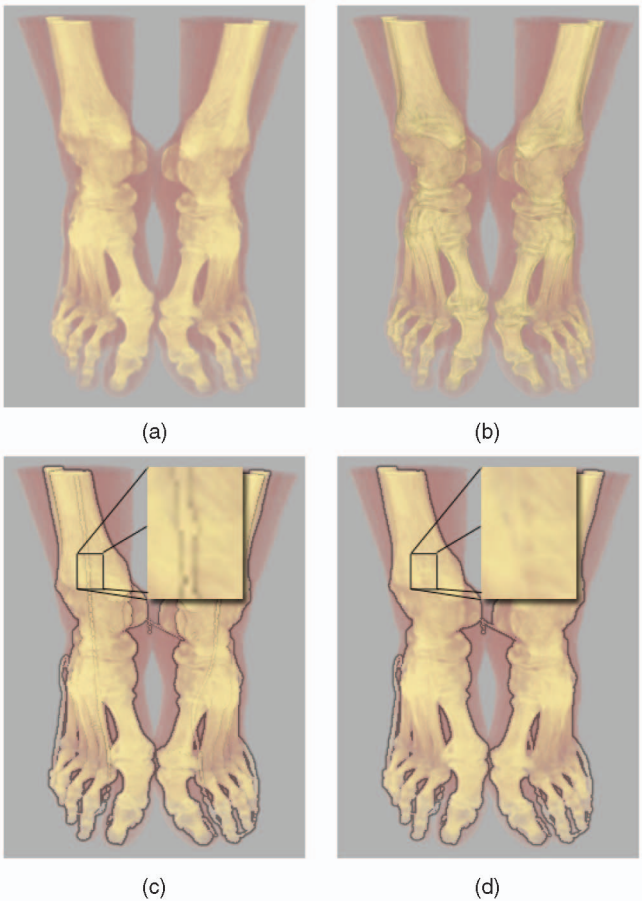

Fig. 2. (a) Regular colored volume rendering, (b) traditional silhouette criteria $((\vec{n},\vec{v}))$ used to darken the edges, (c) basic outline using correct depth value, and (d) basic outline using constant depth value. The enhancements are only applied to the bone and in the second row the areas with fine detail (metallic wire) are magnified.

However, if we examine how the surface normals change for the same part of the volume, the magnitude of the change is much more substantial (Fig. 3b). Therefore, we have developed methods to analyze the normals' field and use it to highlight the needed surface details.

In order to generate the normal field for the feature's surface, we use an extra rendering buffer N. We use the same approach, and the same criteria as in constructing the depth buffer (Section 3.1); the only difference is that we are writing the volume gradient $\vec{n}$ into the target buffer N, instead of writing the current depth to the buffer D.

We have developed two approaches to generate a scalar field ΔN using two different metrics for the normal change in image space. We can use either approach or both of them in combination.

The **central difference** approach computes screen-space central differences for each point in N. Note that

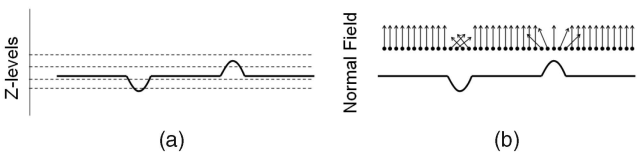

Fig. 3. Comparison of using Z values versus normal field for detecting plane "breaks."

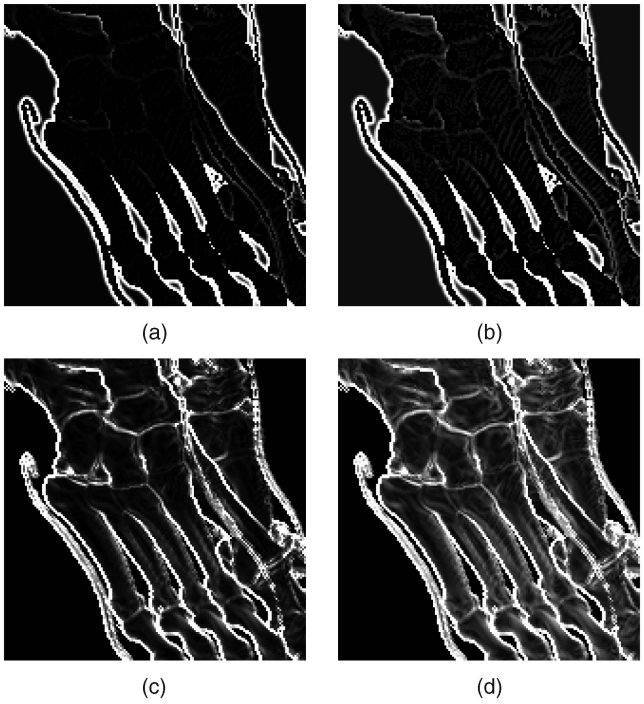

Fig. 4. Magnified view of metatarsal bones, visualizing the values of different masks. Images (a) and (b) use a Depth Unsharp Mask with power terms $p=1$ and $p=0.6$, respectively. Images (c) and (d) use a Normal Difference Mask with power terms $p=1$ and $p=0.6$, respectively.

we want the measure of the difference between normal directions; hence, for the difference metric, we use the dot-product term:

$$\Delta N_{(x,y)} = \left[1-\left(\vec{n}_{(x+1,y)}\cdot\vec{n}_{(x-1,y)}\right)\right]+\left[1-\left(\vec{n}_{(x,y+1)}\cdot\vec{n}_{(x,y-1)}\right)\right]. \tag{5}$$

In the **unsharp-masking** approach, we use the already mentioned unsharp-masking filter G_{σ_N} on the normal field instead of the depth buffer. Again, the difference between the "blurred" normal and original normal is computed through a dot-product:

$$\Delta N_{(x,y)} = 1-\left(\left(G_{\sigma_N}*\vec{n}_{(x,y)}\right)\cdot\vec{n}_{(x,y)}\right). \tag{6}$$

After constructing the ΔN field using either of those methods, we calculate the "normal importance function" I_N by filtering the ΔN values. At this stage, any transfer function can be used, but in practice, we have found that the following function produces good results:

$$I_N = \phi(\Delta N) = \begin{cases} 0 & \text{if } \Delta N < n_{min}, \\ \left(\frac{\Delta N - n_{min}}{n_{max}-n_{min}}\right)^{p_N} & \text{if } n_{min} \le \Delta N \le n_{max}, \\ 1 & \text{if } \Delta N > n_{max}. \end{cases} \tag{7}$$

The parameters in the function in (7) govern the following: n_{min} and n_{max} create a "window" that allows the user to focus on the details with a particular magnitude (e.g., rough surface details versus fine surface details), and the power term p_N changes the smoothness of the intensity transition from n_{min} to n_{max}, which controls the sharpness of the resulting outlines.

Fig. 4 illustrates the difference between the proposed method and the original depth buffer approach and also

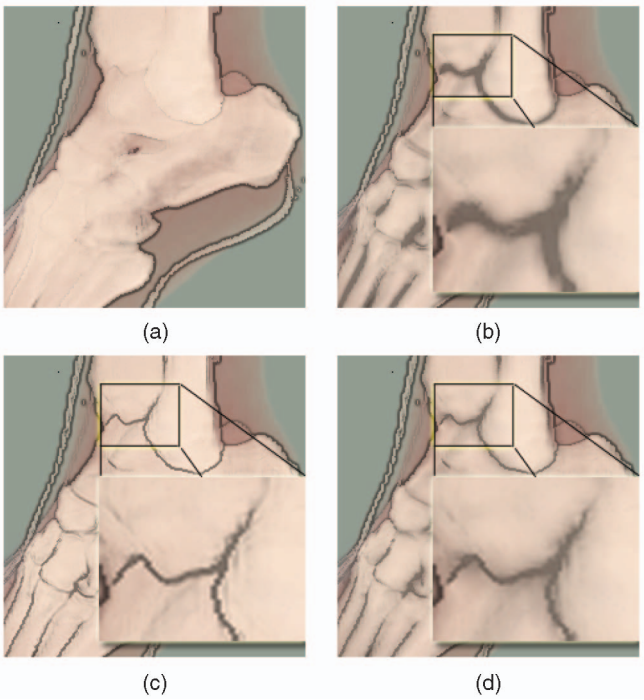

Fig. 5. Comparison of different methods of fine detail outline in a magnified view of the tarsus area of the foot. (a) Is an original depth-darkened image; (b) uses an unsharp-masking term for the normal importance function ($\sigma_N = 2$ percent of diagonal); (c) uses a central difference for the normal importance function; and (d) uses a weighted combination of the two ($\sigma_N = 5$ percent of diagonal). The final importance function is $(max(I, I_N))$.

visualizes the impact of the power term. The images visualize the importance maps only, without the original image, and it is clearly seen that more of the internal details are revealed by the normal central difference approach.

To generate the final darkening map for the image, we use a combination of the original importance function I and the new normal importance function I_N. The images in Figs. 5b, 5c, and 5d have been generated using the importance function of $max(I, I_N)$.

As we see in Fig. 5, the central difference method is able to highlight more details compared to the unsharp-masking method; however, the latter approach is more flexible due to the variable size of the filtering kernel, which makes it better at producing smoother lines and creating secondary outlines of lower intensity (Fig. 5d). Fig. 6 shows the result of application of these techniques for the Visible Male CT Head data set.

3.3 Handling Multiple Features

Our techniques can be generalized to highlight multiple potentially overlapping features of the volume. We approach this issue by having separate feature transfer functions $\tau^{(k)}$ for each feature k and having corresponding depth buffers $D^{(k)}$ and normal buffers $N^{(k)}$ that are constructed independently.

The algorithm then constructs multiple importance functions $I^{(k)}$ for each feature and combines them in order to produce a final image. We can use several heuristics for combining the functions, however, extra care needs to be taken when features overlap in image space, e.g., one is behind the other. In such cases, depending on the rendering

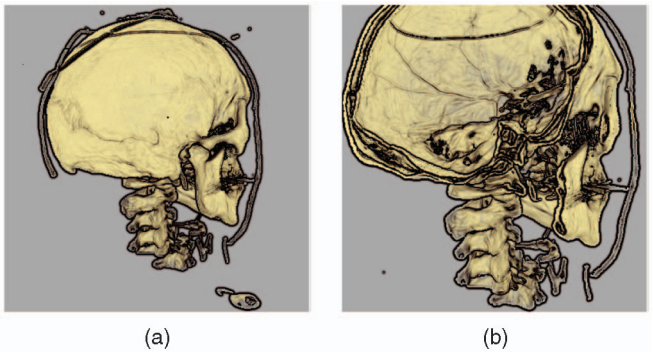

Fig. 6. Using the fine detail outline techniques on the Visible Male CT Head data set. (a) The lambdoid suture is visible on the back of the skull, and (b) the intracranial vessel patterns are visible from inside the skull.

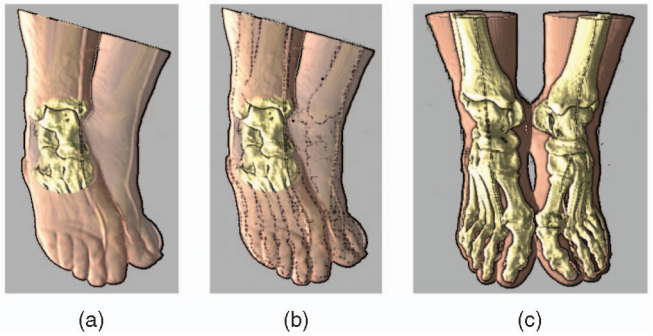

Fig. 7. Example of stylization based on occlusion. This illustration focuses on the tarsal joints of the leg. For the areas where skin occludes the bone, (a) the image hides the bone outline, and on (b), the outline is drawn in dashed lines. In (c), the bone is declared "important" and is always brought to the front even though the skin is still opaque.

style, the outline of the occluded feature can be removed, dimmed, or otherwise stylized (see the examples in Figs. 7a and 7b). In extreme cases, such as when the occluded object is of extreme importance, our rendering pipeline can be easily modified so the feature is brought forth regardless of the depth and overlapping, similar to Viola et al.'s importance rendering results [22] (see Fig. 7c).

4 Depth Perception Enhancements

Gradients of contrast, line weight, periodicity, density, and detail can be used in an illustration to provide the viewer with a sense of looking into a volume of space. This is referred to as "aerial perspective," and is based on observable phenomena in the world. The ability to manipulate the gradients for providing this effect is essential, and simple implementations of depth enhancement for distance color blending have been described in [7] and [17]. Here, we significantly extend the set of interactive effects for improved depth perception, adding depth cueing, line fading, and blurring.

4.1 Depth Parameter δ

The original depth in our depth buffer has a range from 0 to 1, with 1 being the background depth. However, the user does not often want depth-guided subjugation to affect the front features of the image, rather having the subjugation effect start further from the eye. In order to achieve this flexibility, we filter the original depth value d to construct

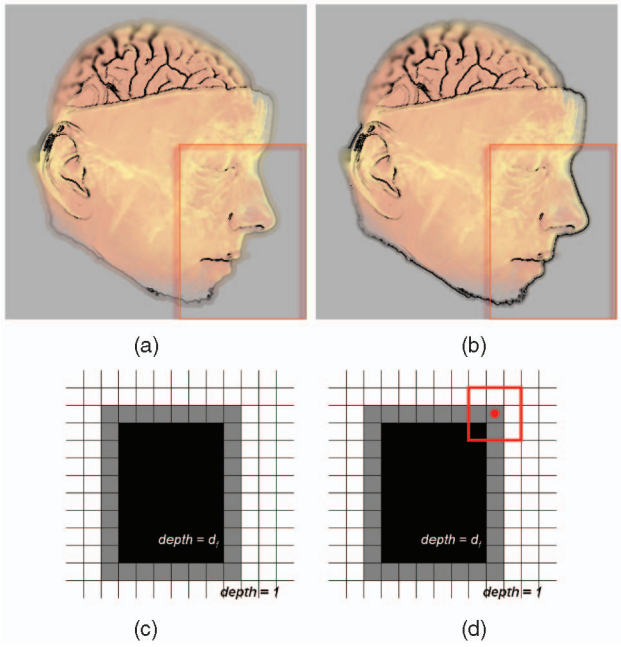

Fig. 8. Depth blurring effect: (a) effect applied incorrectly on the outer outline (framed), as the front features are blurred, and (b) corrected image. Corresponding depth buffer calculations: (c) original depth in border (gray) pixels is 1, (d) to assign a correct depth value, we calculate the minimum depth value in the σ-neighborhood.

$\delta = \phi(d)$ and use it as the parameter that controls the application of various depth enhancements. For the sake of explanation, the filtering function ϕ has one parameter d_{start} (starting depth) that controls how far along the ray the depth enhancement starts to affect the image.

Generally, if $\delta = 0$, the depth effect is not applied. If $\delta = 1$, the effect is applied fully, and, in-between, values of the affected and nonaffected samples are interpolated using (8):

$$\delta = \phi(d) = \begin{cases} 0 & \text{if } d < d_{start}, \\ \left(\frac{d - d_{start}}{1 - d_{start}}\right) & \text{if } d \geq d_{start}. \end{cases} \quad (8)$$

4.1.1 Depth Parameter δ in Outline Regions

In some of the enhancements described in this section, the parameter δ is used to control outlining. However, the original depth of the feature outline against the background (*border pixels*) is equal to 1 (Fig. 8c), and this may cause incorrect subjugation of the lines, as they all will be considered to be at the background depth (Fig. 8a). Thus, in order for the effect to be applied correctly, the border pixels need to be assigned the same δ as the main feature. To do this, we need to identify the border pixels and then look "around" each of these and identify the depth of the feature that corresponds to this outline. Our heuristic for the border pixels is straightforward: they all have original depth $d = 1$, while their outline parameter I is nonzero. Finding the corresponding feature's depth is, in the general case, impossible because of a potential ambiguity, so we use a suitable approximation. Since the outline cannot be thicker than the filtering kernel size σ, we calculate d_{min} to be the minimum depth value of the original image in the σ-neighborhood (Fig. 8d). Finally, the parameter δ is calculated using d_{min} instead of the original depth. The corrected image is shown in Fig. 8b: nose, mouth, and chin outlines are correctly highlighted instead of being subjugated by depth cueing, as in Fig. 8a.

4.2 Depth Enhancements

We have implemented several depth enhancements including depth cueing, color blending, blurring, and line fading. In calculating most of these enhancements, we use the linear interpolation operator *lerp*, which is defined by (9):

$$lerp(X, Y, \delta) = (1 - \delta) \cdot X + \delta \cdot Y. \quad (9)$$

4.2.1 Depth Color Blending

Traditional color blending uses δ to blend between the original color and the background. The extra parameter for this approach is the "depth color" ($Color_{depth}$) that indicates the color blend at the maximum depth. Depending on the style of the drawing, this color may be equal to the background color. This simple effect is included here for the sake of completeness, as it has been described previously in [7]:

$$Color_{new} = lerp(Color_{old}, Color_{depth}, \delta). \quad (10)$$

4.2.2 Depth Cueing

Depth cueing reduces the contrast of the sample based on the depth enhancement coefficient. It is implemented by interpolating between the original color and a user chosen gray color:

$$Color_{new} = lerp(Color_{old}, GRAY, \delta). \quad (11)$$

4.2.3 Depth Blurring

Depth blurring is inspired by both photography and traditional illustration. As the optical limitations of cameras produce an image that is focused on a particular distance, depth blurring has been widely used in both artistic and scientific photography to accentuate details at certain depths and blur the context. To achieve a similar effect, we apply Gaussian blurring to the final image using the parameter δ to control the kernel size. The extra parameter for this approach, analogous to the previously described technique, is the "maximum depth blur" ($Kernel_{max}$): a filtering kernel size for the samples at maximum depth:

$$Kernel = lerp(1, Kernel_{max}, \delta), \quad (12)$$

$$Image_{new} = G_{Kernel} * Image_{old}. \quad (13)$$

Depending on the desired style of subjugation, $Kernel_{max}$ usually lies at 1-10 percent of the result image diagonal, ranging from "slight" to "strong" blurring. As our blurring is done directly on the GPU, the size of the kernel affects the performance of this effect.

4.2.4 Line Fading

Similar to the use of color in the previous effects, we can scale down the importance value I, which gives an effect of line fading with increasing depth of the feature (Fig. 10b):

$$I = lerp(I, 0, \delta). \quad (14)$$

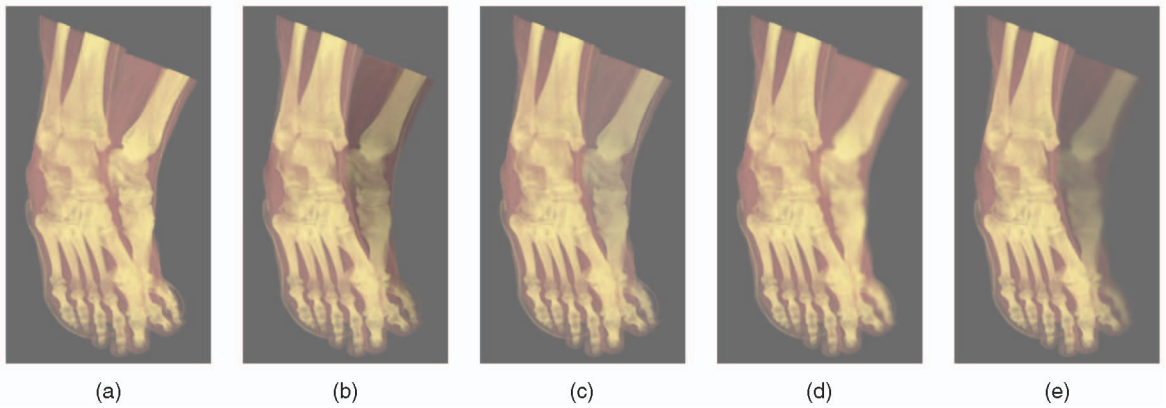

Fig. 9. Subtle application of depth perception enhancement effects. (a) Is an original volume rendering, (b) uses the background color for depth color blending, (c) uses reduced contrast, (d) uses depth blurring, and (e) uses all of these effects in combination.

All of the effects described above can be applied independently or combined with each other, as shown in Fig. 9, for regular depth enhancements and, in Fig. 10, for outline-based depth enhancements.

5 Maintaining Illustrative Styles

The default use of the importance field I is the simple darkening of the image color [13]. However, with excessive use of darkening, the outline conveys the shape information, yet the effect starts to visually "fight" the style of the original image. Thus, we need to control the contextual changes of the effects, based on the underlying nonenhanced image. Based on the traditional illustrator's expertise for the line illustration, we have developed algorithms to apply the importance map I so that the overall style remains consistent. For the scope of this work, we have identified the following heuristics for choosing the color C_I of the generated outlines:

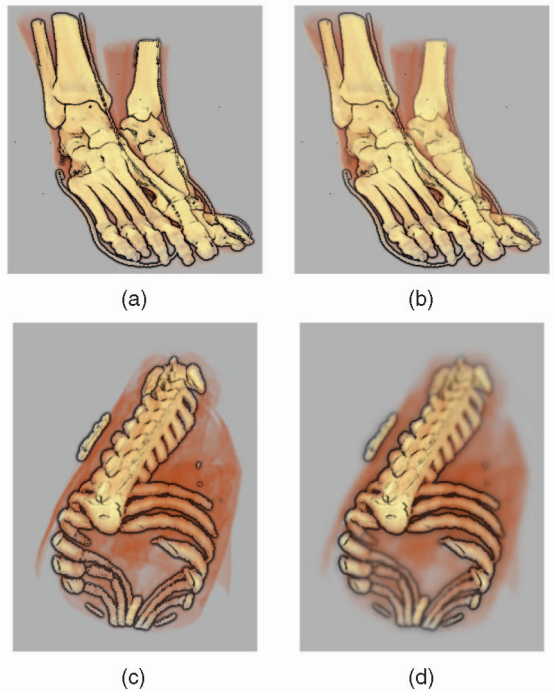

Fig. 10. Application of depth perception enhancement effects to outlines. Feet CT data set: (a) is a volume rendering enhanced with the outlines; (b) uses line fading and depth cueing. Feline vertebrae CT data set: (c) is the original outlined image; (d) uses line fading, depth cueing, and blurring.

- For simple color schemes, the color of the lines can simply be a color specified by the user, as part of the illustration style (Fig. 11a):

$$C_I = C_{user}. \tag{15}$$

- The color can be chosen to be a fraction α of the original background color. This allows for subtler lines for lower perceptual impact of the lines in the illustration (Fig. 11b):

$$C_I = \alpha \cdot C_{back}. \tag{16}$$

- In case of diverse backgrounds, an illustrator can change an outline color of the feature to achieve the highest possible contrast with the background. An illustrator could use many ways to choose the contrast color [1]. Here, for the purpose of explanation, we use the simple heuristic of picking the black or the white color based on background color's brightness (Fig. 11c):

$$C_I = \begin{cases} (1,1,1) & \text{if brightness of } (C_{back}) < 0.5, \\ (0,0,0) & \text{if brightness of } (C_{back}) \geq 0.5. \end{cases} \tag{17}$$

6 Implementation

The implementation is based on the hardware-accelerated interactive volume illustration system described in [18] that allows selective application of illustrative effects by using complex fragment shading and multiple transfer functions for controlling various illustrative effects. In our initial user interface design, we have exposed the most essential parameters to the user: the feature transfer function τ (through a standard transfer function editor), the size of the filtering kernel σ, a slider to control blending of the different methods of fine detail outlining (Fig. 5), the starting depth for depth enhancements, and choices for coloring the

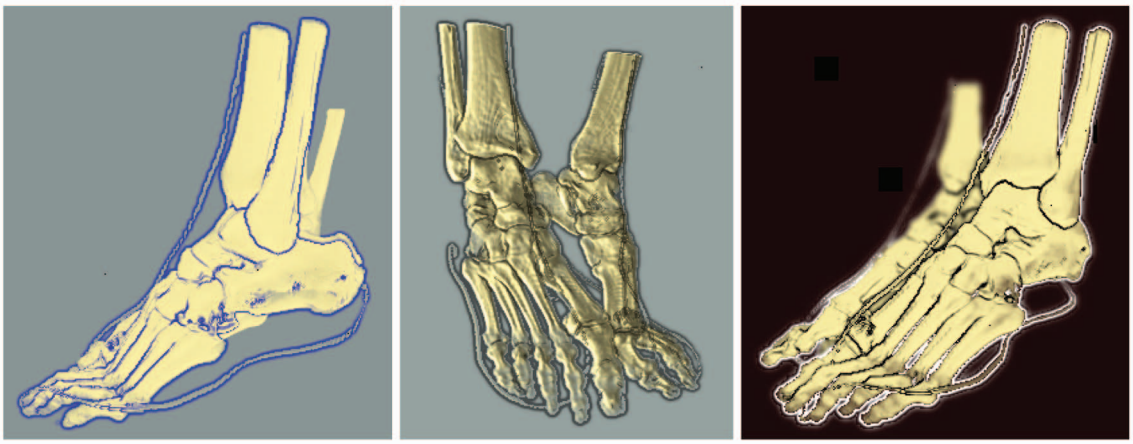

Fig. 11. Outline color choices, based on styles and purpose of an illustration. (a) Flat-shaded yellow-blue style, lines are emphasized with blue color. (b) Subtler outlines add to the overall perception of the Phong-shaded bone without dominating the image. (c) For a high-contrast style, the feature is outlined using either black or white colors.

outlines (Section 5). For greater flexibility and variety in the effects, more parameters can be exposed, such as the power term in fine-scale detail visualization (Section 3.2).

The texture-based volume rendering stage has been modified to create the additional buffers necessary for the effects: the depth buffer (Section 3.1) and the normal buffer (Section 3.2). Current graphics hardware supports such extra buffers with Multiple Render Targets (MRT) extension. We can attach multiple textures as rendering targets to a frame buffer and write into these targets from the fragment program. Since each target is a four-component buffer, we use each target to store both normal buffer (three components) and the depth buffer (1 component) of the feature. The number of possible attachments is four, so we can track up to three features per rendering pass.

The actual application of the outline/depth effects is performed in an extra pass, during which the color buffer and the extra buffers are processed, and the final image is created. Most of the effects are implementable in one pass. To allow for combinations of different effects, we have implemented all of them in a single fragment program that is activated during the extra pass, and the choice of the effects is controlled by the user. The only effect that requires an additional pass is depth blurring, because it needs to perform a "gather" operation on the final image, and during the first extra pass, the surrounding pixels may not be ready.

Fig. 12 visualizes the intermediate values that are used in the rendering of the basic "depth-darkening" effect (Section 3.1). Fig. 12a shows a regular volume rendering of the foot data set. Fig. 12b shows the selective depth buffer generated for the bone part of the data set. Fig. 12c shows the result of Gaussian blur on the depth buffer (filter $\text{kernel} \approx 2.5\ \text{percent}$ of the image diagonal), and Fig. 12d shows the difference, ΔD, between blurred and original depth buffers. Since the difference ranges from $[-1, +1]$, we use simple rescaling to visualize the range, where 50 percent gray color represents zero, brighter shades represent positive values, and darker shades represent negative values. Fig. 12e shows the result of darkening the original image scaled down by the importance function $I = \Delta D^{-}$, which produces a natural depth darkening effect as front features "casts shadow" on the features closer to the background.

The most computational-intensive operation is the Gaussian blur (which is used in unsharp masking and in depth blurring), and the major factor for the performance is the filtering kernel size. Since the Gaussian blur can be operated on each pixel independently, the algorithm is highly parallel, and even the straightforward GPU implementation is rather efficient. For a blurring kernel of

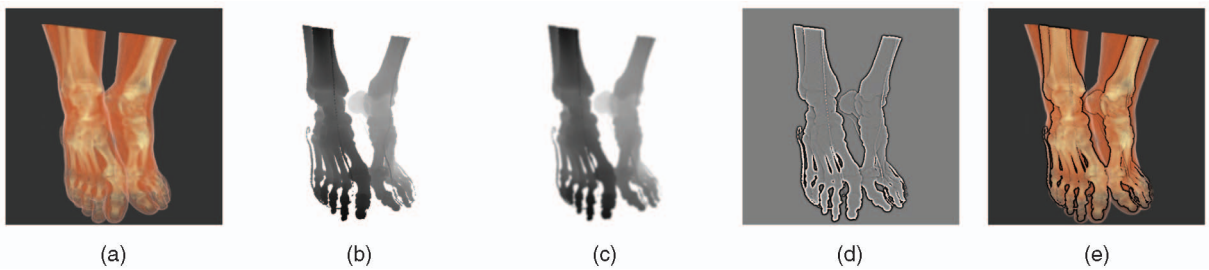

Fig. 12. Intermediate values used in the rendering process. (a) and (b) Are the original image and generated depth buffer. (c) Represents a blurred depth buffer, (d) is a difference between the blurred and original depth buffer, and (e) shows how this information is used for depth darkening of the original image.

Fig. 13. Examples of other data sets: (a) Hand CT data set, (b) Skull Pathology (craniosynostosis) CT data set, and (c) Knee Joint CT data set. The outlines provide quick cues to the shapes of the bone structures, without costly advanced volume shading.

5×5, the CPU-based implementation's performance is 2.7 fps, and the GPU-based (GeForce 8800 GTX) implementation's performance is 33 fps. For a kernel of 33×33, the CPU has the performance of 0.07 fps, and the GPU implementation has 9.5 fps. In addition, due to the ability of graphics hardware to perform operations on 4-component vectors, the blurring of the color buffers takes the same time as it would take for a single-component buffer. Interactive performance and manipulation is demonstrated in a video, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2008.56.

Overall, our algorithm's implementation requires the following graphics hardware capabilities: 1) floating-point frame buffers (32-bit floating point are highly recommended, as we have to store an internal 24-bit depth buffer without loss of precision), 2) hardware support of loops in the fragment programs for blurring, and 3) MRTs for the fast rendering of multiple extra buffers.

7 Conclusion and Future Work

We have presented a collection of new illustrator-guided globally consistent fast interactive methods for volume illustration enhancements that uses artistic styles for outlining features and conveying depth information. These techniques can be applied to generate pure line illustrations (Fig. 1a), as well as helpful feature outlines (Fig. 13). They can also be used as context-aware enhancements to existing images (Fig. 11). Our general approach is a combination of 3D volume rendering and 2D image processing. These methods can be easily combined with existing illustrative techniques and are very effective in creating attentive focus through outlining and subjugating more distant samples.

The selection of volume features can be done through a transfer function or any kind of segmentation mapping. All the methods are interactive and do not require extra volume data to be precomputed and stored in video memory. The changes to the slice rendering stage of the basic volume illustration algorithm [18] are limited to an extra transfer function lookup and do not have a significant impact on the overall performance. The extra effects are performed on the set of buffers at the final image resolution; thus, the performance of the final 2D enhancements depends only on the final image size and not the volume size. All the algorithms are efficiently hardware-accelerated, with all extra buffers residing in the GPU memory.

Due to the limitations of the current graphics hardware (limited number of MRTs), this approach is limited to just three distinct features, which we believe can be solved by multiple passes. This work also does not address the cases when the different outlined features are semitransparent and overlapping. These cases require extra effort in order to achieve correct blending for the outlines in the final image.

Some of the images in this paper also show problems that noisy data can create: the noise voxels that get classified as features are sometimes highlighted and outlined, producing easily noticeable artifacts in the image, while their impact is not as evident in traditional volume-rendered images. Volume smoothing, better transfer functions, and/or segmentation techniques should eliminate these problems.

For future work, this approach could be extended by using extra surface parameters such as curvature values, for more accurate representation of the inner detail and artifact elimination. Our GPU-based blurring algorithm is a brute-force Gaussian convolution, which can be altered and

optimized in order to achieve better performance. Another direction would be applying these methods to other types of data and volume illustration (e.g., flow illustration [19]).

ACKNOWLEDGMENTS

This work was supported by Adobe Systems, NVIDIA, and the US National Science Foundation under Grants 0081581, 0121288, and 0328984. The authors also wish to thank the reviewers for their helpful comments in improving this manuscript.

REFERENCES

[1] J. Albers, *Interaction of Color.* Yale Univ. Press, 2007.
[2] S. Bruckner and E. Gröller, "VolumeShop: An Interactive System for Direct Volume Illustration," *Proc. IEEE Visualization,* pp. 671-678, Oct. 2005.
[3] S. Bruckner and E. Gröller, "Exploded Views for Volume Data," *IEEE Trans. Visualization and Computer Graphics,* vol. 12, no. 5, pp. 1077-1084, Sept./Oct. 2006.
[4] S. Bruckner and E. Gröller, "Enhancing Depth-Perception with Flexible Volumetric Halos," *IEEE Trans. Visualization and Computer Graphics,* vol. 13, no. 6, pp. 1344-1351, Nov./Dec. 2007.
[5] M. Burns, J. Klawe, S. Rusinkiewicz, A. Finkelstein, and D. DeCarlo, "Line Drawings from Volume Data," *ACM Trans. Graphics (Proc. ACM SIGGRAPH '05),* vol. 24, no. 3, pp. 512-518, Aug. 2005.
[6] F. Cole, D. DeCarlo, A. Finkelstein, K. Kin, K. Morley, and A. Santella, "Directing Gaze in 3D Models with Stylized Focus," *Proc. Eurographics Symp. Rendering,* pp. 377-387, June 2006.
[7] D. Ebert and P. Rheingans, "Volume Illustration: Non-Photorealistic Rendering of Volume Models," *Proc. IEEE Visualization,* pp. 195-202, 2000.
[8] J. Fischer, D. Bartz, and W. Straßer, "Illustrative Display of Hidden Iso-Surface Structures," *Proc. IEEE Visualization,* pp. 663-670, Oct. 2005.
[9] V. Interrante, H. Fuchs, and S. Pizer, "Enhancing Transparent Skin Surfaces with Ridge and Valley Lines," *Proc. IEEE Visualization,* pp. 52-59, 1995.
[10] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Moller, "Curvature-Based Transfer Functions for Direct Volume Rendering: Methods and Applications," *Proc. IEEE Visualization,* pp. 513-520, 2003.
[11] A. Lu and D.S. Ebert, "Example-Based Volume Illustrations," *Proc. IEEE Visualization,* pp. 655-662, 2005.
[12] A. Lu, C.J. Morris, D.S. Ebert, P. Rheingans, and C. Hansen, "Non-Photorealistic Volume Rendering Using Stippling Techniques," *Proc. IEEE Visualization,* pp. 211-218, 2002.
[13] T. Luft, C. Colditz, and O. Deussen, "Image Enhancement by Unsharp Masking the Depth Buffer," *ACM Trans. Graphics,* vol. 25, no. 3, pp. 1206-1213, 2006.
[14] E.B. Lum and K.-L. Ma, "Hardware-Accelerated Parallel Non-Photorealistic Volume Rendering," *Proc. Second Int'l Symp. Non-Photorealistic Animation and Rendering (NPAR '02),* pp. 67-74, 2002.
[15] Z. Nagy, J. Schneider, and R. Westermann, "Interactive Volume Illustration," *Proc. Vision, Modeling, and Visualization Workshop,* Nov. 2002.
[16] S. Rusinkiewicz, M. Burns, and D. DeCarlo, "Exaggerated Shading for Depicting Shape and Detail," *ACM Trans. Graphics (Proc. ACM SIGGRAPH '06),* pp. 1199-1205, 2006.
[17] N. Svakhine and D.S. Ebert, "Interactive Volume Illustration and Feature Halos," *Proc. Pacific Conf. Computer Graphics and Applications (Pacific Graphics '03),* pp. 347-354, 2003.
[18] N. Svakhine, D.S. Ebert, and D. Stredney, "Illustration Motifs for Effective Medical Volume Illustration," *IEEE Computer Graphics and Applications,* vol. 25, no. 3, pp. 31-39, May/June 2005.
[19] N. Svakhine, Y. Jang, D.S. Ebert, and K. Gaither, "Illustration and Photography Inspired Visualization of Flows and Volumes," *Proc. IEEE Visualization Conf.,* pp. 687-694, 2005.
[20] T. Taerum, M.C. Sousa, F. Samavati, S. Chan, and R. Mitchell, "Real-Time Super Resolution Contextual Close-Up of Clinical Volumetric Data," *Proc. Eighth Eurographics/IEEE VGTC Symp. Visualization (EuroVis '06),* pp. 347-355, 2006.
[21] S.M.F. Treavett and M. Chen, "Pen-and-Ink Rendering in Volume Visualisation," *Proc. IEEE Visualization,* pp. 203-210, Oct. 2000.
[22] I. Viola, A. Kanitsar, and E. Gröller, "Importance-Driven Volume Rendering," *Proc. IEEE Visualization,* pp. 139-145, 2004.
[23] L. Wang, Y. Zhao, K. Mueller, and A.E. Kaufman, "The Magic Volume Lens: An Interactive Focus + Context Technique for Volume Rendering," *Proc. IEEE Visualization,* pp. 367-374, 2005.

Nikolai A. Svakhine received the BS degree in computational mathematics from Moscow State University, Russia, and the MS and PhD degrees in computer science from Purdue University. He is currently employed by Adobe Systems Inc. His research interests include computer graphics and scientific visualization, in application to medical illustration and flow visualization.

David S. Ebert received the PhD degree from the Computer and Information Science Department, Ohio State University. He is a full professor in the School of Electrical and Computer Engineering, Purdue University. His research interests include scientific, medical, and information visualization, computer graphics; animation, procedural techniques, volume rendering, illustrative visualization, realistic rendering, procedural texturing, modeling, and modeling natural phenomena. He has served on the ACM SIGGRAPH Executive Committee and served as the editor-in-chief of the *IEEE Transactions on Visualization and Computer Graphics*.

William M. Andrews received the BA degree in art from the University of Texas, Austin, and the MA degree in biomedical communications from the University of Texas Health Science Center, Dallas. He is currently working toward the PhD degree in health promotion, education, and behavior at the University of South Carolina, Columbia. He is an assistant professor and education program coordinator at the Medical College of Georgia. He has been an active professional member of the Association of Medical Illustrators (AMI) for 25 years and has also served as president of the AMI and on the board of governors. His research interests include health education, visual perception, history of medical illustration, communication theory, and business practices. His teaching areas of interest include line and color illustration in both traditional and digital media, as well as visual problem solving, linear and nonlinear storytelling, communication theory, learning resource management, and business practices. He is a fellow of the AMI. He has been an editor of the *National Newsletter* and editor for the *Medical Illustration Source Book.* He has been a certified medical illustrator since 1993.