Visualization by Proxy: A Novel Framework for Deferred Interaction with Volume Data

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

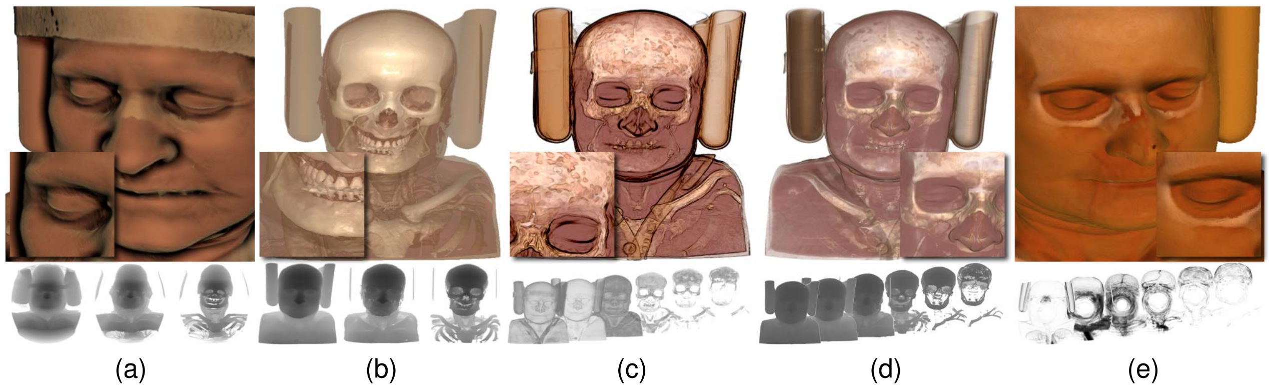


Fig. 1. Synthesized images (top) of a CT head data set, created using proxy images (bottom). (a) Using multi-perspective proxy images, we can visualize multiple layers of a 3D object (shown in (b)) in multiple orientations. We can also inexpensively simulate complex lighting models such as proxy-based ambient occlusion, while retaining the spatial relationships between semi-transparent layers. This is done without accessing volume data. (c) Accumulated attenuation proxy images allow us to explore data sets in transfer function space and to apply depth-aware enhancement, which highlights shape and alleviates depth ambiguities, often found in static images. (d) We can combine accumulated attenuation proxy images (shown in (c)) with depth proxy images (below) to support lighting effects. (g) Multi-perspective accumulated attenuation proxy images allow us to visualize a volumetric object in multiple orientations.

Abstract—Interactivity is key to exploration of volume data. Interactivity may be hindered due to many factors, e.g. large data size, high resolution or complexity of a data set, or an expensive rendering algorithm. We present a novel framework for visualizing volume data that enables interactive exploration using *proxy images*, without accessing the original 3D data. Data exploration using direct volume rendering requires multiple (often redundant) accesses to possibly large amounts of data. The notion of visualization by proxy relies on the ability to defer operations traditionally used for exploring 3D data to a more suitable intermediate representation for interaction - proxy images. Such operations include view changes, transfer function exploration, and relighting. While previous work has addressed specific interaction needs, we provide a complete solution that enables real-time interaction with large data sets and has low hardware and storage requirements.

Index Terms—Volume visualization, deferred interaction, image-based rendering, volume distortion camera.

1 Introduction

Interactivity provides enhanced perception of depth and adds realism to rendered images. Thus, it is essential for meaningful exploration of volume data sets. Exploration in 3D space usually implies exploration in both view and transfer function domains. Direct volume rendering provides these capabilities; however, it requires multiple (often redundant) accesses to the entire volume. Thus, interactivity may be adversely affected by large data size, high resolution or complexity of a data set, or an expensive rendering algorithm. This problem is exacerbated when the data is visualized remotely, as is common for in-situ and distance visualization, and rendering on low-end devices.

One solution to this problem is reducing the amount of data that has to be accessed to generate new images of a data set. Previous approaches store a collection of rendered "training" images instead of the volume itself. New results are synthesized by finding correspondences and combining the training images. For example, image-based rendering techniques enable view changes by storing a collection of images, rendered from different viewpoints. The lack of explicit 3D information, however, prevents the user from changing the color and opacity of the data depicted in the images. Exploration in transfer function space can be enabled by a similar approach: generating a collection of images, rendered with different transfer function settings. Then, transfer function changes can be simulated by synthesizing new images from the training set. Other previous work focuses on providing fast transfer function changes by caching view-dependent samples. In such approaches, 3D information is still necessary and the cached data is invalidated as the view changes.

In this paper, we describe a novel framework for interactive exploration of 3D data that unifies the representations for multiple types of interaction. The notion of visualization by proxy relies on the ability to defer operations, traditionally used to explore 3D volume data, to a more suitable intermediate representation for interaction - *proxy images*. Such operations include view changes, transfer function exploration, and relighting.

To achieve this, we represent a volume as a collection of proxy images. Each type of a proxy image contains different information, necessary for enabling a particular exploratory operation. Users can mix and match proxy images, based on the kind of exploratory operations they wish to perform. To enable rotation, we develop a novel *volume distortion camera model* for generating multi-perspective images. It

• *Anna Tikhonova, Carlos D. Correa, and Kwan-Liu Ma are with the University of California, Davis. E-mails: {tikhonov, correac, ma}@cs.ucdavis.edu.*

Manuscript received 31 March 2010; accepted 1 August 2010; posted online 24 October 2010; mailed on 16 October 2010.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

allows us to store information for different orientations of a 3D object in a singe image, as opposed to rendering multiple images from different viewpoints. We also present a novel image-based transfer function design algorithm. It employs inexpensive machine learning techniques to accurately predict color, based on varying distributions of attenuation. To enable exploration in transfer function space, instead of storing a set of training images with different combinations of transfer function parameters, we fit a model that predicts the final color for different combinations of accumulated attenuation. We generate proxy images containing accumulated attenuation, or distribution of attenuation along a ray with respect to intensity values. Depth proxy images enable interactive relighting and application of different shading methods, such as proxy-based ambient occlusion, which increases realism by taking into account the attenuation of light. When depth information is not available, we use the accumulated attenuation proxy images to apply a non-realistic shading method - feature enhancement. Although depth images have been used in the past to defer lighting and rendering operations on volumes, we have tackled the difficult task of extending them to multi-view perspective proxy images to perform rotation and relighting of an object for different orientations. We also combine multi-perspective proxy images with light attenuation models to synthesize new views of volume data from different orientations, with different lighting parameters, and with various opacity mappings.

Our goal is to provide users with a compact representation of 3D data and an interactive visualization system that can run on a standard desktop computer, mobile device, or within a browser. In this paper, we describe interactive exploration methods using proxy images that can be implemented on the GPU with relative ease. Through visual and quantitative evaluation, we show that our approach is an affordable and effective solution for preview and exploratory visualization.

2 Related Work

Due to the ubiquity and popularity of image processing, image-based rendering is a popular alternative for view synthesis, 3D geometry reconstruction, relighting, and multi-layer rendering [7]. A large body of research is devoted to the problem of view reconstruction from a single image and from multiple images. The most common approach involves unconventional cameras, such as the pushbroom camera [5] and the two-slit camera [19]. To alleviate disocclusion artifacts, the light field [9] and the lumigraph [4] use a 2D array of planar pinhole cameras. No disocclusion errors occur, but view synthesis for new orientations requires a large database of images. Other approaches include the multiple-center-of-projection cameras [23] and multiperspective rendering [32, 34]. McMillan and Bishop [17] describe a 3D warping approach using depth images. The scene is rendered from new viewpoints by splatting or by rendering a mesh connecting the warped samples. Mark et. al. propose to accelerate conventional rendering by warping two reference images [14]. When rendering individual objects rather than complex scenes, simple non-pinhole cameras seem to suffice, as demonstrated by the depth discontinuity occlusion camera by Popescu and Aliaga [21], the occlusion camera by Mei et. al. [18], and the general pinhole camera by Popescu et. al. [22]. Their work, however, assumes that an object can be represented as a surface model and that the intersections of that surface with the view ray can be uniquely defined. We present a new camera model, inspired by the occlusion camera and extended to volumetric objects. It enables the reconstruction and recomposition of multiple semi-transparent layers or intervals using a small number of 2D images.

Layer-based rendering techniques offer a different solution. They store samples occluded in the reference view in additional layers at each pixel. Shade et al. propose *layered depth images (LDI)*, whose pixels contain a list of color and depth values [27]. Depth values allow to display the appropriate parallax induced by camera motion. Multi-layered representations have been popularized in commercial rendering software to simulate complex materials on synthetic objects, such as skin and translucency [2, 3]. In volume rendering, layer-based representations have been used to defer operations such as lighting and classification. Ropinski et al. simulate the application of transfer functions as a combination of individual layers, each generated by an individual transfer function [26]. Rautek et al. use a similar multi-layer metaphor to combine different rendering styles [24]. Deferring the compositing operation allows them to apply non-photorealistic shading effects, difficult to obtain directly in the volumetric space. Wu and Qu use an optimization approach to synthesize images produced by a transfer function resembling the visual result of combining individual layers. [33]. Other types of layers have been obtained for classification purposes, such as the opacity [25] and feature peeling [13] approaches. In their case, layers represent different regions where attenuation is saturated. Therefore, by manipulating these layers, one can extract different features in a view-dependent manner. Unlike the previous approaches, peeling methods operate in 3D space.

Using layers to defer operations has been proven effective to cache results [11, 1, 8] or certain volumetric properties along the view rays [12], which can be later reused for efficient transfer function exploration. Srivastava et al. use a set of compressed runs of volume data, an extension of layer based data that enables variable sampling for fast transfer function exploration [29]. The use of compressed samples and layer-based methods is even more effective for the rendering of unstructured meshes, as demonstrated by Shareef et al. [28]. Tikhonova et al. propose a method for changing the color and opacity mappings of volume rendered images, based on alpha estimation of a given set of layers [30]. These methods, however, are view-dependent and require re-compression or re-layering for new orientations. In this paper, we provide a set of proxy representations that allows us to unify different operations on volume data, such as transfer function and view exploration. A more general image-based transfer function design, as opposed to data-based approaches [20] was proposed by He et al. and Mark et al [6, 15]. New views of a volume data set can be obtained from a gallery of images, containing representative samples from the space of combinations of opacity and color mappings. Both approaches allow users to find good transfer functions, based on the results of previous interactions. Most of these approaches, however, require a large number of images to be useful for exploration. This paper presents a novel approach, which produces proxy images, or compact intermediate representations of volume data. Proxy images can be used to inexpensively reconstruct volume rendered images from new views, with new transfer functions, and with new lighting parameters.

3 Framework for Volume Visualization by Proxy

The traditional image-based rendering approaches use a large number of volume rendered images to generate each new result. In our proposed framework, we convert volume data into an intermediate representation for interaction - proxy images. We defer the operations commonly used for interactive exploration of 3D data (view and transfer function domain exploration, and relighiting) to our intermediate representation. New images are then produced from proxy images, without accessing the original 3D data. We describe the different proxy representations below.

3.1 Depth

The simplest proxy image is *depth*. Depth images are commonly used for defering lighting computations, including complex non-photorealistic effects such as contours and silhouettes. It is possible to simulate the light bouncing off an isosurface using volume gradient as normal to that surface. For a single viewpoint, the normals can also be approximated from a depth image.

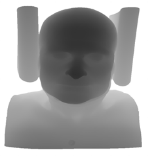


Fig. 2. Depth proxy image.

We extend the notion of depth proxy images to multiple layers (or isosurfaces) of a volume. Instead of a single depth map for an entire data set, we store depth images for several intensity intervals. This allows us to simulate semi-transparent rendering and lighting from different directions, emphasize spatial relationships between different features, and apply such complex operations as ambient occlusion with little computational cost. Fig. 2 shows an example of a depth proxy image, where grayscale represents the depth of a point with respect to the camera plane.

3.2 Multi-view Perspective

Depth images are computed for a single viewpoint, limiting the degree of interaction. We extend them to multi-view perspective proxy images, thus enabling viewpoint changes for more effective exploration.

The multi-view perspective proxy is an image produced by our novel volume distortion camera model. In addition to the samples visible from the reference view, we store samples that would only be visible from neighboring viewpoints - all in a single proxy image. Similarly to depth proxy images, we perform this operation for several layers. Fig. 3 shows an example of a multi-view perspective proxy image, which shows parts of an object (sides, top, and bottom) not visible in Fig. 2.

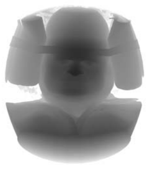


Fig. 3. Multi-view perspective proxy image.

3.3 Accumulated Attenuation

The proxy images defined above, require users to select particular isovalues for depth computation. In volume rendering, we construct images as the composition of semi-transparent samples, rather than discrete surfaces. Therefore, we present another type of a proxy image, which stores the accumulated attenuation for a number of intervals in the scalar field domain. As described in Section 6, we can decompose a volume-rendered image into multiple attenuation images, each of them containing the total contribution to attenuation for each intensity interval. We discovered that the color resulting from a change in opacity mapping can be represented as a linear combination of attenuation. These linear parameters can be estimated using machine learning techniques. Fig. 4 shows an example set of attenuation proxy images, which appear as semi-transparent layers. When combined, they reconstruct a volume-rendered image that can be explored in transfer function space.

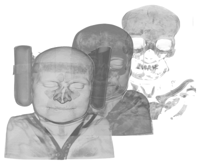


Fig. 4. Accumulated attenuation proxy image.

4 View Exploration with Multi-perspective Proxies

Previous approaches to view synthesis interpolate multiple images, obtained from different viewpoints. Such approaches suffer from the diminishing returns associated with rendering each additional image and the number of non-overlapping samples. In our approach, we comprise the information required to reconstruct multiple views into a single proxy image. This is achieved in a pre-processing stage, using a novel volume distortion camera model. At a later stage, we allow interactive rotation of a volume data set with a fast view synthesis algorithm. The algorithm is general and applies to both layer-based and attenuation-based proxy images. These stages are described below.

4.1 Volume Distortion Camera Model

The volume distortion camera is a non-pinhole single-pole camera with 3D radially distorted rays. In a traditional pinhole camera, an image is obtained by compositing samples along rays emanating from the eye position towards the image plane and into the volume. In our new camera model, we first define a single polar point at the center of an image, which defines a pole line in the view direction, to which all rays converge. We define a ray from the camera plane towards the pole line as follows. For an image pixel, we define a ray starting at a pixel position on the camera plane $p_{im} = (x_p, y_p, z_p)$ and ending at $E(p_{im})$:

$$E(p_{im}) = p_e + v(|v^\top (p_{im} - p_e)| + d(x_p, y_p)), \tag{1}$$

where v is the normalized view direction, p_e is the eye position, and $d(x,y) = ||(x,y) - (x_0,y_0)||$ is the distance from a pixel to the screen center (x_0,y_0). This generates radially distorted rays, converging at the pole, typically the screen center. This is depicted in Fig. 5 and is similar to the occlusion camera, defined in [18]. In this paper, we define a novel pipeline for the generation of distorted images and subsequent view synthesis that applies to both layer-based and volumetric semi-transparent objects.

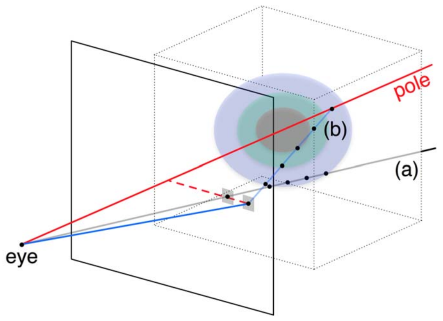


Fig. 5. Volume distortion vs. traditional pinhole camera model. (a) In a traditional pinhole camera, rays are cast from the eye position towards the image plane and into the volume. (b) In our camera, distorted rays are cast from the image plane toward the pole line. We retain the distortion amount and depth as proxy images.

4.2 Generation of Multi-perspective Proxies

We obtain a multi-perspective object representation using our camera model. Instead of compositing samples belonging to different isosurfaces into a single pixel, we store several multi-perspective proxy images, corresponding to different isosurfaces. In addition to storing depth, we store the distortion at each intersection point. The distortion is found by computing the difference between the projected position of a point in a pinhole camera (or orthographic projection) and the actual point in the image plane. Let us define q as the 3D intersection point between a multi-perspective ray, emanating from p_{im} and a given isosurface. The projection of this point in a single perspective camera is q_{im}. Therefore, the distortion ds for this point is:

$$ds = ||q_{im} - p_{im}||. \tag{2}$$

Note that a single value suffices, since we know that distortion is always applied radially towards the pole.

4.2.1 Fast Distortion on the GPU

We implement the generation of multi-perspective proxy images entirely on the GPU. Unlike surface models, volume data does not inherently possess any shape. Therefore, explicit distortion approaches, such as the one proposed by in [18], do not apply to volumes. Instead, we exploit vertex shader capabilities to deform the bounding box of a volume. Then, we walk through a volume in the fragment shader, using radially distorted rays. Whenever we intersect an isosurface of interest, we stop the traversal and output both the depth and distortion of that point. The amount of distortion depends on the distance between the ray start position and the pole. Points further away from the pole are distorted more than the points closer to the camera plane.

4.3 View Synthesis

We synthesize different views of a volume from multi-perspective proxy images using undistortion. It extracts the 3D position of each pixel in a proxy image and then re-projects it to a new 3D point. This can be efficiently accomplished on the GPU, using vertex and fragment shaders. For a given pixel in the multi-perspective proxy image (x_d, y_d), we obtain the undistorted camera position (x_{im}, y_{im}, z_{im}) as:

$$(x_{im}, y_{im}) = (x_d, y_d) - \frac{(x_d, y_d) - (x_0, y_0)}{||(x_d, y_d) - (x_0, y_0)||} * ds. \tag{3}$$

$z_{im} = Z(x_d, y_d)$ is the depth stored in the proxy image. The point's new position is found by concatenating the inverse matrix from the original viewpoint, and re-projecting back to the new viewpoint:

$$(x_{new}, y_{new}, z_{new}) = M_{new} M_{orig}^{-1} (x_{im}, y_{im}, z_{im}). \tag{4}$$

For a set of layer-based proxy images, this is done efficiently in the vertex shader. Once we synthesize an image for a new viewpoint, we

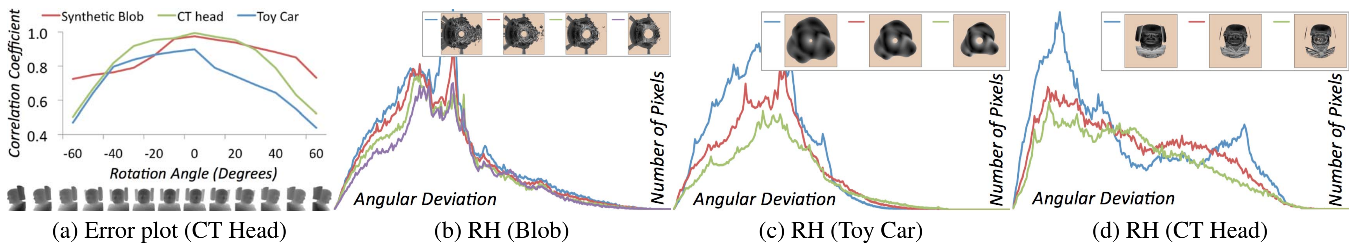


Fig. 6. Accuracy of view synthesis for three data sets. (a) Error plot as a function of angular deviation. (b-d). Radiality histograms (RH) as a metric for predicting the quality of radial distortion. The actual amount of angular deviation for each feature is shown as an image (top).

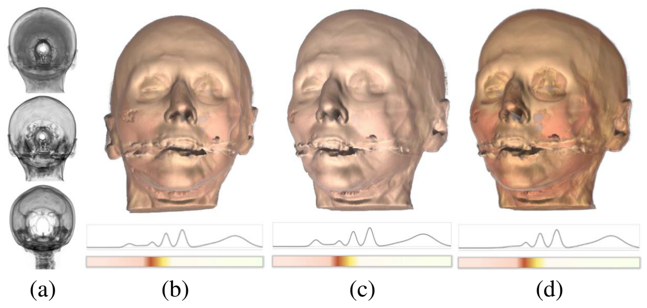


Fig. 7. Accumulated attenuation and distortion. (a) Accumulated attenuation proxy images for skin, muscle, and bone, generated using a multi-perspective camera. (b-d) New views, synthesized using volumetric undistortion. Modifying the opacity mapping, we create novel views of the bone and muscle layers.

output the new depth, which we subsequently use to apply lighting. Several examples are shown in Fig. 1(a-b), where we show a number of proxy images (bottom) and the synthesized results (top), simulating different lighting parameters. The single-view depth proxy images, shown in Fig. 1(b), are used to simulate diffuse lighting with specular highlights. The multi-perspective proxy images, shown in Fig. 1(a), are used to synthesize new views of a CT head from a different orientation and to simulate proxy-based ambient occlusion at low cost. An interactive example is shown in the accompanying video.

4.3.1 Volumetric View Synthesis

In many cases, it is desired to preserve the volumetric nature of a data set (instead of a collection of overlapping layers) in the generated images. We adapt the multi-perspective camera to handle volumetric attenuation as follows. Instead of storing depth for a number of isosurfaces, we store the accumulated attenuation along the distorted rays for a number of intensity intervals, as described in Section 6. To synthesize new views, we perform *virtual* ray marching into a the bounding box of a volume, along points $p = s + v't$, where s is the ray start position (typically, intersection between the view direction and the bounding box), v' is the new view direction for the synthesized orientation, and t is a parameter along the ray. While marching, we keep track of the effective distortion, computed as the distance: $d_{3D} = ||\Pi^\top(p-s)||$, where Π is the normal of the camera plane of the original distorted view. We use the projection $p_{im} = \Pi^\top p$ to find the stored distortion in the proxy image. If the current distortion is greater than the stored distortion, i.e. $d_{3D} > d(p_{im})$, we use the attenuation stored in the proxy image and move to the next interval; otherwise, we continue marching. This virtual marching is relatively inexpensive and effective for synthesizing new views of volumetric objects, without storing large 3D textures in system or graphics memory. Fig. 7 shows the synthesis of new views for a CT head data set and the change in opacity for one intensity interval.

4.4 Evaluation

Figure 6(a) evaluates the accuracy of view synthesis as a function of view rotation. We plot the correlation between the depth images generated from 3D data versus the depth images reconstructed from multi-view perspective proxy images for multiple angles. The angles range from 0 to 60 degrees away from the original view. The figure shows that the reconstruction for the original view is almost perfect (the correlation coefficient is almost 1.0), and, as the angle increases, there is a non-linear loss in accuracy. The synthetic blob data set is asymmetric (while the CT head is symmetric). Thus, we observe a higher loss of accuracy for the side that is less smooth. The car data set is another example of an asymmetric data set (front vs. back). It also contains many turbulent structures. Thus, the reconstruction accuracy is lower.

In Fig. 6(b-d), we demonstrate a new metric for measuring the expected quality of view synthesis, *radiality histograms*, for several data sets. The quality of reconstruction depends on the frequency of high curvature, i.e. it deteriorates when the normal at the intersection of a distorted ray and an isosurface deviates significantly from the direction of a distorted ray. Radiality histograms measure the amount of this deviation for the entire image. If most deviation is in the low end of the spectrum, such as in Figure 6(b-c), the expected error of rotation is low. If a histogram is skewed towards the high end of the spectrum, such as in Fig. 6(d), the expected error of rotation is high.

5 Relighting Using Depth Proxy Images

Depth images can be used to simulate lighting effects. We can approximate the normals of a surface from depth, up to an unknown scaling factor, as $n(x,y) = (\frac{\partial Z}{\partial x}, \frac{\partial Z}{\partial y}, 1)$, where $Z(x,y)$ is the depth at that point. We can use these normals to apply local illumination models as well as proxy-based ambient occlusion. Fig. 1 shows a number of relighting examples for a CT head data set.

Single-perspective vs. Multi-perspective Relighting. Unlike previous normal-from-depth methods, we extend this idea to multi-perspective cameras. In general, the normal computed from distorted depth undergoes a non-linear transformation. It is given by the Jacobian of the distortion, a matrix of partial derivatives of the distortion with respect to depth, $n'(x,y) = \mathrm{J}_D n(x,y)$, where J_D is the Jacobian of the distortion. Although this method retains 3D information implicitly encoded in the distorted image, the computation of the Jacobian matrix requires extra per-pixel operations. As an alternative, we compute the normal in two steps. First, we synthesize a depth image for the new view, as described in the previous section. Then, we compute lighting from the newly synthesized depth (an example is shown in Fig. 1(a)). Fig. 8 shows the application of lighting parameters to a volumetric blob data set, including specular highlights and proxy-based ambient occlusion, while synthesizing new views.

Proxy-based Ambient Occlusion. Ambient occlusion is an approximation of full global illumination. It approximates the occlusion of a point from a sampling of the neighboring space. For 3D volumes, this is generally expensive without simplifications or the use of complex data structures. Luft et al. [10] propose an image-based method, that uses unsharp masking of a depth image to simulate similar effects. This method can be efficiently implemented using convolution; however, the use of isotropic smoothing often results in exaggerated shading at objects' edges. With depth images, we can improve the

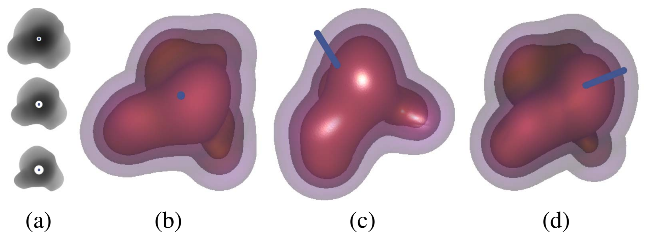


Fig. 8. Relighting using multi-perspective proxy images for synthetic blob dataset. (a) Multi-perspective proxy images. (b) Diffuse; (c) Diffuse with specular highlights. (d) Proxy-based ambient occlusion. Results are shown for three different views, synthesizing rotations of more than 45 degrees.

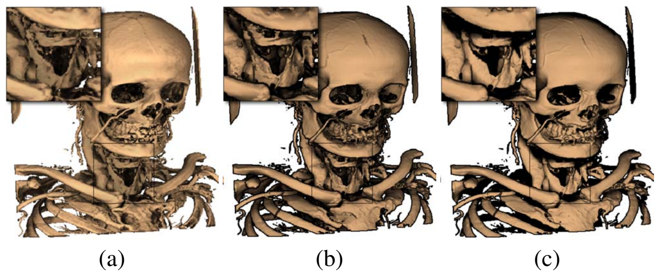


Fig. 9. Comparison of ambient occlusion methods. (a) 3D ambient occlusion, applied to an isosurface. (b) Our method, based on depth images and anisotropic kernels. (c) Image-based method that uses unsharp masking of the depth buffer [10]. Our method is better at retaining 3D information, inferred from depth images.

results using an anisotropic kernel. The purpose of this kernel is to approximate the neighborhood of a pixel as the projection of a hemisphere, defined by the normal at that point. For a given point p, the ambient occlusion is measured as:

$$AO(p) = \sum_{q \in Neigh(p)} \lfloor Z(p) + \frac{\partial Z}{\partial p}^{\top}(q-p) - Z(q) \rfloor_0, \qquad (5)$$

where q denotes a point in a neighborhood of p, $Z(p)$ is the depth at point p, and $\partial Z/\partial p$ is the 2D spatial derivative of depth. $\lfloor \cdot \rfloor_0$ is a clamping operation, that keeps the value greater than or equal to 0. Results are shown in Fig. 9. We compare the ground truth 3D ambient occlusion (a) with our approach (b), and the result obtained with unsharp masking (c). Our approach is better at retaining 3D information inferred from depth images, as shown in regions below the chin and collar bone.

6 Transfer Function Design Using Attenuation Proxy Images

The approaches described in previous sections treat a volume as a collection of semi-transparent isosurfaces, which can be combined for multiple viewpoints and under different lighting conditions. For certain applications, such as flow visualization, it may be more informative to produce images via semi-transparent volume rendering. Thus, it is critical to be able to manipulate the opacity values assigned to specific intervals to bring out different features and structures in the data. Previous solutions store a large collection of volume rendered images, that encode different combinations of layers in a volume, and simulate transfer function changes by synthesizing new images from the available set. In our framework, the same capability is provided using only two sets of proxy images. The first one consists of a collection of the accumulated attenuation proxy images for a finite number of intensity intervals. The second proxy is a compact model, that maps the most salient attenuation distributions to color. We use this model to synthesize new images of a volume with various opacity mappings.

6.1 Accumulated Attenuation

According to the volume rendering integral [16], the color resulting from compositing volume data is:

$$C = \int_0^D C(t)T(t)dt \text{ , with } T(t) = \tau(t)e^{-\int_0^t \tau(s)ds}, \qquad (6)$$

where $C(t)$ is the radiance or color and $T(t)$ is the total attenuation of a sample t along the view direction, defined in terms of the attenuation coefficient $\tau(t)$, typically an opacity mapping.

Subdividing the intensity domain into a finite number of intervals, we obtain a distribution of attenuation for each pixel in the image. For a given interval (l,h) in the intensity domain, the per-ray accumulated attenuation is the combined contribution of all samples with an intensity that falls within that interval:

$$a(l,h) \approx \sum_{i=0, l<V(i)\leq h}^{M} \alpha(i) \prod_{j=0}^{i}(1-\alpha(j)), \qquad (7)$$

where $\alpha(i)$ is the opacity of a sample along the ray, $V(i)$ is the scalar intensity value at that point, and M is the number of samples along the ray.

This approximation becomes more useful as we increase the number of intensity intervals. In the worst case, each interval represents a single intensity value. However, we show that even a small number of intervals, say 16-32, allows us to defer the composition operation for exploration in transfer function space. In our previous work [31], we propose a similar method. We re-composite the attenuation distributions (instead of the original samples) to synthesize new images, using different opacity and color mappings. However, this approach is limited to volumes containing nested isosurfaces with few overlapping structures. It also assumes a monotonically increasing relationship between the opacity and intensity mappings. In this paper, we propose a different compositing method, that produces superior results and can be applied to arbitrary intensity distributions. This novel method uses inexpensive machine learning techniques to predict color, based on the distribution of attenuation, as described in the following section.

6.2 Attenuation Clustering and Model Learning

In Eq.6, the final color of a volume rendered image is a combination of the attenuation of all samples along the ray. Therefore, one can think of the volume rendering integral as a mapping from attenuation distributions to color. To simplify our discussion, let us define the color of a pixel as a tuple $c_{3\times1}$ and the attenuation distribution as a tuple $a_{L\times1}$, where L is the number of intervals in the intensity domain. Then, the volume rendering integral can be defined as a function $f: R^L \mapsto R^3$, so that $c = f(a)$. Using its first order Taylor expansion, we can approximate this equation as:

$$c \approx f(a_0) + \mathsf{J}_f(a-a_0) = c_0 + \mathsf{J}_f(a-a_0) \qquad (8)$$

for a given attenuation distribution a_0, corresponding to color c_0, and where J_f is the Jacobian of the transformation, a $3\times L$ matrix containing the partial derivatives of the attenuation distribution with respect to the color components. We use RGB color space, but different color spaces may be used as well. This equation is a general mechanism for predicting color c, corresponding to any given attenuation distribution a, as a linear variation from a representative color-attenuation tuple (c_0, a_0) (as long as we can find good estimates for c_0, a_0, and J_f). If we represent all the attenuations in a set of images with a few key colors and attenuation distributions, the remaining combinations can be approximated using this linear approximation. The quality inherently depends on the number of representative colors and attenuation distributions. To find a good set of representative colors and attenuations, we employ a clustering approach. Our process is shown in Fig. 11. We now describe it in a series of steps:

1. Store the attenuation and color for each pixel in a small set of input volume rendered images (typically three or four).

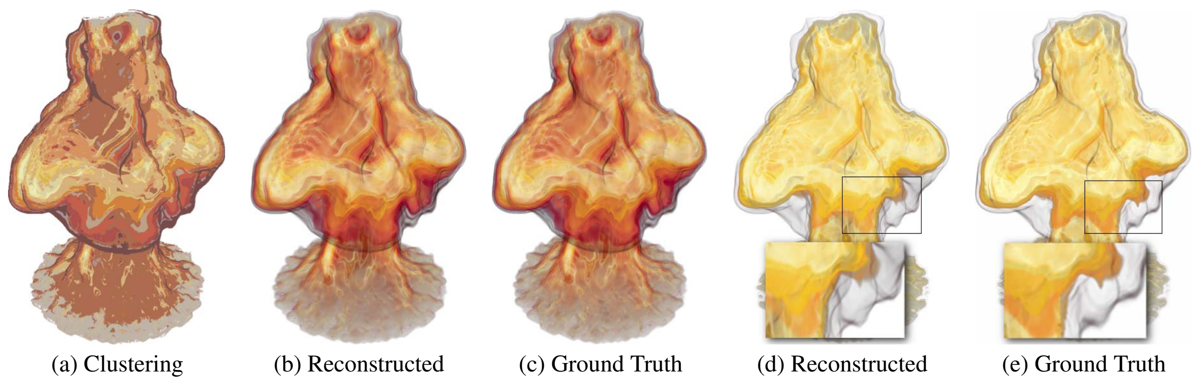


Fig. 10. Transfer function design based on attenuation clustering for a fire simulation of a methane pool. (a) Extracted clusters for a given transfer function. Color is based on the color of each attenuation centroid. (b) Image reconstructed from our prediction model. (c) Ground-truth volume rendered image. (d) Synthesized image with modified opacity, that brings out inner structures. Compare to (e), the ground truth image for that opacity setting. Our approach correctly attenuates color in regions where red isosurfaces are no longer present.

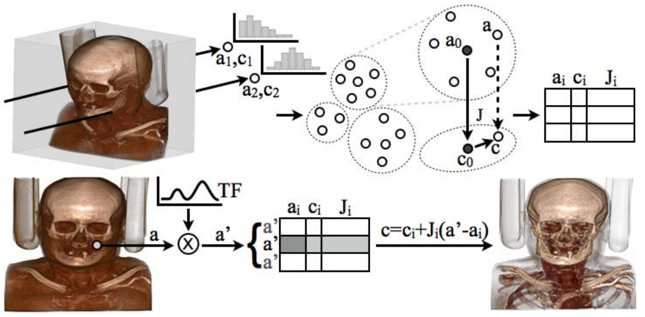


Fig. 11. Transfer function exploration using model learning. Pre-processing stage (top): we compute attenuation distributions a for each pixel and cluster them into groups. For each cluster, we find the linear projection J that relates color and attenuation and store these values in a table. Interactive stage (bottom): the user explores different transfer functions, which modify the attenuation distributions. We search the table for the closest distribution to the new attenuation distribution a' and use the linear model to find the corresponding color.

2. Cluster the attenuation distributions based on Euclidean distance or another suitable metric. In our work, we apply the k-means algorithm on the distributions. Each of the resulting clusters has a corresponding representative color from an input image, defined as the color corresponding to the cluster's centroid. Fig. 10(a) shows an example clustering result for the rendering of a fire simulation, using just $K = 24$ centroids. The colors indicate the color associated with the cluster's centroid.

3. For each cluster, we fit a linear model to find the $3 \times L$ parameters of the transformation J_f. Assuming that each color component is independent, we find it by solving the over-constrained problem for each column $i \in 1,2,3$ of J_f:

$$\begin{bmatrix} a_1 - a_0 \\ \cdots \\ a_n - a_0 \end{bmatrix} \mathsf{J}_f^{(i)} = \begin{bmatrix} c_1 - c_0 \\ \cdots \\ c_n - c_0 \end{bmatrix} \tag{9}$$

for n pixels in the training image, where c_0, a_0 are the color and attenuation of a centroid. Since we use a small number of clusters ($K = 24$) and a small number of attenuation intervals ($L = 16$), this operation is inexpensive.

4. The result is stored as a look-up table of size $((3+1)L+3) \times K$, consisting of a $3 \times L$ Jacobian matrix (J), the centroid distribution a_i, and color c_i for each cluster $i, i \in \{1,2,\ldots,K\}$. In our experiments, we use a clustering of $K = 24$ clusters for $L = 16$ intervals. The resulting transfer function proxy is of size 67×24.

Fig. 10(b) demonstrates the clustering results for a fire simulation. From a few clusters (24), we can represent the colors of volume rendering based on the variation of accumulated attenuation with respect to each cluster's centroid. Compare to the ground truth volume rendered image in 10(c) (which requires access to 3D data). As can be seen, the difference is very small.

6.3 Opacity Changes

The above approach allows us to simulate opacity changes using only proxy images. There are three steps involved in this operation:

1. We approximate the resulting accumulated attenuation a' after an opacity change. This can be accomplished by estimating the change in attenuation, based on a change in opacity for a given intensity interval. Then, we propagate attenuation to the rest of the intervals, using the approach in [31].

2. We find the nearest cluster centroid to the resulting attenuation distribution. This search gives us the centroid color c_k, attenuation distribution a_k and Jacobian $\mathsf{J}_f^{(k)}$, so that :

$$||a_k - a'|| < ||a_i - a'|| \forall i \in \{1,2,\ldots,K\}. \tag{10}$$

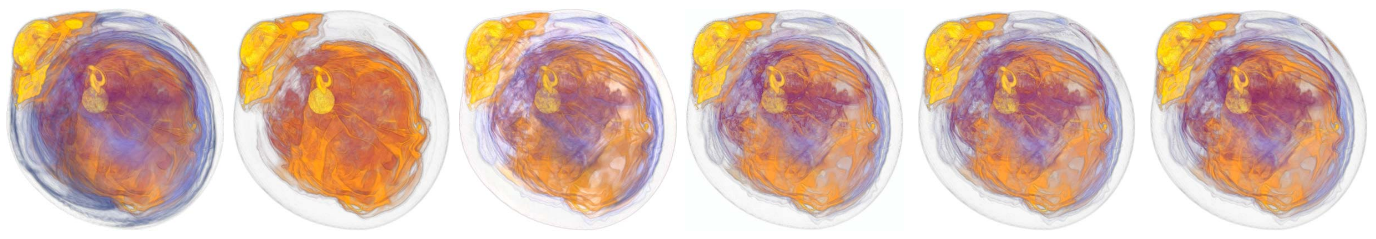
3. Finally, we use Eq.8 to find the resulting color c' that corresponds to the new accumulated attenuation.

Fig. 10 shows the result for the fire simulation after simulating an opacity change. In this case, we simulate an opacity change that removes several isosurfaces (in red) to bring out the regions of high temperature (yellow and orange). Notice that we can almost exactly reproduce the volume rendered ground truth result, shown in Fig. 10(e). The resulting color is accurately reproduced, even for the regions that were previously occluded by the red features.

Another example is shown in Fig. 12. We use two training images of a supernova entropy simulation (a-b) and show the reconstruction of an opacity function not present in the training set, while increasing the number of clusters. We observe that a single cluster still shows the main distribution of colors as a single linear fit, but it cannot capture the correct compositing (c). Adding a few images might result in color quantization, which is undesirable (d). For this example, 48 clusters capture the color distribution accurately (d), as compared to the ground truth image (e).

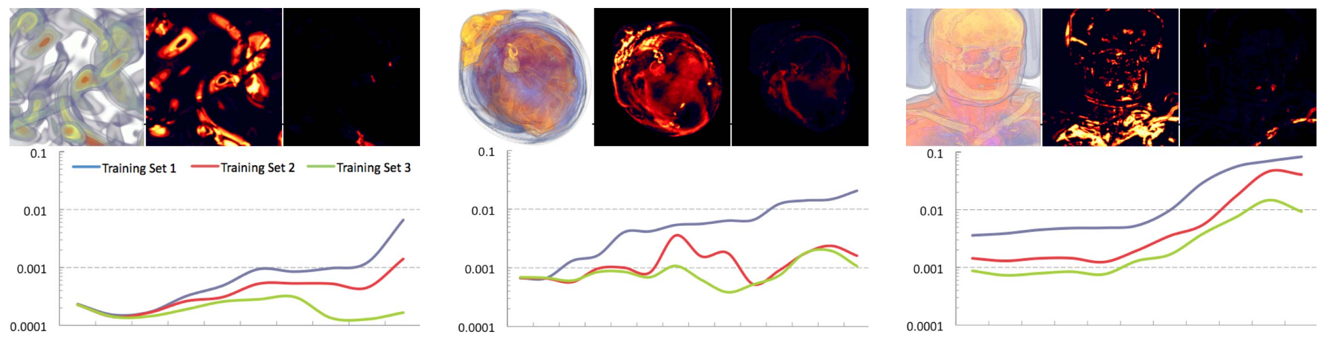
6.4 Evaluation

To understand the sources of error in our approximation, we measure the accuracy of reconstruction as pixel-wise difference between the reconstructed and ground truth images. Fig. 13 summarizes the results as error plots for three representative data sets: smooth vorticity, turbulent supernova flow simulation, and CT head. Each image shows



(a) Training Image (b) Training Image (c) Reconstructed K=1 (d) Reconstructed K=8 (e) Reconstructed K=48 (f) Ground Truth

Fig. 12. Transfer function exploration of the entropy field of a supernova simulation. (a-b) Training images. (c-e) Predicted result for a different attenuation distribution for $K = 1, 8$, and 48 clusters. A single cluster misrepresents the compositing of different intervals. The result obtained using 8 clusters is better, but quantizes some colors due to suboptimal clustering. Using 48 clusters, results in an accurate prediction, as compared to the ground truth (f).



(a) Smooth Vorticity (b) Turbulent Flow (c) CT Scan

Fig. 13. Accuracy of prediction model from attenuation. Each plot shows the sum of squared pixel differences (SSD) between the predicted and the ground truth images for several types of data sets using three training sets. On top of each plot, we show one of the training images and the accumulated error using a radiance heat map for training set 1 (middle) and training set 3 (right). (a) For smooth vorticity, the error is low. In some cases, adding more training images results in an order of magnitude improvement. (b) The improvement is not as dramatic for a turbulent flow. (c) A different pattern occurs for CT data sets, due to different attenuation distributions.

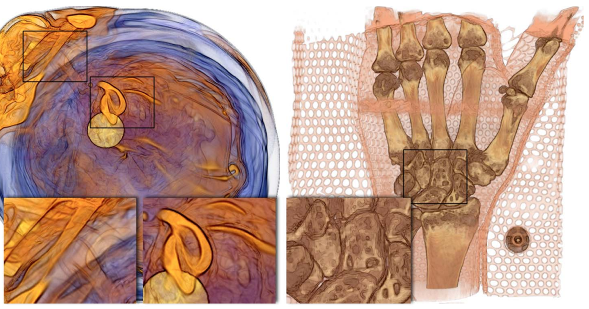
the pixel error as the sum of square differences (SSD) on a logarithmic scale. The x axis represents different observations, where we progressively modify the opacity function to make inner structures more visible. We perform this experiment for three training sets, with $1, 2$, and 3 training images, respectively. We observe a different pattern for flow and CT data sets. For flow, the error seems to diverge and improve dramatically as we increase the number of training images. For smooth vorticity, each added image results in an order of magnitude of improvement. The behavior is not as obvious for the supernova flow simulation, due to the presence of turbulent structures. In some cases, adding more images seems to decrease accuracy, when the function cannot be approximated linearly, and the linear fit only increases the amount of error. For the CT head, however, we see a rather uniform change. This suggests a different attenuation distribution. Each figure includes error images, obtained as the accumulation of pixel differences for all test images, using a radiant heat color map (black indicates less error, yellow to white - more error). We observe that for a small number of training images, the error is quite large and accumulates in regions of overlap (middle image), while for training set 3 (right), the error becomes almost imperceptible.

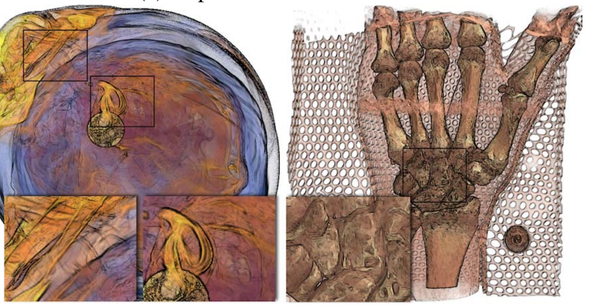
6.5 Depth-aware Enhancement

A close inspection of our color prediction model reveals that it cannot correctly predict colors if there is a lighting model, since lighting changes colors in ways not related to attenuation. Lighting is still desired to enhance shape. We can simulate lighting by combining accumulated attenuation proxy images with depth proxy images, generated for each interval (example shown in Fig. 1(d)), at the cost of increasing the number of proxy images. Another option is to approximate the effects of lighting by enhancing the salient regions of a volume, based on their gradient. Although a non-realistic shading method, this effect highlights shapes that cannot be seen even with lighting models. Moreover, this enhancement can be applied inexpensively as a modulation of color, based on the gradient of the attenuation distribution. Because attenuation *implicitly* encodes depth, the result is a depth-aware enhancement of volumetric edges. To achieve this, we apply an enhancement factor ρ to the output color, defined as the Euclidean norm of the spatial derivatives of the attenuation distribution:

$$c_{enhanced} = (1-\rho)c \text{ , where } \rho = \lambda \left(\sum_{i=0}^{L} || \frac{\partial a_i}{\partial x}, \frac{\partial a_i}{\partial y} ||^2 \right)^{\gamma}, \quad (11)$$

where L is the number of intensity intervals, a_i is the accumulated attenuation for the interval i, $|| \cdot ||$ is the Euclidean norm of a vector, and λ and γ are constants, controlling the strength and contrast of the enhancement, respectively. Fig. 14 shows two examples of enhancement for a supernova simulation and a CT wrist data set. For the supernova dataset, we are interested in visualizing inner core forces. Thus, we render the isosurfaces of interest with low opacity, which makes it difficult to identify certain regions, even with 3D lighting. The enhancement is able to bring out some of the structures, for example, the jet of entropy from the core to the outer layers (inset on left). The same jet is difficult to see with lighting. It is also incorrectly highlighted, if we simply enhance the edges of the resulting image. Using image-based edge enhancement, although similar to our technique, does not preserve the depth relationships between nested structures and is, therefore, misleading. For the wrist CT data set, enhancement proves effective for diagnosing illnesses. In this case, radiologists are searching for erosions in the bone, as a symptom of cancer. Some of the superficial erosions, indicators of early symptoms, are difficult to see with lighting. Image-based edge enhancement, on the other hand, fuses bone edges together and highlights noisy structures. Our depth-aware enhancement highlights the erosions correctly. We observe that, for certain tasks, enhancement based on attenuation is equally as or even more effective than more expensive lighting computations.


(a) Depth-aware Enhancement


(b) Image-based Edge Enhancement

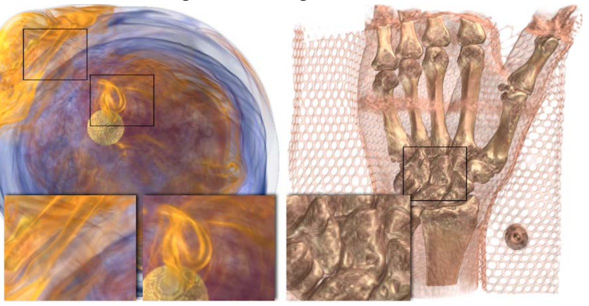

(c) Volumetric Lighting

Fig. 14. Enhancement examples for supernova simulation and wrist CT data sets. For the supernova data set, fine structures cannot be seen even with lighting, while image-based edge enhancement shows incorrect depth relationships between layers (one structure appears incorrectly behind the core). For the wrist data set, only highlighting shows fine erosions in the bone, indicator of the presence of a carcinogenic.

7 Discussion and Limitations

In this paper, we show a number of results that can be obtained with our framework. We evaluate the accuracy of our approach via a visual comparison and a quantitative analysis, but an utility assessment requires a full deployment of our framework in a working system. We believe this deployment can be performed with relative ease. Our approach, except for pre-processing of attenuation, is fully implemented on the GPU, using vertex and fragment shaders. Since most operations in our framework are performed on 2D images, they are usually faster than volume rendering. The performance of our methods depends on several factors: N, the dimensions of a volume, M, the dimensions of a 2D image, and k, the number of layers. Fig. 15(a) compares the space requirements for traditional and proxy-based rendering. We compare storage requirements for multiple volume sizes and for different numbers and sizes of proxy images. In the worst case, assuming that the number of layers grows linearly with N (say a tenth of the dimensions of a volume), using proxies is still more efficient than the original volume data. This is depicted in Fig. 15(b), where we show the relative cost of these metrics as we increase N and k, while keeping M constant (512^2). For small volumes, naturally, performing complex operations on a 2D image may be as costly as performing them in 3D space. However, as the size of data sets increases, we expect a more significant benefit. In most cases, as demonstrated in this paper, k can be kept constant, as a small number of layers, typically 16.

Our framework offers unprecedented capabilities for synthesizing new views of volume data for different orientations, opacity mappings, and relighting parameters. We now discuss some of the design aspects and limitations of our approach.

Approximation. Images, synthesized using our approach, are an approximation. We envision our framework used as an exploratory technique, mainly for interactively selecting views and transfer functions, useful for identifying or discovering patterns of interest. When the user is satisfied with the chosen parameters, the system can produce an accurate volume rendered image. However, some tasks, that take advantage of volume rendering, do not depend on a faithful representation of data. For example, diagnoses of illnesses often use judgements about the relative size of features. In many cases, this implies data filtering, such as filtering out unimportant or noisy structures.

Selection of Proxy Images. Although desirable, fully automated selection of layers and attenuation intervals is, in general, not possible. However, layer-based lighting and view synthesis are independent of the particular strategy used in selecting representative isovalues. In general, this selection is user-defined and task-dependent, akin to selecting a good transfer function. In this paper, the depth layers correspond to opacity peaks in a user-defined opacity function and the intensity domain is subdivided into uniform ranges. We plan to experiment with adaptive interval sizes, which will allow us to take into account the actual distribution of data along each ray. There is also a trade-off between the quality and amount of detail that can be encoded in the accumulated attenuation proxy images and the size of the volume representation. Our results demonstrate that we can achieve acceptable results even with a relatively small number of intervals. The number of layers is also application-dependent, but a general strategy is to select as many layers as needed to highlight the features of interest. Because users can explore the data in transfer function space, the opacity of less important layers can be reduced or they can be removed entirely. Thus, pessimistic strategies, where more layers are computed than needed, do not pose a limitation to our approach, except for the cost associated with storing extra images.

Single Pole Distortion. Our volumetric multi-perspective camera is restricted to single pole distortion. This prevents the users from exploring certain complex data sets with inter-occlusion between multiple features. We plan to extend our approach to multi-pole distortion. Similar volumetric reconstructions, although costlier, can be provided.

Rotation Error. We limit view synthesis to a single distortion image, thus there is considerable error in the results for large angles of rotation. As mentioned in Section 4.4, the error is due to curvature. We evaluate the effects of curvature in Fig. 6. We believe the error can be alleviated by incorporating additional multi-perspective images, generated from different viewpoints, into the view synthesis algorithm.

Number of Training Images. Our color prediction model uses training images to learn the relationship between attenuation and color. Combinations not present in the training images may be underrepresented and poorly predicted; for example, when a feature is completely occluded. Therefore, we need to enhance our approach with a smart (instead of ad hoc) selection of training images.

Linear Relationship Assumption. Another limitation is that the linear relationship applies correctly only to accumulated attenuation. Therefore, we cannot use the same model for different compositing strategies, such as additive blending or maximum intensity projection. The development of these models is currently under study.

8 Conclusion

We introduce a novel framework for visualizing volume data that enables deferred interactive exploration, without accessing the original 3D data. We unify a number of operations, including transfer function design, relighting, and viewpoint exploration, as a set of operations on compact representations, called proxy images. For the purpose of data exploration, perceptual and technological limitations make it difficult to provide interactivity, while requiring full access to volume data. We believe that our framework is the first complete solution to the prob-

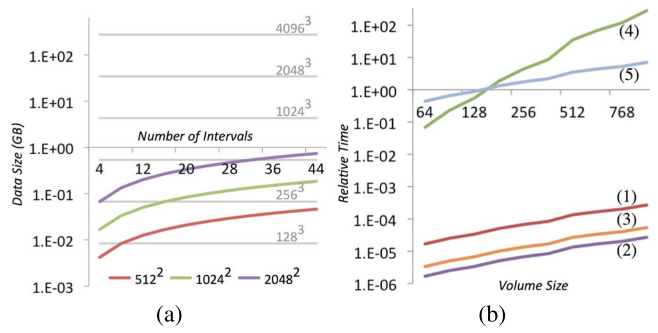


Fig. 15. Cost Evaluation. (a) Space requirements for traditional and proxy-based rendering. We compare storage requirements for volume sizes 128^3-4096^3 versus storing k (k range: 4-44) proxy images of sizes 512^2, 1024^2, and 2048_2. (b) Cost comparison for (1) 3D volume rendering, (2) proxy-based compositing, (3) multi-perspective compositing, (4) 3D ambient occlusion, (5) proxy-based ambient occlusion. This is a pessimistic case, where k is $\frac{1}{10}\times$ volume size and proxy image size is constant (512^2). Typically, k is constant, as in (a).

lem of deferring interaction via more compact intermediate representations. It can be efficiently deployed in remote settings, where the server computes proxy representations, while thin clients reproduce complex rendering effects at low cost. It is also an attractive solution to in-situ visualization, due to the low cost of computing proxies. Their generation can be interleaved with the simulation calculations, without a significant amount of additional computation cost. Some complex operations, such as multi-perspective distortion and attenuation clustering, can be computed in real-time, using modern GPUs. We envision that our framework for visualization by proxy can be extended for other types of operations or domains beyond volume rendering.

ACKNOWLEDGMENTS

This research was supported in part by the U.S. National Science Foundation through grants OCI-0749227, OCI-0749217, OCI-00905008, OCI-0850566, and OCI-0325934, and the U.S. Department of Energy through the SciDAC program with Agreement No. DE-FC02-06ER25777 and DE-FG02-08ER54956. Data sets courtesy of John Blondin of North Carolina State University, Philip Smith of University of Utah, Deborah Silver of Rutgers University, Abhijit J. Chaudhari and Ramsey D. Badawi of UC Davis School of Medicine, Kenji Ono of Riken Laboratories, Japan, Osirix Foundation, and University of North Carolina at Chapell Hill. I would like to thank James Shearer for providing his OpenGL harness code and debugging help and Hongfeng Yu for providing his excellent volume renderer.

REFERENCES

[1] J. Choi and Y. Shin. Efficient image-based rendering of volume data. In *Proc. of Pacific Conference on Computer Graphics and Applications*, page 70, 1998.

[2] C. Donner and H. W. Jensen. Light diffusion in multi-layered translucent materials. In *Proc. of SIGGRAPH*, pages 1032–1039, 2005.

[3] C. Donner, T. Weyrich, E. d'Eon, R. Ramamoorthi, and S. Rusinkiewicz. A layered, heterogeneous reflectance model for acquiring and rendering human skin. In *Proc. of SIGGRAPH Asia*, pages 1–12, 2008.

[4] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In *Proc. of SIGGRAPH*, pages 43–54, 1996.

[5] R. Gupta and R. I. Hartley. Linear pushbroom cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):963–975, 1997.

[6] T. He, L. Hong, A. Kaufman, and H. Pfister. Generation of transfer functions with stochastic search techniques. In *Proc. of Visualization*, pages 227–ff., 1996.

[7] S. B. Kang. A survey of image-based rendering techniques. In *Proc. of Videometrics, SPIE*, pages 2–16, 1999.

[8] E. LaMar and V. Pascucci. A multi-layered image cache for scientific visualization. In *Proc. of IEEE Symposium on Parallel and Large-Data Visualization and Graphics*, pages 61–68, 2003.

[9] M. Levoy and P. Hanrahan. Light field rendering. In *Proc. of SIGGRAPH*, pages 31–42, 1996.

[10] T. Luft, C. Colditz, and O. Deussen. Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics*, 25(3):1206–1213, 2006.

[11] E. J. Luke and C. D. Hansen. Semotus Visum: a flexible remote visualization framework. In *Proc. of Visualization*, pages 61–68, 2002.

[12] K.-L. Ma, M. F. Cohen, and J. S. Painter. Volume seeds: A volume exploration technique. *Computer Graphics and Visualization*, 2:135–140, 1991.

[13] M. M. Malik, T. Möller, and M. E. Gröller. Feature peeling. In *Proc. of Graphics Interface*, pages 273–280, 2007.

[14] W. R. Mark, L. McMillan, and G. Bishop. Post-rendering 3d warping. In *In Proc. of Interactive 3D Graphics and Games Symposium*, pages 7–ff., 1997.

[15] J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. Design galleries: a general approach to setting parameters for computer graphics and animation. In *Proc. of SIGGRAPH*, pages 389–400, 1997.

[16] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[17] L. McMillan and G. Bishop. Plenoptic modeling: an image-based rendering system. In *Proc. of SIGGRAPH*, pages 39–46, 1995.

[18] C. Mei, V. Popescu, and E. Sacks. The occlusion camera. 24(3):335–342, 2005.

[19] T. Pajdla. Geometry of two-slit camera, 2002.

[20] 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 Computer Graphics and Applications*, 21(3):16–22, 2001.

[21] V. Popescu and D. G. Aliaga. The depth discontinuity occlusion camera. In *Proc. of Interactive 3D Graphics and Games Symposium*, pages 139–143, 2006.

[22] V. Popescu, P. Rosen, L. L. Arns, X. Tricoche, C. Wyman, and C. Hoffmann. The general pinhole camera: effective and efficient non-uniform sampling for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 99(1), 2010.

[23] P. Rademacher and G. Bishop. Multiple-center-of-projection images. In *Proc. of SIGGRAPH*, pages 199–206, 1998.

[24] P. Rautek, S. Bruckner, and M. E. Groller. Semantic layers for illustrative volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1336–1343, 2007.

[25] C. Rezk-Salama and A. Kolb. Opacity peeling for direct volume rendering. *Computer Graphics Forum*, 25(3):597–606, 2006.

[26] T. Ropinski, J.-S. Prassni, F. Steinicke, and K. H. Hinrichs. Stroke-based transfer function design. In *Proc. of IEEE/EG International Symposium on Volume and Point-Based Graphics*, pages 41–48, 2008.

[27] J. Shade, S. Gortler, L. wei He, and R. Szeliski. Layered depth images. In *Proc. of SIGGRAPH*, volume 1, pages 231–242, 1998.

[28] N. Shareef, T.-Y. Lee, H.-W. Shen, and K. Mueller. An image-based modeling approach to gpu-based unstructured grid volume rendering. In *Proc. of Volume Graphics*, pages 31–38, 2006.

[29] V. Srivastava, U. Chebrolu, and K. Mueller. Interactive transfer function modification for volume rendering using compressed sample runs. In *Proc. of Computer Graphics International Conference*, pages 8–13, 2003.

[30] A. Tikhonova, C. D. Correa, and K.-L. Ma. Explorable images for visualizing volume data. In *Proc. of IEEE Pacific Visualization Symposium*, pages 177–184, 2010.

[31] A. Tikhonova, C. D. Correa, and K.-L. Ma. An exploratory technique for coherent visualization of time-varying volume data. *Computer Graphics Forum*, 29(3):783–792, 2010.

[32] D. N. Wood, A. Finkelstein, J. F. Hughes, C. E. Thayer, and D. H. Salesin. Multiperspective panoramas for cel animation. In *Proc. of SIGGRAPH*, pages 243–250, 1997.

[33] Y. Wu and H. Qu. Interactive transfer function design based on editing direct volume rendered images. *IEEE Transactions on Visualization and Computer Graphics*, 13(5):1027–1040, 2007.

[34] J. Yu and L. McMillan. A framework for multiperspective rendering. In *Proc. of Eurographics Symposium on Rendering*, pages 61–68, 2004.