SCI INSTITUTE
TECHNICAL REPORT

A Directional Occlusion Shading Model for Interactive Direct Volume Rendering

Mathias Schott[*], *Vincent Pegoraro*[*], *Charles Hansen*[*], *Kévin Boulanger*[†], *Josh Stratton*[*], *and Kadi Bouatouch*[†]

[*]Scientific Computing and Imaging Institute, School of Computing, University of Utah, USA

[†]IRISA, Université de Rennes I, France

UUSCI-2008-009

Scientific Computing and Imaging Institute
University of Utah
Salt Lake City, UT 84112 USA

December 11, 2008

Abstract:

Volumetric rendering is widely used to examine 3D scalar fields from scanners and direct numerical simulation datasets. One key aspect of volumetric rendering is the ability to provide shading cues to aid in understanding structure contained in the datasets. While shading models that reproduce natural lighting conditions have been shown to better convey depth information and spatial relationships, they traditionally require considerable (pre-)computation. In this paper, we propose a novel shading model for interactive direct volume rendering that provides perceptual cues similar to that of ambient occlusion, for both solid and transparent surface-like features. An image space occlusion factor is derived from the radiative transport equation based on a specialized phase function. Our method does not rely on any precomputation and thus allows for interactive explorations of volumetric data sets via on-the-fly editing of the shading model parameters or (multi-dimensional) transfer functions. Unlike ambient occlusion methods, modifications to the volume, such as clipping planes or changes to the transfer function, are incorporated into the resulting occlusion-based shading.

THE UNIVERSITY OF UTAH

A Directional Occlusion Shading Model for Interactive Direct Volume Rendering

Mathias Schott[1], Vincent Pegoraro[1], Charles Hansen[1], Kévin Boulanger[2], Josh Stratton[1], Kadi Bouatouch[2]

[1]Scientific Computing and Imaging Institute, School of Computing, University of Utah, USA
[2]IRISA, Université de Rennes I, France

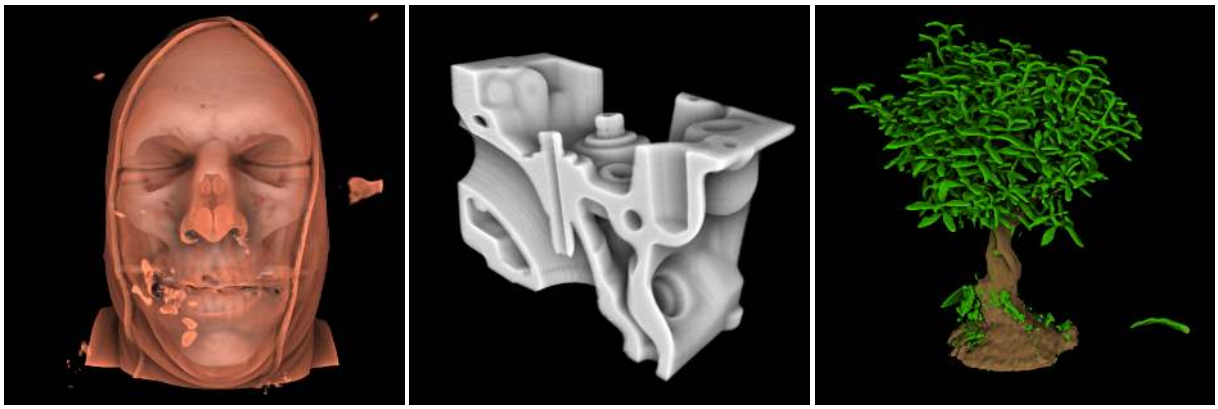


Figure 1: *From left to right: a) Visible male data set with occlusion of solid and transparent materials (3.4 FPS, 996 slices) b) CT scan of an engine block where a clipping plane was used to show the exhaust port (13.3 FPS, 679 slices) c) Bonsai data set of which complex features are exposed by our ambient occlusion approximation (4.1 FPS, 1492 slices)*

Abstract

Volumetric rendering is widely used to examine 3D scalar fields from scanners and direct numerical simulation datasets. One key aspect of volumetric rendering is the ability to provide shading cues to aid in understanding structure contained in the datasets. While shading models that reproduce natural lighting conditions have been shown to better convey depth information and spatial relationships, they traditionally require considerable (pre-)computation. In this paper, we propose a novel shading model for interactive direct volume rendering that provides perceptual cues similar to that of ambient occlusion, for both solid and transparent surface-like features. An image space occlusion factor is derived from the radiative transport equation based on a specialized phase function. Our method does not rely on any precomputation and thus allows for interactive explorations of volumetric data sets via on-the-fly editing of the shading model parameters or (multi-dimensional) transfer functions. Unlike ambient occlusion methods, modifications to the volume, such as clipping planes or changes to the transfer function, are incorporated into the resulting occlusion-based shading.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism— Subjects: Color, shading, shadowing, and texture

1. Introduction

Volumetric rendering is widely used to examine 3D scalar fields from scanners and direct numerical simulation datasets. One key aspect of volume rendering is the ability to provide shading cues to aid in understanding structures contained in the datasets. Various shading models, as discussed in Section 2, have been proposed to improve the comprehensibility of complex features in volumes. Especially methods which take the relative spatial arrangement of features into

account have been successfully applied to enhance the quality of visualizations of volumetric data sets [LB00].

An example of such techniques is ambient occlusion, which is the amount of light reaching a point from a spherical environment light source with uniform intensity enclosing the whole scene. The surrounding area light source causes occluding structures to cast soft shadows onto features of the scene, giving critical cues to a human observer about their relative spatial arrangement.

Computing ambient occlusion is very challenging, since the required global information about the volume makes the task computationally intensive and commonly involves precomputation as an approach to make the technique feasible. Expensive computation however conflicts with the desire for an interactive volume rendering technique with user flexibility. Allowing the change of transfer functions or clipping planes precludes expensive pre-computations and makes it necessary to recompute shading parameters for every frame.

This paper presents a novel occlusion-based shading method yielding depth cues similar to those of ambient occlusion. The proposed method renders occlusion effects interactively and thus allows user driven manipulation of multi-dimensional transfer functions, which have been shown to improve the classification of materials in data sets of complex composition [KKH02]. User-placed clipping planes are incorporated into the computation of the occlusion effects. Transparent and solid surfaces are shaded to give additional insight when it is desirable to show multiple occluding surfaces at the same time, and volumetric data sets with high feature complexity benefit from the presented method.

This paper is structured the following way: the next section contains a discussion of related work with focus on volumetric illumination and shading models. Section 3 derives the directional occlusion shading method from the physics of radiative transport and outlines an integration into a slice-based 3D texture volume renderer. Results are presented and discussed in Section 4, followed by conclusions and future work in Section 5.

2. Related Work

The shading model used during volume rendering plays a critical role in helping the user gaining insight during the exploration of a volumetric data set. Simple approximations to light transport, such as emission-absorption models [Max95] or local illumination shading models such as the Blinn illuminations model [Bli77] build the foundation of many volume rendering systems, thanks to their low computational cost and their simplicity of implementation.

However, light contributions from surrounding features give important perceptual cues supporting the comprehension and understanding of complex features in volumetric data sets. Shadows have been incorporated into volume rendering systems, as basic means of gaining additional lighting and occlusion information, e.g. as precomputed shadow volumes [BR98], variations of image space shadow maps [KKH02, DEP05, HKSB06] or via ray casting [RKH08].

Ambient occlusion methods, especially for polygonal geometry, have recently received considerable attention since they provide, with an expensive pre-processing step, a method to realistically shade scenes under uniform diffuse illumination, at a cheaper cost than full global illumination. Here, we will only discuss ambient occlusion techniques for volume rendering and refer the reader to a recent survey [AMFS08] on ambient occlusion techniques in general.

The Vicinity Shading [Ste03] method determines as a preprocessing step, a vicinity value for each voxel by taking into consideration the densities of surrounding voxels. Voxels with densities higher than the current voxel's density are considered to be occluding. Other voxels on paths between the current voxel and the occluding voxels are not considered during computation of vicinity values. The volume of vicinity values is then used during rendering as part of a shading equation. Obscurance Shading [RBV*08] generalizes the Vicinity Shading technique by taking the distances between a voxel and its occluding voxels into account during the computation of vicinity/obscurance values. It also allows for a simple and low cost implementation of color bleeding. However both methods require a recomputation of the vicinity/obscurance values when volume classification parameters such as the iso-value or the transfer function change. Vicinity Occlusion Maps [DYV08] approximate Vicinity Shading in image space by computing halos via comparison of depth values with the surrounding average depth values as a measure for the arrangement of occluding voxels in the vicinity.

Hernell *et al.* [HLY07] compute a local ambient occlusion approximation by considering voxels in a neighborhood around each voxel under illumination of a spherical light source. They cast rays for some number of directions against those spheres and integrate optical depths along them to attenuate light intensities and add emissive contributions which they store as volume textures. They exploit multiresolution block-based data structures and spatial under sampling to accelerate the computation of the illumination volumes. Changing the transfer function causes an interactive incremental refinement of the ambient and emissive illumination volumes by progressively adding more ray directions to the solution as well as a recomputation of the block-based data structures due to their dependency on the voxel opacities specified by the transfer function. A subsequent publication [HLY08] extends this local ambient occlusion shading with global shadows and first order scattering effects. They reuse the computational results of the local ambient occlusion to compute global shadowing effects via piecewise in-

tegration. Global shadows and first order scattering effects are stored in coarser resolution block-based data structures.

Ritschel [Rit07] considers the illumination of a volume data set under distant spherical illumination represented by an environment map. A visibility function is computed by integrating the transmittance along hundreds of rays emanating from each voxel. Spatial undersampling in conjunction with an adaptive traversal through a hierarchical data structure, called *expected opacity map*, is used to accelerate the computation of the visibility function. A spherical harmonics decomposition of both the visibility function and the environment maps allows efficient storage and evaluation of low frequency indirect illumination values during direct volume rendering. Changing the transfer function requires a few seconds for recomputation of the visibility function.

Desgrange *et al.* [DE07] use a linear combination of opacity volumes blurred with different filter sizes to approximate the occlusion by the average opacity of surrounding voxels. They use summed area volumes to accelerate the computation of the average opacity volumes, which need to be recomputed after the transfer function changes.

Filterable occlusion maps [PM08] treat the number of occluding voxels in the neighborhood as a measure of the occlusion. An approach similar to variance shadow maps [DL06] is used to represent statistical information about neighboring voxels and use them to compute ambient occlusion and soft shadow effects during the rendering of isosurfaces. The computation of the filterable occlusion maps, a hierarchical representation of bimodal distributions, runs interactively on recent GPUs. However, this statistical approximation is only valid if the data set exhibits two clearly separable clusters of density values, as often found in CT scans, but less often in MRI scans. Arbitrary iso-values can be chosen, but the representation gets more inaccurate the further the iso-value approaches the mean of the assumed probability distribution. They show that it is possible to combine filterable occlusion maps with direct volume rendering, but partially transparent materials don't influence the occlusion computation.

Ropinski *et al.* [RMSD*08] compute and quantize the configuration of neighboring voxels independent of a specific transfer function as a lengthy precomputation step. Combining those precomputed values with a transfer function allows them to interactively render ambient occlusion and color bleeding effects. In contrast to their method, we support multi-dimensional transfer functions and post-classification.

3. Directional Occlusion Shading

The proposed directional occlusion shading model captures the important perceptional cues caused by soft shadow effects of ambient occlusion techniques without having to evaluate a prohibitively expensive global ambient occlusion solution. Unlike previous approaches which only consider local occlusion in a spherical neighborhood, we take into account all features between a point and the environment light. However, instead of computing occlusion on the whole sphere, we make use of a specialized phase function, namely a backward-peaked cone phase function of user specified aperture angle, as illustrated in Figure 2. This is in spirit similar to the *light transport angle* used to render scattering effects interactively [KPH*03] or to the *cone angle* used by artists to control ambient occlusion effects in offline image generation [Chr03]. A Monte-Carlo raytracer was used

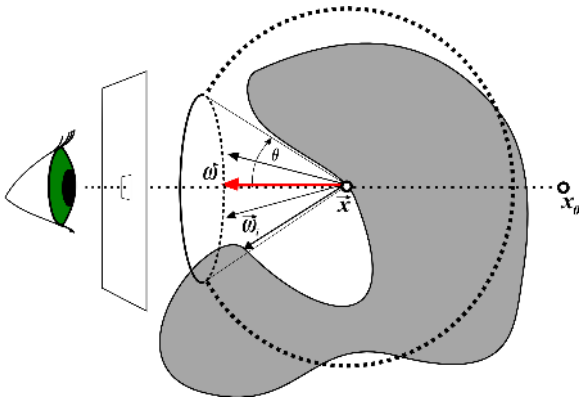


Figure 2: *The geometric setup of the proposed direct occlusion shading model showing the conical subset of the sphere used to compute the occlusion.*

to empirically compare directional occlusion (Figure 4(b)) and volumetric ambient occlusion with an isotropic phase function (Figure 4(a)). While the results differ, our specialized phase function is able to highlight features of interest for visualization purposes in a manner similar to full ambient occlusion.

By changing the aperture angle of the cone, as discussed in Section 4, it is possible to vary the influence of features at different distances, thus allowing the user to choose a balance between visualizing local and global structures. A complete evaluation of the occlusion, albeit being closer to the physics of light transport, would restrict this flexibility.

In this section, we outline the radiative transport equation and the derivation of the directional occlusion shading model. Given this theoretical framework, we then show how a slice-based 3D texture volume renderer can be extended to support directional occlusion shading.

3.1. The Radiative Transport Equation

The radiance integrated along a ray with origin $\vec{x}_0$ and direction $\vec{\omega}$, passing through a participating medium is described by the radiative transport equation (RTE) [CPCP*05], which

reads

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{x_0})L_b(\vec{x_0},\vec{\omega}) + L_m(\vec{x},\vec{\omega}) \quad (1)$$

$$L_m(\vec{x},\vec{\omega}) = \int_{\vec{x}}^{\vec{x_0}} T(\vec{x},\vec{x'})\sigma_a(\vec{x'})L_e(\vec{x'},\vec{\omega})d\vec{x'} + \quad (2)$$

$$\int_{\vec{x}}^{\vec{x_0}} T(\vec{x},\vec{x'})\sigma_s(\vec{x'})L_i(\vec{x'},\vec{\omega})d\vec{x'} \quad (3)$$

This equation expresses the radiance at a given point as the sum of the attenuated background radiance $L_b(\vec{x_0},\vec{\omega})$ and the medium radiance $L_m(\vec{x},\vec{\omega})$. The latter integrates the contributions of the emitted radiance $L_e(\vec{x},\vec{\omega})$, weighted by the wavelength dependent absorption coefficient σ_a, and the in-scattered radiance $L_i(\vec{x},\vec{\omega})$, also weighted by the wavelength dependent scattering coefficient σ_s. The extinction coefficient $\sigma_t = \sigma_a + \sigma_s$ describes the amount of radiance lost through absorption and out-scattering via the transmittance $T(\vec{x_a},\vec{x_b})$ and the optical thickness $\tau(\vec{x_a},\vec{x_b})$.

$$T(\vec{x_a},\vec{x_b}) = e^{-\tau(\vec{x_a},\vec{x_b})} \quad (4)$$

$$\tau(\vec{x_a},\vec{x_b}) = \int_{\vec{x_a}}^{\vec{x_b}} \sigma_t(\vec{x})d\vec{x} \quad (5)$$

The emitted radiance only depends on local properties of the medium, but the in-scattered radiance $L_i(\vec{x'},\vec{\omega})$ depends on global properties of the medium, since it is defined as the integral of the radiance incoming at the point $\vec{x'}$ over all directions $\vec{\omega_i}$, weighted by the material specific phase function $\Phi(\vec{\omega},\vec{\omega_i})$.

$$L_i(\vec{x'},\vec{\omega}) = \int_{4\pi} L(\vec{x'},\vec{\omega_i})\Phi(\vec{\omega},\vec{\omega_i})d\vec{\omega_i}. \quad (6)$$

The recursive evaluation of $L(\vec{x'},\vec{\omega_i})$ in Equation 6 makes the RTE in its generality very challenging to evaluate. However, simplified models suffice for visualization purposes while allowing for a significant reduction in computational cost.

3.2. The Directional Occlusion Shading Model

We model the directionality of our occlusion evaluation with a cone-shaped phase function $\Phi_\theta(\vec{\omega},\vec{\omega_i})$, as given in Equation 7. The aperture angle of the cone is given as $\theta \in [0, \frac{\pi}{2})$.

$$\Phi_\theta(\vec{\omega},\vec{\omega_i}) = \begin{cases} 0 & \text{if} \langle\vec{\omega},\vec{\omega_i}\rangle < \cos(\theta) \\ \frac{1}{2\pi(1-\cos\theta)} & \text{otherwise} \end{cases} \quad (7)$$

The radius $r = \frac{1}{2\pi(1-\cos\theta)}$ is such that the phase function is normalized such that $\int_{4\pi}\Phi(\vec{\omega},\vec{\omega_i})d\vec{\omega_i} = 1$. In our model, $\vec{\omega}$ corresponds to the viewing direction and the $\vec{\omega_i}$ are chosen to be within the cone. We then simplify the radiance transport equation by assuming that the medium does not emit light ($L_e(\vec{x'},\vec{\omega}) = 0$) and scatters light only at directions within a cone, described by a cone phase function $\Phi_\theta(\vec{\omega},\vec{\omega_i})$. For the in-scattered radiance we only consider first order scattering events. For a certain direction $\vec{\omega}$, $L(\vec{x'},\vec{\omega_i})$ equals to the constant background radiance $L_b(\vec{x'_0},\vec{\omega_i})$ attenuated by the transmittance along $\vec{\omega_i}$ and scaled by $\Phi_\theta(\vec{\omega},\vec{\omega_i})$, dropping all other terms of the recursive evaluation of $L(\vec{x},\vec{\omega})$ in Equation 6. Equations 8 to 11 describe our model for directional occlusion shading:

$$L(\vec{x},\vec{\omega}) = T(\vec{x},\vec{x_0})L_b(\vec{x_0},\vec{\omega}) + L_m(\vec{x},\vec{\omega}) \quad (8)$$

$$L_m(\vec{x},\vec{\omega}) = \int_{\vec{x}}^{\vec{x_0}} T(\vec{x},\vec{x'})\sigma_s(\vec{x'})L_i(\vec{x'},\vec{\omega})d\vec{x'} \quad (9)$$

$$L_i(\vec{x'},\vec{\omega}) = L_b(\vec{x'_0},\vec{\omega})V(\vec{x'},\vec{\omega}) \quad (10)$$

$$V(\vec{x'},\vec{\omega}) = \int_{4\pi} T(\vec{x'},\vec{x'_0})\Phi_\theta(\vec{\omega},\vec{\omega_i})d\vec{\omega_i} \quad (11)$$

The in-scattered radiance $L_i(\vec{x'},\vec{\omega})$ in Equation 10 can be further decomposed into the product of a fractional visibility term $V(\vec{x'},\vec{\omega})$, as given in Equation 11, and the background radiance $L_b(\vec{x'_0},\vec{\omega})$. The fractional visibility is conceptually equivalent to the *occlusion factor* of surface ambient occlusion methods, since it describes how much of the environment's light is reaching the point $\vec{x'}$. Section 3.4 explains how to use additional buffers to interactively calculate occlusion factors during volume rendering using a slice-based volume renderer.

It is difficult to measure σ_s and σ_a for real world data sets, since typical data acquisition devices such as CT scanners or MRI scanners measure densities or orientations of magnetic fields. Therefore, transfer functions are used to map attributes specified at positions in the volume to optical properties σ_s and σ_a or information about the phase function Φ. For defining optical properties using 2D transfer functions, we follow the accepted practice of specifying *colors* and *opacities* for certain data values in the data set. Color maps to the chromatic scattering coefficients σ_s and opacity to the achromatic extinction coefficient σ_t. The absorption coefficient σ_a is irrelevant, since we do not consider emissive radiance in our model.

3.3. High Level Rendering Algorithm

We integrated our algorithm, as outlined in appendix A, into a slice-based 3D texture volume renderer [WVW94]. A slice-based volume renderer creates proxy geometry, usually by intersecting view aligned planes with the bounding box of the volume. Then, graphics hardware is used to render those slices during which a fragment program computes color and opacity values which finally get composited with the image of previous slices already stored in the frame buffer.

Similar to the half-angle based illumination and shading methods [KKH02,KPH*03], we employ an additional buffer to store and compute the occlusion factor for each slice. As we traverse the volume slice by slice, we alternately update the eye buffer and the occlusion buffer. During the additional render pass to update the occlusion buffer, we approximate Equation 11 by averaging samples taken from the previous occlusion buffer.

After computation of the proxy geometry on the CPU, we

render each slice at first from the camera's point of view, projecting it into the eye buffer. During this pass, we determine the occlusion factor by projecting the fragment position into the occlusion buffer and mapping its position in normalized device coordinates to the typical $[0,1]^2$ domain of the texture coordinates. We then multiply the color of the sample by the occlusion factor and the background radiance and blend it into the eye buffer. We use front-to-back compositing to update the occlusion buffer in locked-step with the eye buffer.

Next, we update the occlusion information by rendering the slice again, but now into the next occlusion buffer. During this pass, we integrate the occlusion factors of the previous occlusion buffer by reading and averaging multiple samples, as described in greater detail in Section 3.4. Finally, we swap the current and next occlusion buffer and proceed to process the next slice.

When the final eye buffer has been computed, we copy it into the frame buffer of the visible window, converting the final values from floating point format to the fixed point format usually used for displaying images on the screen. If required, tone-mapping could be applied at this stage of the algorithm.

3.4. Interactive Computation of the Occlusion Factor

It is possible to approximate the integration of the occlusion factors described in Equation 11 by averaging samples from the previous occlusion buffer in a neighborhood around the current fragment for propagation of the occlusion to the current occlusion buffer. Figure 3 shows the geometric setup for the computation of the occlusion factors.

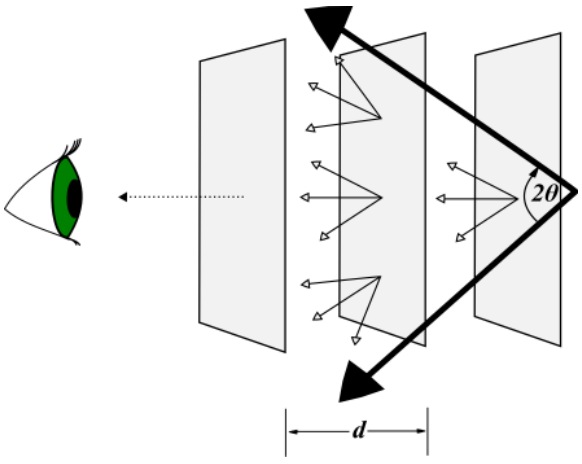


Figure 3: *The geometric setup used for the interactive computation of the occlusion factors.*

Given the aperture angle of the cone θ and the inter-slice distance d, we define the *occlusion_extent* as the radius of the cone's base enclosing all samples, and compute it as depicted it Equation 12.

$$occlusion\_extent = d\tan(\theta) \tag{12}$$

Sample positions p are generated on the disk with radius *occlusion_extent* and transformed according to Equations 14 to 17. Perspective projection of the cone's base causes the *occlusion_extent* to vary as a function of the z coordinate of the slice. A correction factor c, as described in Equation 14, is computed in the vertex shader using the world-view-projection matrix WVP and the z-component of the vertices of the slices, which is constant across a given slice.

$$\begin{pmatrix} x_{scale} \\ y_{scale} \\ z_{scale} \\ w_{scale} \end{pmatrix} = WVP \begin{pmatrix} 1 \\ 1 \\ z \\ 1 \end{pmatrix} \tag{13}$$

$$c = \begin{pmatrix} \frac{x_{scale}}{w_{scale}} \\ \frac{y_{scale}}{w_{scale}} \end{pmatrix} \tag{14}$$

The correction factor c is then passed to the fragment shader where it is used to compute texture coordinates for sampling the previous occlusion buffer, given in Equation 17. A perspective division projects the interpolated clip space position xy_c of a fragment into normalized device coordinates xy_d. Then, a sample position p on a disk with radius *occlusion_extent* is scaled by the correction factor c and added to xy_d. This yields a sample position in normalized device coordinates p_d which is scaled by 0.5 and biased by 0.5 to determine the texture coordinate p_t to be used to sample the previous occlusion buffer.

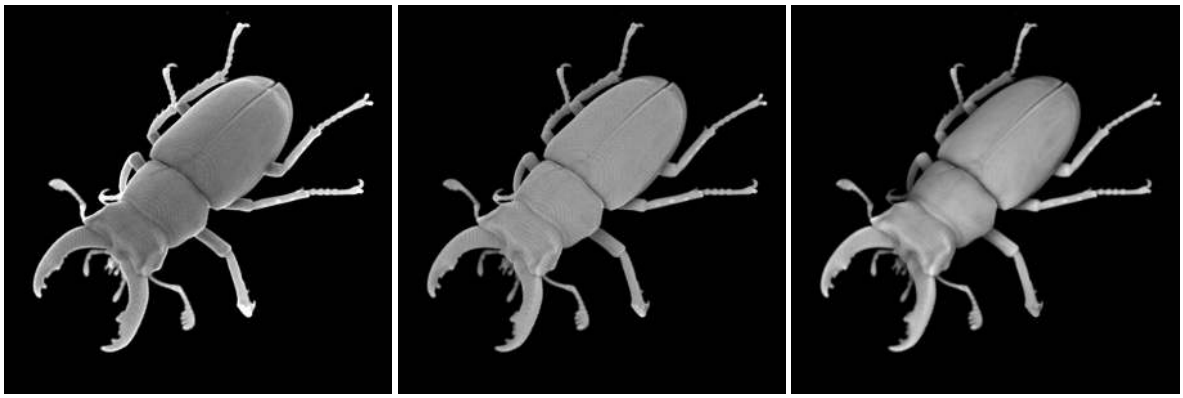
$$xy_d = \frac{xy_c}{w_c} \tag{15}$$

$$p_d = xy_d + c * p \tag{16}$$

$$p_t = 0.5 * p_d + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} \tag{17}$$

Computing the integral approximation is done by averaging the texture representing the previous occlusion buffer at all sample positions p_t.

The position p and number of samples have a major impact on the performance and quality of the approximated integration. Stochastic sample distributions have been shown to reduce aliasing considerably [Coo86], though they require high numbers of samples in order to reduce the introduced noise. However, to maintain interactivity, it is desirable to take a low amount of samples. Thus, a uniform grid of user specifiable resolution has been chosen as a compromise between performance and image quality. Its symmetry and orientation of samples along the axes of the occlusion buffer reduce inter-frame continuity artifacts when the camera position changes. Most of the presented images have been rendered using grid resolutions between 2×2 and 4×4.



(a) Monte-Carlo, isotropic phase function, off-line (b) Monte-Carlo, cone phase function, off-line (c) Interactive directional occlusion shading, 7.3 FPS

Figure 4: *The stag beetle data set rendered using a) Monte-Carlo integration of Equation 8, with an isotropic phase function b) Monte-Carlo integration of Equation 8, with a cone phase function of aperture angle* $\theta = 80^\circ$ *and c) interactive directional occlusion shading with 1046 slices and aperture of* 80°.

4. Results and Discussion

Figure 4 compares the results obtained with the GPU implementation of the directional occlusion shading model against the results generated by an offline Monte-Carlo raytracer. Although our approximation presents slight differences with the reference images, it allows most of the complex shading effects to be rendered at interactive frame rates without any precomputation.

The presented images in this paper were rendered on an NVIDIA GeForce GTX 280 GPU with 1 GB of video memory using an unoptimized implementation based on OpenGL and CG. An inter-slice distance $d = 0.001$ was used to capture details introduced by multi-dimensional transfer functions. Floating point buffers with 16-bit precision were used to render images with 512×512 pixel resolutions.

Figure 5 shows the directional occlusion shading model and the Lambertian diffuse shading model applied to different data sets. Even though the diffuse shading model provides basic hints about features, the shading differences introduced by the directional occlusion shading model accentuates structures such as concavities and depth discontinuities. The creases of the brain in Figure 5(a) and the eye sockets in Figure 5(b) are examples of the former; the soft shadows cast by the zygomatic bone onto the sphenoid bone in Figure 5(b) and the emphasized boundaries of the fish bones with respect to the background surface in Figure 5(c) for the latter.

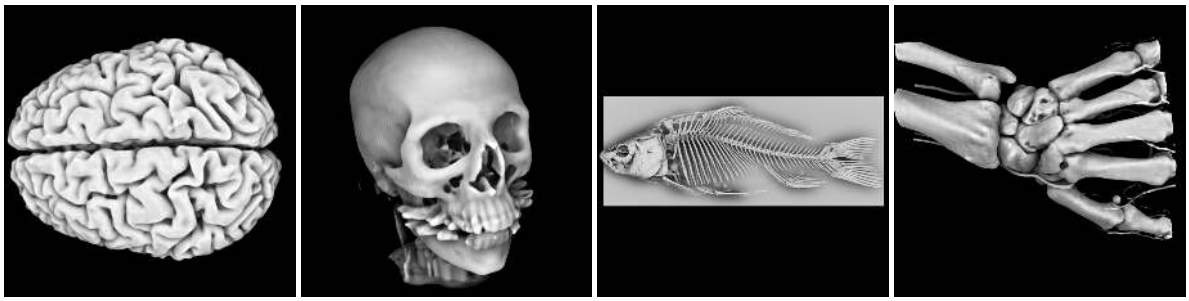
Figure 6 shows the effect of varying the aperture angle of the cone phase function. Higher values of θ cause more distant features to increase the shading variation of surfaces, as demonstrated in Figures 6(c) and 6(a) which contrast a value of $\theta = 50^\circ$ with a value of $\theta = 80^\circ$. Ambient occlusion and obscurance-based techniques expose similar levels of control by letting the length of the occlusion rays determine the influence of more distant features in the volume.

Increasing the resolution of the sampling grid reduces the aliasing especially for high values of θ where the base of the cone covers a relatively high numbers of texels. For example in Figures 6(c) and 6(d), the cone aperture angle is $\theta = 80^\circ$, thus the occlusion extent varies between 8.5 and 12.7 units in texel space, depending on the z value of the slice. For low values of θ, such as in Figures 6(a) and 6(b), the occlusion extent covers only a relatively small region of the occlusion buffer, therefore there are only small differences between the different grid resolutions. However, the performance is strongly dependent on the grid resolution, as one can see in the reported frame rates.

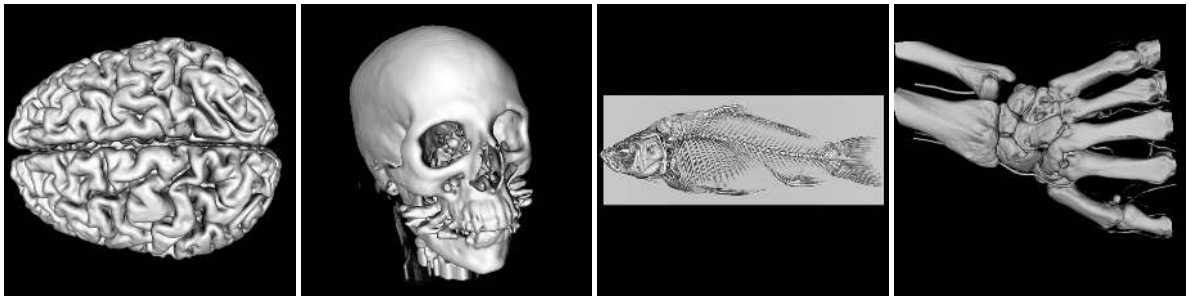
5. Conclusion and Future Work

In this paper, we have presented a directional occlusion shading model which adds occlusion effects to direct volume rendering. Restricting the occlusion computation to a cone oriented toward the viewer enables interactive rendering of plausible occlusion effects, qualitatively similar to those of full ambient occlusion techniques. The lack of precomputation allows interactive manipulation of camera position, transfer function and clipping planes, which are all taken into account during the occlusion computation.

In the future, we would like to extend our shading model to render volumetric soft shadows from arbitrary area light sources by adapting half-angle based volume shading techniques. Integration of directional occlusion shading into volume rendering techniques based on ray-casting would be also desirable, as well as the application to rendering of iso-

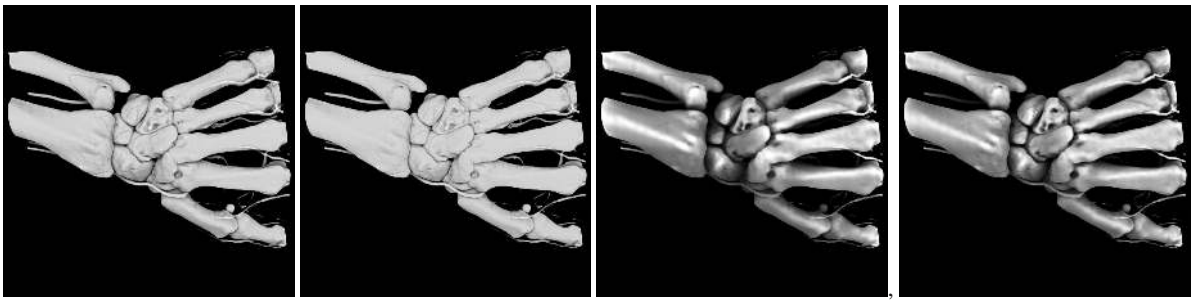


(a) 2.5 FPS (b) 1.0 FPS (c) 34.0 FPS (d) 6.0 FPS



(e) 13.3 FPS (f) 16.0 FPS (g) 59.0 FPS (h) 29.1 FPS

Figure 5: *Various data sets showing the difference between directional occlusion shading model (top row) and the diffuse Lambertian shading (bottom row) for different data sets. Two-dimensional transfer functions have been used to highlight different features of an MRI scan of a brain with a resolution of* $256 \times 256 \times 160$ *voxels, a CT scan of a head with a resolution of* $128 \times 256 \times 256$ *voxels, a CT scan of a carp with a resolution of* $256 \times 288 \times 512$ *voxels and a CT scan of a hand with a resolution of* $244 \times 124 \times 257$ *voxels. The number of slices used to render each image are 1000, 1458, 220 and 619 respectively.*



(a) $\theta = 50°$, 2×2 grid, 11.4 FPS (b) $\theta = 50°$, 13×13 grid, 0.9 FPS (c) $\theta = 85°$, 2×2 grid, 11.1 FPS (d) $\theta = 85°$, 13×13 grid, 0.9 FPS

Figure 6: *The value of* θ *and the resolution of the sampling grid determine the effect of surrounding features. In the two images to the left, the base of the cone covers between 0.8 and 1.3 texels for a* 512×512 *view port. In the two images to the right, the base of the cone covers between 8.5 and 12.7 texels, for the same view port resolution. The images were rendered with 619 slices.*

surfaces. Another direction of research would be to investigate the application of directional occlusion shading to rendering of polygonal meshes, e.g. as a result from iso-surface extraction.

References

[AMFS08] ÀLEX MÉNDEZ-FELIU, SBERT M.: From obscurances to ambient occlusion: A survey. *The Visual Computer* (2008).

[Bli77] BLINN J. F.: Models of light reflection for computer synthesized pictures. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 11*, 2 (1977), 192–198.

[BR98] BEHRENS U., RATERING R.: Adding shadows to a texture-based volume renderer. In *Proceedings of the IEEE Symposium on Volume Visualization* (1998), pp. 39–46.

[Chr03] CHRISTENSEN P. H.: Global illumination and all that. *SIGGRAPH course notes 9 (RenderMan: Theory and Practice)* (2003).

[Coo86] COOK R. L.: Stochastic sampling in computer graphics. *ACM Transactions on Graphics 5*, 1 (1986), 51–72.

[CPCP*05] CEREZO E., PEREZ-CAZORLA F., PUEYO X., SERON F., SILLION F.: A survey on participating media rendering techniques. *The Visual Computer 21*, 5 (2005), 303–328.

[DE07] DESGRANGES P., ENGEL K.: Fast ambient occlusion for direct volume rendering. *United States Patent Application*, #20070013696 (2007).

[DEP05] DESGRANGES P., ENGEL K., PALADINI G.: Gradient-free shading: A new method for realistic interactive volume rendering. *Proceedings of Vision, Modeling, and Visualization* (2005), 209–216.

[DL06] DONNELLY W., LAURITZEN A.: Variance shadow maps. In *Proceedings of the Symposium on Interactive 3D Graphics and Games* (2006), pp. 161–165.

[DYV08] DÍAZ J., YELA H., VÁZQUEZ P.: Vicinity occlusion maps: Enhanced depth perception of volumetric models. In *Computer Graphics International* (2008).

[HKSB06] HADWIGER M., KRATZ A., SIGG C., BÜHLER K.: GPU-accelerated deep shadow maps for direct volume rendering. In *Proceedings of the 21st ACM SIGGRAPH/EUROGRAPHICS Symposium on Graphics Hardware* (2006), pp. 49–52.

[HLY07] HERNELL F., LJUNG P., YNNERMAN A.: Efficient ambient and emissive tissue illumination using local occlusion in multiresolution volume rendering. In *IEEE/ EG Symposium on Volume Graphics* (2007), pp. 1–8.

[HLY08] HERNELL F., LJUNG P., YNNERMAN A.: Interactive global light propagation in direct volume rendering using local piecewise integration. In *IEEE/ EG Symposium on Volume and Point-Based Graphics* (2008), pp. 105–112.

[KKH02] KNISS J., KINDLMANN G., HANSEN C.: Multi-dimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (2002), 270–285.

[KPH*03] KNISS J., PREMOZE S., HANSEN C., SHIRLEY P., MCPHERSON A.: A model for volume lighting and modeling. *IEEE Transactions on Visualization and Computer Graphics 9*, 2 (2003), 150–162.

[LB00] LANGER M. S., BÜLTHOFF H. H.: Depth discrimination from shading under diffuse lighting. *Perception 29*, 6 (2000), 649–660.

[Max95] MAX N.: Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 1*, 2 (1995), 99–108.

[PM08] PENNER E., MITCHELL R.: Isosurface ambient occlusion and soft shadows with filterable occlusion maps. In *IEEE/ EG Symposium on Volume and Point-Based Graphics* (2008), pp. 57–64.

[RBV*08] RUIZ M., BOADA I., VIOLA I., BRUCKNER S., FEIXAS M., SBERT M.: Obscurance-based volume rendering framework. In *IEEE/ EG Symposium on Volume and Point-Based Graphics* (2008), pp. 113–120.

[Rit07] RITSCHEL T.: Fast GPU-based visibility computation for natural illumination of volume data sets . In *Eurographics Short Papers* (2007), pp. 57–60.

[RKH08] ROPINSKI T., KASTEN J., HINRICHS K. H.: Efficient shadows for GPU-based volume raycasting. In *Proceedings of the 16th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)* (2008), pp. 17–24.

[RMSD*08] ROPINSKI T., MEYER-SPRADOW J., DIEPENBROCK S., MENSMANN J., HINRICHS K. H.: Interactive volume rendering with dynamic ambient occlusion and color bleeding. *Computer Graphics Forum (Proceedings of Eurographics) 27*, 2 (2008), 567–576.

[Ste03] STEWART A. J.: Vicinity shading for enhanced perception of volumetric data. In *IEEE Visualization* (2003), pp. 355–362.

[WVW94] WILSON O., VANGELDER A., WILHELMS J.: *Direct volume rendering via 3D textures*. Tech. rep., University of California at Santa Cruz, Santa Cruz, CA, USA, 1994.

Appendix A: Pseudo Code for the Rendering Algorithm

Algorithm 1 The main rendering algorithm

ComputeProxyGeometry
$occlusion\_buffer_{previous} \Leftarrow occlusion\_buffer_{next} \Leftarrow 1$
$eye\_buffer \Leftarrow 0$
for all slices s of proxy geometry **do**
  {render slice s into eye_buffer}
  BindTexture($occlusion\_buffer_{previous}$)
  SetRenderTarget(eye_buffer)
  SetBlendingMode($1 - destination_{\alpha}, 1$)
  for all fragments f of s **do**
    $(x, |\nabla x|) \Leftarrow$ SampleVolume(f)
    $(color, opacity) \Leftarrow$TransferFunction($x, |\nabla x|$)
    $occlusion \Leftarrow$ SampleTexture($occlusion\_buffer_{previous}$, $f_{clip\_space\_position}$)
    $\alpha \Leftarrow 1 - e^{-opacity * slice\_distance}$
    $frame\_buffer \Leftarrow (L_b * color * occlusion * \alpha, \alpha)$
  end for

  {render slice s into $occlusion\_buffer_{next}$ }
  BindTexture($occlusion\_buffer_{previous}$)
  SetRenderTarget($occlusion\_buffer_{next}$)
  DisableBlending()
  compute c in the vertex shader as in Equation 14
  for all fragments f of s **do**
    $(x, |\nabla x|) \Leftarrow$ SampleVolume(f)
    $(opacity) =$TransferFunction($x, |\nabla x|$)
    $occlusion \Leftarrow 0$
    for all samples p on the base of the cone **do**
      compute p_t as in Equation 17
      $occlusion+ =$ SampleTexture($occlusion\_buffer_{previous}$, p_t)
    end for
    $occlusion \Leftarrow occlusion/|p|$
    $\alpha \Leftarrow 1 - e^{-opacity * slice\_distance}$
    $frame\_buffer \Leftarrow occlusion * \alpha$
  end for
  Swap($occlusion\_buffer_{previous}$,$occlusion\_buffer_{next}$)
end for
{eye_buffer contains image of shaded volume}
CompositeWithWindowFrameBuffer(eye_buffer)