PREPRINT, submitted to EUROGRAPHICS 2012 / P. Cignoni, T. Ertl (Guest Editors)

Interactive direct volume rendering with physically-based lighting

T.Kroes[1,2], F. H. Post[1] C. P. Botha[1,3]

[1]Data Visualization, Department of Mediamatics, TU Delft, The Netherlands
[2]Department of Orthopaedics, Leiden University Medical Centre, The Netherlands
[3]LKEB, Department of Radiology, Leiden University Medical Centre, The Netherlands

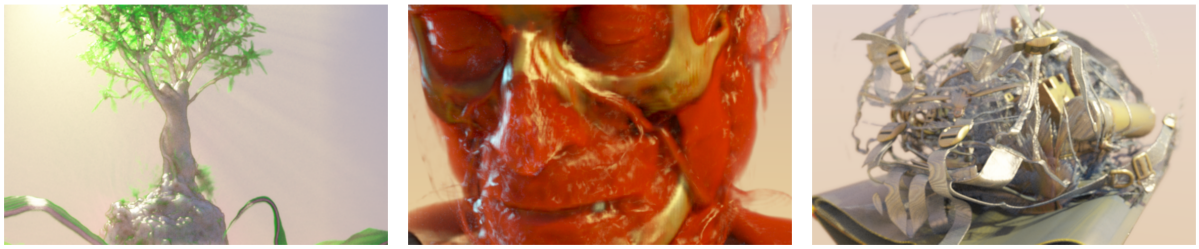

Figure 1: *From left to right: rendering of the publicly available bonsai data set, which demonstrates that our method is able to render very thin participating media, Manix and Backpack data set. The examples illustrate how volumetric and surface scattering can be effectively combined in a single volume rendering.*

Abstract

The field of volume visualization has undergone rapid development during the past years, both due to advances in suitable computing hardware and due to the increasing availability of large volume datasets. Recent work has focused on increasing the lighting realism in interactive raycasting by integrating a number of visually plausible but often effect-specific optimizations. These improvements yield impressive results, but often rely on certain assumptions, such as for example a fixed light-source, or a limited number of light-sources. On the other hand, stochastic Monte Carlo ray-tracing, coupled with physically based light transport, is able to synthesize highly realistic images. It has been successfully applied to boundary surfaces and participating media, applying either BRDF shading or phase functions. However, it has not yet been applied to interactive DVR. With this work, we present an interactive DVR approach integrating ray-traced lighting. Our method enables the configuration of any number of arbitrarily shaped and textured area lights, models a real-world camera, including its lens and aperture, and incorporates complex materials, whilst still maintaining interactive display updates. In addition, our method is able to shift dynamically between BRFDF and phase function shading to cope with heterogeneous volumetric data, and also alleviates the unwanted start-up noise usually associated with progressive rendering. Finally, we have made our complete implementation available in source and binary forms under a permissive free software license.

Keywords: Volume Rendering

1. Introduction

Realistic illumination in volume visualization plays a central role in 3D shape perception. Consequently, recent years have

seen a great deal of research towards enhancing interactive *Direct Volume Rendering* (DVR) approaches with more realistic illumination, for example ambient occlusion [ZIKS98], shadows [BR98, HKSB06] and scattering [RS07]. However, research up to now has focused on fast approximations of illumination that could be integrated with GPU-based volume renderers, both texture- and raycasting-based, as the physically-based modeling of illumination was prohibitively expensive.

Though recent literature has demonstrated that physically based rendering on GPUs is feasible [vA11], to our knowledge, physically based Monte Carlo (MC) light transport has not yet been applied to *interactive* Direct Volume Rendering (DVR). With this work, we wanted to explore 1) if current GPUs are efficient enough to support real-time simulation of MC light transport in interactive DVR and 2) what the challenges are in doing so. In order to answer these questions, we have built a CUDA-based ray-caster and enhanced it with several ray-tracing extensions that greatly increase the realism of the resultant images. We model a real-world camera, including lens and aperture, integrate physically-based light scattering and solve problems with startup noise, caused by MC integration.

The resulting interactive DVR framework is able to generate high quality images at interactive speed. All scene parameters, e.g. transfer function, camera and lighting, can be modified interactively.

Based on our experiments we conclude that stochastic MC based simulation of light transport is an attractive solution to the problem of photo realistic rendering in interactive DVR. Stochastic MC based simulation of light transport is particularly interesting because it allows to integrate various physically based effects into a unified approach without significant effort, whereas other solutions restrict the number of lights, the shape of lights, the camera model, and so forth. Furthermore, due to its sampling nature, problems with aliasing and stepping artifacts are dealt with easily. The proposed solution is able to cope with complex lighting on the fly, and the increased quality of the images help to convey shape and detail.

Our implementation does not yet include indirect lighting, such as for example inter-reflection. However, in the case of volumetric datasets, and especially medical volume datasets, this does not contribute significantly over the volumetric shadows and full lighting support that we already have. That being said, it is a question of time until physically-based indirect lighting will start to make its entrance in interactive volume rendering.

With this work, our contributions are the following:

- We present a new GPU-based interactive direct volume renderer that integrates stochastic ray-traced lighting, thus enabling physically-based volumetric shadows, any number of arbitrarily positioned, shaped and textured area lights and finally the modeling of a real-world camera, including its lens and aperture.
- We propose a practical method to mix surface and volumetric scattering in a sensible way, exploiting the best properties of both approaches.
- We show that volume data can in fact be rendered with photo-realistic ray-traced lighting at interactive rates on a desktop computer, by employing CUDA on a consumer GPU, and variance based noise reduction.
- We make available the complete open source implementation of our renderer under a permissive open source license, so that others can reproduce our results and extend our work.

The rest of the paper is structured as follows. In section 2 we survey related work on volumetric shadows, ambient occlusion and physically-based light transport. In section 3, we document our technique and in section 4 we analyze the performance and present some example renderings. Finally in section 5 we summarize our findings and point out directions for future research.

2. Related work

The work described in this paper enhances interactive direct volume rendering with ray-traced lighting and stochastic sampling, yielding a volume renderer that supports real shadows, any number of arbitrarily positioned, shaped and textured lights and high quality progressively updated high dynamic range images. In this section, we compactly survey related work, which consists primarily of volume rendering extensions to simulate shadows, of ambient occlusion in volume rendering and finally of physically-based methods.

Global illumination, and especially shadows, are compelling ways of conveying depth and shape in 3D visualization in general and in volume rendering specifically. Some of the first improvements of the more straightforward light transport approximations, such as that presented by Max [Max95], were made with the introduction of shadows in volume rendering. Behrens and Ratering introduced shadows in volume rendering by pre-computing a shadow volume, for a given relative light source position, that could then be rendered using a standard texture-based volume rendering algorithm [BR98]. Kniss et al. presented a volumetric lighting model that integrated half angle slicing, a texture-based volume rendering technique to calculate volumetric shadows, a lookup table-based phase function implementation and an approximation of multiple forward scattering based on aggregating light from previous slices [KKH02, KPH*03].

Hadwiger et al. [HKSB06] adapted deep shadow maps [LV00], a technique for computing semi-transparent volumetric shadows, to raycasting on the GPU. Ropinski et al. [RKH08] presented an alternative implementation of deep shadow maps for GPU raycasting that supported caching when the light source configuration was kept

constant, and compared it to normal shadow maps (non-transparent shadows) and shadow rays. In all cases, volume rendering realism was greatly improved with the integration of shadows. However, the mentioned examples were all limited to modeling a single point light source. Because our work also integrates stochastically sampled shadow rays, it supports the modeling of any number of arbitrarily shaped and even textured light sources.

Ambient occlusion, introduced by Zhukov et al. [ZIKS98] with the term *obscurances*, is an effective and usually inexpensive technique for approximating global illumination. Ambient occlusion computes the light intensity on a shading point by determining the hemispherical occlusion of environmental light. In their survey, Méndez-Feliu and Sbert point out the difference between obscurances and ambient occlusion: Whilst the latter represents the degree of openness of a point, the former also takes into account diffuse indirect lighting, yielding more physically correct lighting and for example color bleeding effects [MFS09]. However, the two terms are often used interchangeably. The vicinity shading method introduced by Stewart [Ste03] was the first to incorporate ambient occlusion into volume rendering, with a method called vicinity shading. Their method uses neighboring voxels and their obscurances to compute local illumination, which results in darkened crevices and depressions. Their method requires preprocessing for every scene modification and requires an additional buffer to store the results of the illumination. Penner and Mitchell [PM08] used histograms to classify the obsurance around a voxel. The method by Ropinski et al. [RMSD*08] used local histogram clustering for the precomputation of occlusion information. It is important to note that ambient occlusion and even obscurance do not take into account a specific light position, but are both based on the geometric occlusion of a sample point, and hence approximate the light that could conceivable reach that point.

Ritschel et al. [Rit07] combine a form of ambient occlusion represented as spherical harmonics, which they call the visibility function, with a DVR approach, relating the emission at each point to both its density and the interaction between the incoming light, from a single source, and the direction-dependent visibility function. Lindemann et al. [LR10] extend the work of Ritschel et al. with a spherical harmonic representation of the incident direct and indirect lighting that integrates chromatic attenuation as well as a local approximation of subsurface scattering. They claim to support area light sources, but from the paper it is not clear how these are defined. In the work of Kronander et al. [KJL*11] lighting, as well as visibility are encoded in the spherical harmonics. Their method supports directional, point and environment lights.

Schott et al. [SPH*09] introduced a direct volume rendering technique, named directional occlusion, inspired by the ambient occlusion algorithm. Their method is limited to a light source that has to coincide with the camera. This restriction was later partially removed by allowing the user to place a light within a hemisphere, oriented towards the camera, with the introduction of a multi directional occlusion model by Solteszova et al. [ŠPBV10]. Ropinski et al. enhanced their GPU raycasting framework with simulated scattering and shadowing [RDRS10]. The illumination volume was generated with slice-based front-to-back chromaticity accumulation, and in a second pass back-to-front scattering accumulation, and could be generated on the fly when the transfer function or light position was updated. All of these methods yield impressive results, do not require significant pre-computation, and run at interactive frame rates. However, all are limited to modelling a single point light source.

The previously discussed papers yield results that have proven to aid in the perception of shape and depth. The methods render a fairly good approximation of real light interaction with volumes or iso-surfaces. Another class of volume renderers takes a more physically based approach. They typically solve the light transport equation for a volume in a preprocessing step and store the result in an additional buffer. Wyman et al. [WPSH06] introduced such a method, in which the direct lighting, shadows and diffuse inter-reflection are captured in an illumination buffer. This buffer is then used to texture iso-surfaces. This method was further developed by Beason et al. [BGB*06], introducing translucency and caustics, at the cost of static lighting. Both of these approaches focus on rendering isosurfaces and are not able to do full volumetric rendering.

Salama et al. [RS07] presented a GPU framework for Monte Carlo rendering of volumetric data sets. Their work comes closest to ours in the sense that they employ stochastic rendering techniques. However, they render a number of layers using an isosurface in the volume as basis for calculating later scattering. The first pass calculates local illumination on the selected isosurface, the second pass is an ambient occlusion pass and the final layer, usually rendered with a single pass, models scattering. This is done by by starting a transmissive ray at the isosurface, scattered within a Phong lobe, and reflecting this ray from the second isosurface it hits until it exits the volume. The three layers are composited to form the final image. Our technique further differs from this approach in that it directly renders the volume in a unified way, and deliberately does not treat isosurfaces differently. At each and every ray sample, two more rays traverse the volume accumulating radiance.

The previous approaches to photo-realistic rendering are often motivated by computational limitations of both CPU, GPU and other dedicated hardware. For this reason, most of the work presents compelling, yet approximated simulation of light transport. We expect that, sparked by significant increases in computational resources of graphics hardware, these approximations will eventually be superseded

by physically-based light transport modeling. The work presented here is inspired by the excellent findings from previous authors and tries to apply the ideas in a stochastic rendering framework. To our knowledge, the work presented here is the first to demonstrate that interactive, brute force, progressive stochastic rendering of volumetric data sets in combination with arbitrary lighting conditions is possible. Our method estimates the light transport equation on the fly and does not depend on pre-calculated quantities or additional volumes. The memory footprint of our solution is hardly dependent on lighting and camera configurations. Furthermore our system is also not limited to a fixed transfer function.

3. Method

Monte Carlo Ray Tracing, or MCRT, has been around for quite some time and is known to produce high quality images. Much to our surprise, this rendering technique has not yet been applied to interactive DVR. In this section we document the approach we took to apply progressive MCRT with physically based lighting to interactive DVR. We start by giving an overview of the rendering pipeline, followed by a detailed description of how we implemented the ray-caster with physically based light transport.

Figure 2 depicts the rendering pipeline. This pipeline is executed for every iteration of the Monte Carlo (MC) algorithm. At the heart of the proposed method lies the stochastic raycaster, enhanced with physically based light transport through single scattering, which is discussed in detail in Section 3.1. The raycaster generates a High Dynamic Range (HDR) MC estimate of the light arriving at the film plane. This estimate is filtered with a separable Gaussian kernel, with standard deviation 1.5 and window 5×5 pixels, to alleviate anti-aliasing, and then added to a MC HDR estimate accumulation buffer. The MC integration step then yields a running HDR estimate by dividing the samples in the accumulation buffer by the number of iterations that have been completed so far. This estimate is tone mapped to compress it to a LDR image, and subsequently gamma corrected. From the LDR estimate, the mean running image variance is computed which is used to control the parameters of an anisotropic filter which is primarily applied during the initial iterations to remove any unwanted noise. The details of this approach are discussed in Section 3.3.

The following sections focus on the stochastic raycasting step, the KNN noise reduction filtering and finally the implementation.

3.1. Raycasting with physically based light transport

In this section we discuss our approach to include physically based light transport in a ray-caster.

In contrast to standard ray-casting, in which camera rays originate from a single point and are cast through pixel centers on the screen, our method implements a thin lens camera

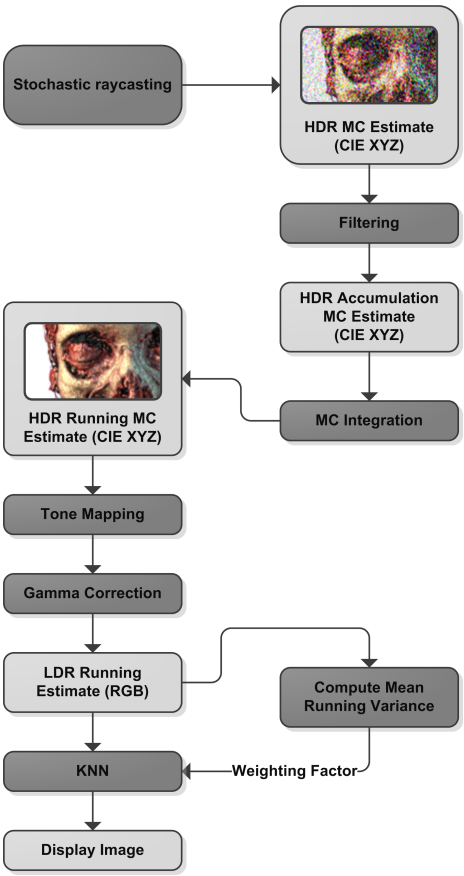

Figure 2: *A high level overview of the proposed rendering pipeline.*

model [BHK*03], see Figure 3. This model, which incorporates a finite aperture, models a real world camera. The aperture size controls the depth of field, i.e., the range of distances in which objects appear to be sharp. As a result we are able to easily incorporate out of focus effects, which are particularly effective in photo realistic rendering.

In our implementation, camera rays are constructed by sampling a point on the lens and sampling a perturbed point within the finite area of a pixel on the film plane with stratified sampling. Camera rays are first intersected with the volume's *Axis Aligned Bounding Box* (AABB), which results in a parametric range s along each camera ray. Rays that do not intersect with the volume are discarded and not propagated through the volume.

Ray-casters generally propagate through the volume with *Ray Marching*. However, when shadowing calculations are required at every step of ray marching, this method becomes prohibitively slow. To address this issue, we make use of *Woodcock tracking* to calculate the scattering point $\vec{p}_s$ along each camera ray R_{camera} [SKTM11]. The scattering point

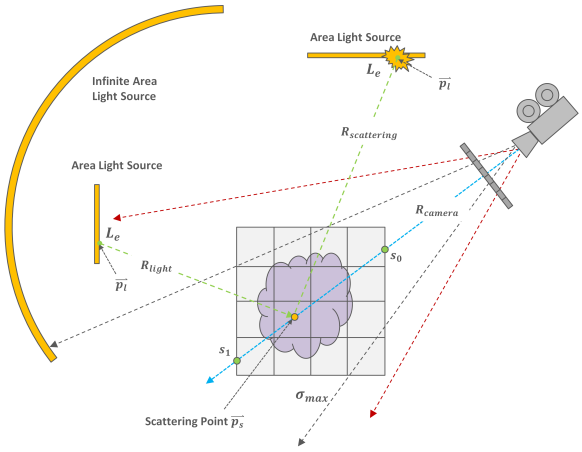

Figure 3: *A high level overview of the proposed stochastic ray-casting pipeline.*

defines an interaction event within the volume between a light or camera ray and a material particle. Woodcock tracking yields a single suitable scattering point along the ray at which the volume is sampled and direct lighting is calculated. Due to the great number of randomly sampled rays, this yields realistically lit images.

In our implementation, we estimate the direct lighting at scattering point $\vec{p}_s$ by:

1. Computing the contribution of light that flows along light ray R_{light} which is formed by connecting the scattering point $\vec{p}_s$ with a stochastically sampled point on an stochastically chosen light source
2. Finding the nearest intersection point $\vec{p}_l$ with a light source, found by sampling the phase function, which yields $R_{scattering}$, see Section 3.2. With this ray, the scattering contribution is computed.
3. Combining the two contributions using *Multiple Importance Sampling* (MIS) [Vea97].

For shadow computations we also apply Woodcock tracking. In this case, the scattering point $\vec{p}'_s$, yielded by Woodcock tracking determines if a light ray is blocked or unblocked, whether $\vec{p}'_s$ is beyond $\vec{p}_s$ or not [SKTM11]. If the ray is unblocked, the exitant radiance L_e from the light source is used in the scattering computation.

3.2. Scattering

A challenge with applying physically based lighting to volumetric data is that there is no explicit boundary surface. MCRT algorithms that deal with participating media typically are unaware of shading normals, i.e., they use phase functions for computing volumetric scattering. DVR applications typically aim at visualizing inner structures, making semi-transparent rendering crucial. The common scattering approach in DVR is to modulate the incoming light with a Phong shader in order to produce specular highlights.

The left image in Figure 4 demonstrates what happens when surface-like scattering is used throughout the volume with the use of a *Bidirectional Reflection Distribution Function* (BRDF). The BRDF, defined by $f_r(x, \vec{\omega}_o, \vec{\omega}_i)$ evaluates the fraction of incoming radiance $\vec{\omega}_i$ that is reflected in direction $\vec{\omega}_o$ at point x. The BRDF relies on a normal N, which complicates its application to volumetric data since heterogeneous volumetric data typically contains regions of well and ill defined gradient vectors. However, it became clear that though this approach provides good reflections in areas with well defined normals, the areas with ill defined normals contain substantial high frequency noise which hardly resolves over time, making it unattractive for interactive DVR.

The right image in Figure 4 demonstrates what happens when an isotropic phase function is used throughout the volume to compute the scattering. An isotropic phase function scatters light uniformly, unaffected by the direction of incident light ω_i and the viewing direction ω_o. The fraction of reflected light is defined by $p_{isotropic}(\omega \rightarrow \omega') = \frac{1}{4\pi}$. The image demonstrates that although the phase function deals well with semi transparent voxels, more opaque voxels show less detail, in the absence of surface based shading.

Our proposed method, as shown in Figure 5, is of a hybrid nature, as both BRDF scattering and volumetric scattering are mixed sensibly. As a result, volume renderings show detail through specular highlights in areas of well defined gradient vectors, at the same time including volumetric scattering in areas of ill defined gradient vectors.

In order to mix between BRDF and volumetric scattering, we propose to switch between the two based on the local voxel gradient magnitude, which is computed on the fly, and an additional weighting parameter controlling the amount of BRDF and volumetric scattering respectively. The gradient vector, used for BRDF calculations is computed on the fly with finited differences.

The method mixes surface and volumetric scattering by stochastically choosing whether a BRDF or phase function is used with the following probability:

$$P_{brdf} = 1 - e^{-|\nabla \tau_n(\vec{x})| \cdot g}$$

Where $|\nabla \tau_n(\vec{x})|$ is the normalized gradient magnitude at $\vec{x}$, and g the *gradient factor*, which controls the amount of volumetric vs. surface-like scattering. This parameter is controlled by the user, however, the mean gradient magnitude is generally a good starting point for this parameter, alleviating the user from setting this value, see Figure 6.

This method is integrated into the MC algorithm by drawing a random number ξ from a uniform distribution and comparing it to the BRDF probability P_{brdf}:

Figure 4: *Rendering on the left with only surface-based scattering (BRDF) and on the right with only volumetric scattering (phase function), but with identical transfer functions, lighting and cameras.*

Figure 5: *The proposed method integrates both surface based scattering and volumetric scattering, by mixing BRDF and phase function. Note how detail is preserved through specular highlights, while still supporting volumetric scattering.*

$$f(\vec{p}_s) = \begin{cases} f_r(x, \vec{\omega}_o, \vec{\omega}_i) & \xi < P_{brdf} \\ p_{isotropic}(\omega_o \rightarrow \omega_i) & \text{otherwise} \end{cases}$$

3.3. Interactivity

MCRT algorithms estimate the incident light on the camera film plane by sampling N light transport paths and computing the estimate with:

$$I_i = \frac{1}{N} \sum_{j=1}^{N} \frac{f_i(C_j)}{p(X_j)}$$

The quality of this estimate is measured with variance, which is the square root of the expected error. As the number of samples N increases, the expected error decreases with $O(\sqrt{N})$. The first few iterations of the MC algorithm yield high variance estimates, which results in images with high frequency noise, see Figure 8. This noise is an undesirable side effect of MCRT and should be removed as soon as possible, in order to guarantee interactivity. In the case of interactive volume rendering it is important that the image quality is high also during interaction. However, every interaction triggers a completely restart of the MC algorithm, causing unacceptable high frequency noise.

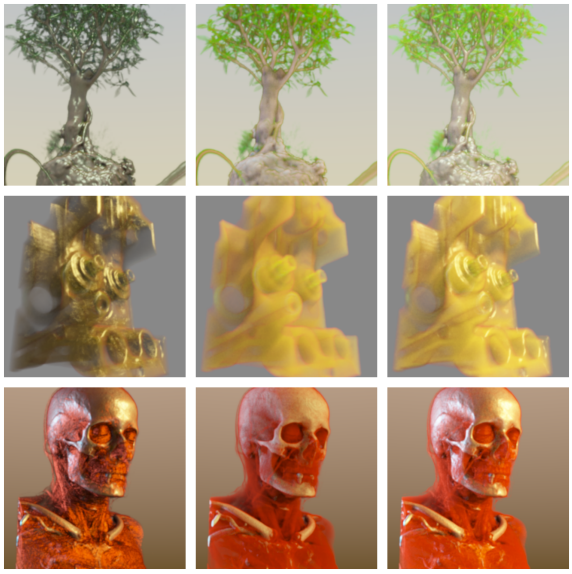

Figure 6: *Three data sets (Bonsai, Engine and Manix), rendered with fixed transfer function, lighting and camera configuration. From left to right, surface scattering, volumetric scattering and the hybrid method. The gradient factor g was automatically set, based on the mean gradient magnitude $|\bigtriangledown \tau_n(\vec{x})|$, and did not need any further tweaking.*

In order to solve this problem we propose to apply an anisotropic noise reduction filter to the final LDR MC estimate at every iteration, where the amount of noise reduction is dictated by the mean running variance of the LDR MC estimate. For our experiments, we implemented the *K Nearest Neighbor* (KNN) noise reduction filter. The KNN filter is a complex Gaussian blur filter in which the pixel weights are determined by the color similarity of neighboring pixels [BCM06]. The filer is able to run efficiently on CUDA enabled hardware, and is relatively easy controlled. In our implementation we use a window size of 7 pixels. The *lerpC* parameter of the KNN filter blends between reconstructed and original pixels, and is in our implementation controlled by the mean running variance, with the aim to control the KNN filter sensibly. As the variance decreases, so does the influence of the KNN filter. The mean variance of the LDR estimate is computed with a numerically stable algorithm, as described in [Knu97]. All other parameters of the filter

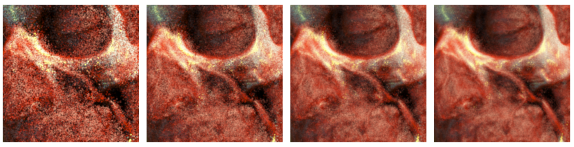

Figure 7: *This figure illustrates the problem: high frequency noise is visible in the image, particularly at the beginning of the algorithm. The images depict a close-up of the Manix data set at subsequent timepoints as the rendering is being progressively refined.*

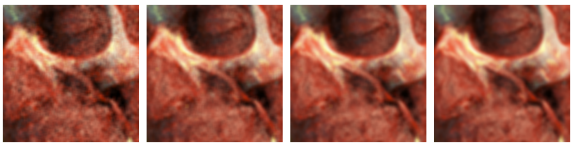

Figure 8: *The Manix data set, rendered with the same configuration, with a noise reduction post processing filter. By applying the noise reduction, the objectionable noise at the startup of the MCRT algorithm is reduced to a great extent, at the expense of blurring.*

remain constant during rendering, for more details concerning the KNN filter configuration we refer to our open source implementation.

Figure 8 depicts the development of the LDR MC estimate image with our noise reduction strategy applied. Note that especially in the first iteration the noise is greatly reduced, at the expense of slightly increased blurring.

3.4. Implementation Details

The proposed method has been implemented in Exposure Render, a complete interactive DVR framework (see Figure 9), which has been made publicly available through a Google Code project: `http://code.google.com/p/exposure-render/`. The program runs on CUDA 4.0, is written in C/C++, contains a full graphical user interface using Qt, and is shipped with sample volume data, transfer function, lighting and camera presets.

In order to support physically based lighting, we deviated from the conventional Phong based surface scattering model which does not obey reciprocity and energy conservation. We used the Fresnel blend shading model as described by Ashikmin et al. [APS00]. It models a diffuse surface with a glossy surface overlaid in a physically plausible way. It blends between diffuse and glossy reflection by incorporating an approximated Fresnel term. The Fresnel blend BRDF uses Schlick's approximation to Fresnel reflectance. As a result there is a gradual increase in reflectivity towards

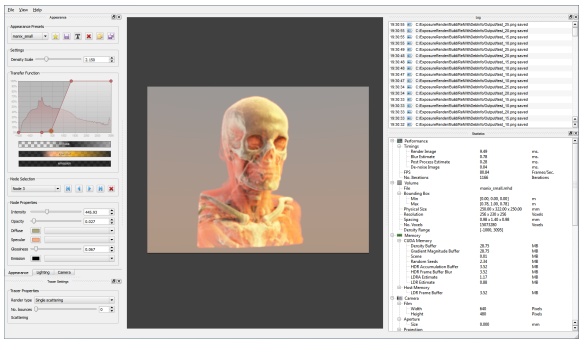

Figure 9: *Screen shot of the Exposure Render graphical user interface*

glancing angles. The Fresnel blend BRDF models rough surfaces as a collection of small microfacets. We chose the microfacet distribution developed by Blinn [Bli77] because it obeys reciprocity and is energy conserving. Furthermore, the BRDF is controlled by relatively simple parameters (e.g., diffuse refection, specular reflection and the Blinn exponent, which controls the size of the highlights), which turns out to be important in the context of transfer function specification. We extended the standard transfer function with extra channels. In each node of the transfer function, a user can specify: opacity, diffuse color, specular, glossiness and emission. We have included several illustrative examples, see Section 4. For volumetric scattering we used a straightforward isotropic phase function, however our MIS pipeline is able to support any arbitrary phase function.

Our current implementation supports area lights and environment lights. In contrast to standard ray-casting, all lights occupy a physical space in the scene, so they are actually visible by the camera. During configuration of the lighting, the image is continuously updated and the quality of the image improves progressively. The initial estimate after a few iterations is of such quality that the user can make design changes based on the presented image.

4. Results and Conclusions

In this section, we document the results of our work. We analyze the performance of the proposed method and illustrate the method's potential with example renderings.

4.1. Benchmarks

All experiments were performed on a PC with an NVIDIA GeForce GTX 470, with 877 MB of graphics memory, and an Intel ®Core^{TM}i7 CPU 920 with 12GB of RAM.

In order to determine the performance of our method we have subjected our renderer to several benchmarks. First, we

Data set	Size	Front	Left	Top
Mecanix	512 x 743 x 512	35.2	34.1	36.3
Manix	512 x 460 x 512	32.7	39.7	42.4
Engine	256 x 256 x 128	65.5	53.7	65.7
Bonsai	256 x 256 x 257	57.7	66.6	68.1
Artifix	512 x 347 x 512	47.3	40.3	50.8

Table 1: *Performance measurements expressed in the number of estimates per second for five data sets, rendered at* 800×600 *pixels from a front, left and top view respectively. Scenes are lit with an environment light and two additional area lights. Al data sets are encoded in a (16 bit) format*

document the average number of iterations per second for various datasets and viewpoints. A number of these iterations are required before the image is formed, behavior that we investigate in the second set of benchmarks, the convergence tests, based on which we draw conclusions with respect to interactivity. Thirdly we determine the convergence characteristics of a single dataset under different camera and lighting conditions.

We measured the average number of estimates that our renderer generates per second for a film resolution of 800×600 pixels for five typical data sets from three different camera angles. Table 1 shows these results. After each iteration, all rays have been finalized and their contributions have been added to the relevant pixels.

4.2. Convergence tests

Since our method progressively updates the image, we are particularly interested in the convergence characteristic of our approach. To this end we performed three types of benchmarks: We compared the convergence characteristics for various data sets, we tested how varying the aperture affects convergence and we studied the impact of the number of lights on convergence.

We measured convergence by calculating the normalized root mean squared (NRMS), the error between the running MC estimate and a fully converged MC estimate. Figure 10 shows the results of these tests. From this, it can be seen that the image converges quite rapidly and that 10% NRMS is reached in a fraction of a second for all datasets.

In order to back up our claim that our algorithm is relatively insensitive to effects such as a narrow depth of field, we plotted the convergence characteristics for a constant scene in terms of shading and lighting, but with variable aperture size. Judging by the results in figure 11, there is a marginal difference between convergence using an infinitely small aperture, medium and large aperture.

We were also interested to see how the number of lights affects convergence. We benchmarked this by rendering the

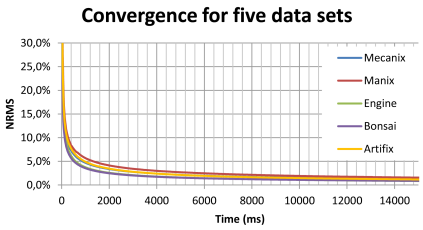

Figure 10: *This graph represents the convergence characteristics for various data sets.*

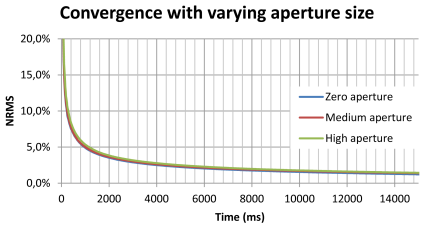

Figure 11: *This graph represents the convergence characteristics for the Manix data set, with constant lighting and shading, subject to infinitely small aperture, medium aperture and large aperture respectively.*

manix data set several times, with constant camera and shading parameters, but varying the number of lights. The results are plotted in Figure 12 and show that the number of lights have little effect on the convergence behavior.

Based on these benchmarks, we see that the renderer converges quite rapidly and is robust to different datasets, varying depth of field and increasing number of light sources. Importantly, the visualization remains interactive due to the progressive updates, yielding a high-quality volume rendering that can be used for volume and viewing parameter exploration.

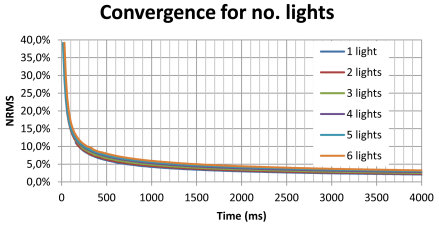

Figure 12: *This graph represents the convergence characteristics for a single data sets, with constant shading and camera parameters and increasing number of lights*

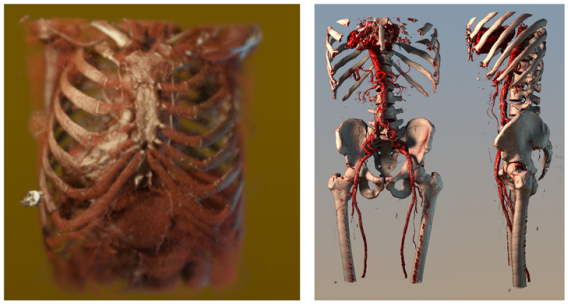

Figure 13: *The left image shows the Artifix CT data set, which is publicly available from the Osirix website. The focal distance of the camera has been fixed to the front of the rib cage, and a typical photo-studio lighting setup is chosen. The right image shows the Macoessix CT data set, also available from the Osirix website. The data set is lit with an environment light and a rather large area light, creating soft shadows.*

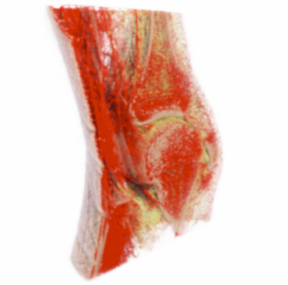

Figure 14: *Rendering of a knee. Publicly available MRI data set from the Osirix website.*

4.3. Examples

Example renderings are shown in figures 1, 6, 13, 14 and 15. The accompanying movie demonstrates the interactivity of our method. Please also see the Exposure Render website† for more example renderings, the implementation itself and demo movies.

5. Conclusions

In this paper we have presented a new GPU-based direct volume renderer that integrates stochastic ray-traced lighting. This combination enables physically-based volumetric shadows, any number of arbitrarily positioned, shaped and textured area lights and finally the simulation of a real-world camera, including its lens and aperture. With this setup, we

† http://exposure-render.googlecode.com/

Figure 15: *Rendering of the* manix *head and upper thorax dataset.*

are able to reproduce complex lighting setups that are currently used in photography studios, which adds an extra dimension of expressiveness to our volume visualizations.

Furthermore, we have shown that this type of rendering can be performed interactively on current consumer-grade graphics hardware. Because at every interaction progressive image updating starts immediately and results in an image of acceptable fidelity quite rapidly, as our benchmarks demonstrated, volume exploration is eminently possible. Our framework allows interactive adjustment of all components: Camera, lights, the transfer function and other more complex material properties. The performance of our framework, speed and memory-wise, is robust to an increase in the number of lights or the addition of effects such as depth of field. It scales graciously with volume and image size, and is able to generate very high resolution HDR images.

We do not yet simulate indirect lighting effects, such as inter-reflection. The physically-based modeling of camera, lights and shadowing was deemed to add more realism to volume rendering and was thus our first priority. Although indirect lighting will probably not have great impact on the realism of rendering volume datasets, especially medical volume datasets, we do plan to investigate its integration within our framework. Our current implementation supports spectral rendering at the expense of slower convergence. We plan to also implement spectral importance sampling in order to improve the performance.

Although we have implemented a number of techniques to improve runtime performance, we anticipate that serious improvements can be achieved by implementing more intelligent sampling schemes such as Metropolis random walks. Furthermore, we would like to investigate whether the work scheduling strategy in CUDA can be improved, in order to achieve higher SIMD efficiency.

With this work, our goal was to investigate whether graphics hardware has become fast enough to enable the interac-

tive simulation of physically-based light transport. This is part of a broader question on whether direct volume ray-tracing might soon replace direct volume rendering as the interactive volume visualization method of choice. The photo-realistic renderings our framework is able to generate, the added expressiveness and the measured and perceived interactivity of the visualizations, combined with the current trends in graphics hardware development, lead us to answer both questions positively.

6. Acknowledgements

We would like to thank the previous reviewers for providing excellent comments on our work. This work was partly supported by the Dutch Technology Foundation (STW) project 10812 *Novel pre-operative planning and intra-operative guidance system for shoulder replacement surgery*.

References

[APS00] ASHIKMIN M., PREMOŽE S., SHIRLEY P.: A microfacet-based brdf generator. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00* (2000), 65 – 74. 7

[BCM06] BUADES A., COLL B., MOREL J.: Neighborhood filters and pde?s. *Numerische Mathematik 105*, 1 (2006), 1–34. 6

[BGB*06] BEASON K., GRANT J., BANKS D., FUTCH B., HUSSAINI M.: Pre-computed illumination for isosurfaces. In *Proceedings of SPIE* (2006), vol. 6060, Citeseer, pp. 98 – 108. 3

[BHK*03] BARSKY B., HORN D., KLEIN S., PANG J., YU M.: Camera models and optical systems used in computer graphics: part ii, image-based techniques. *Computational Science and Its Applications?ICCSA 2003* (2003), 983–983. 4

[Bli77] BLINN J.: Models of light reflection for computer synthesized pictures. In *Proceedings of the 4th annual conference on Computer graphics and interactive techniques* (1977), ACM, pp. 192–198. 7

[BR98] BEHRENS U., RATERING R.: Adding shadows to a texture-based volume renderer. In *Proceedings of the 1998 IEEE symposium on Volume visualization* (1998), ACM, pp. 39 – 46. 2

[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* (New York, NY, USA, 2006), ACM, pp. 49 – 52. 2

[KJL*11] KRONANDER J., JONSSON D., LOW J., LJUNG P., YNNERMAN A., UNGER J.: Efficient visibility encoding for dynamic illumination in direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 99*, PrePrints (2011). 3

[KKH02] KNISS J., KINDLMANN G., HANSEN C.: Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics* (2002), 270 – 285. 2

[Knu97] KNUTH D.: *Art of Computer Programming, Volume 1: Fundamental Algorithms*. Addison-Wesley Professional, 1997. 6

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

[LR10] LINDEMANN F., ROPINSKI T.: Advanced light material interaction for direct volume rendering. 101–108. 3

[LV00] LOKOVIC T., VEACH E.: Deep shadow maps. In *Proc. ACM SIGGRAPH* (2000), pp. 385–392. 2

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

[MFS09] MÉNDEZ-FELIU L., SBERT M.: From obscurances to ambient occlusion: A survey. *The Visual Computer 25* (2009), 181–196. 10.1007/s00371-008-0213-4. 3

[PM08] PENNER E., MITCHELL R.: Isosurface ambient occlusion and soft shadows with filterable occlusion maps. In *Volume and Point-Based Graphics 2008* (Aug. 2008), pp. 57 – 64. 3

[RDRS10] ROPINSKI T., DORING C., REZK-SALAMA C.: Interactive volumetric lighting simulating scattering and shadowing. In *Pacific Visualization Symposium (PacificVis), 2010 IEEE* (2010), IEEE, pp. 169 – 176. 3

[Rit07] RITSCHEL T.: Fast gpu-based visibility computation for natural illumination of volume data sets. In *Short Paper Proceedings of Eurographics 2007* (Sept. 2007), Cignoni P., Sochor J., (Eds.), pp. 17 – 20. 3

[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)* (2008), pp. 17 – 24. 2

[RMSD*08] ROPINSKI T., MEYER-SPRADOW J., DIEPENBROCK S., MENSMANN J., HINRICHS K.: Interactive volume rendering with dynamic ambient occlusion and color bleeding. *Computer Graphics Forum 27*, 2 (2008), 567 – 576. 3

[RS07] REZK-SALAMA C.: GPU-based monte-carlo volume raycasting. In *Proc. of the Pacific Conference on Computer Graphics and Applications (PG07)* (2007), IEEE Computer Society, pp. 411 – 414. 2, 3

[SKTM11] SZIRMAY-KALOS L., TÓTH B., MAGDICS M.: Free path sampling in high resolution inhomogeneous participating media. *Computer Graphics Forum 30*, 1 (2011), 85–97. 4, 5

[ŠPBV10] ŠOLTÉSZOVÁ V., PATEL D., BRUCKNER S., VIOLA I.: A multidirectional occlusion shading model for direct volume rendering. *Computer Graphics Forum 29*, 3 (June 2010), 883 – 891. 3

[SPH*09] SCHOTT M., PEGORARO V., HANSEN C., BOULANGER K., BOUATOUCH K.: A directional occlusion shading model for interactive direct volume rendering. In *Computer Graphics Forum* (2009), vol. 28, Citeseer, pp. 855–862. 3

[Ste03] STEWART A.: Vicinity shading for enhanced perception of volumetric data. 3

[vA11] VAN ANTWERPEN D.: Improving simd efficiency for parallel monte carlo light transport on the gpu. 2

[Vea97] VEACH E.: *Robust Monte Carlo methods for light transport simulation*. PhD thesis, Citeseer, 1997. 5

[WPSH06] WYMAN C., PARKER S., SHIRLEY P., HANSEN C.: Interactive display of isosurfaces with global illumination. *IEEE Transactions on Visualization and Computer Graphics* (2006), 186 – 196. 3

[ZIKS98] ZHUKOV S., IONES A., KRONIN G., STUDIO G.: An ambient light illumination model. In *Rendering techniques' 98: proceedings of the Eurographics Workshop in Vienna, Austria, June 29-July 1, 1998* (1998), Springer Verlag Wien, p. 45. 2, 3