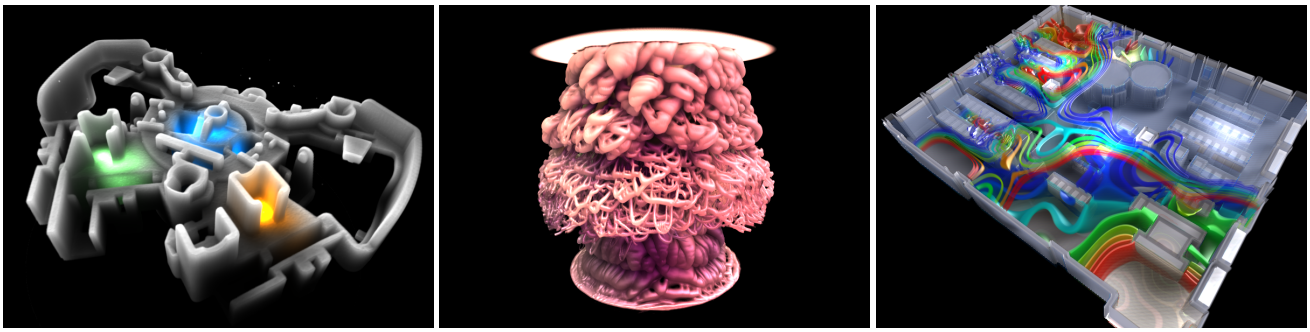
Fast Global Illumination for Interactive Volume Visualization

Yubo Zhang*
UC Davis

Kwan-Liu Ma†
UC Davis



(a) Mechanical Parts (b) Vortex Field (c) Machine Room Pressure

Figure 1: *Some volume visualization examples using our method: (a) Rendering of a CT scan of mechanical parts; (b) Rendering of a large vortex field from a turbulent flow simulation; (c) Visualization of a pressure field inside a machine room using a dynamically specified color and opacity mapping.*

Abstract

High quality global illumination can enhance the visual perception of depth cue and local thickness of volumetric data but it is seldom used in scientific visualization because of its high computational cost. This paper presents a novel grid-based illumination technique which is specially designed and optimized for volume visualization purpose. It supports common light sources and dynamic transfer function editing. Our method models light propagation, including both absorption and scattering, in a volume using a convection-diffusion equation that can be solved numerically. The main advantage of such technique is that the light modeling and simulation can be separated, where we can use a unified partial-differential equation to model various illumination effects, and adopt highly-parallelized grid-based numerical schemes to solve it. Results show that our method can achieve high quality volume illumination with dynamic color and opacity mapping and various light sources in real-time. The added illumination effects can greatly enhance the visual perception of spatial structures of volume data.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism I.3.8 [Computer Graphics]: Applications;

Keywords: global illumination, volume visualization

1 Introduction

Volume rendering is useful for visualizing field data such as numerical simulation results and medical imaging (CT, PET, MRI, etc.) datasets. It involves projecting 3D discretely sampled datasets onto 2D image planes with proper shading. Interactive and high quality illumination techniques are crucial for effective volume visualization and can enhance the visual perception of spatial structures in volume datasets. For example, global shadows can greatly improve the perception of depth cues and multiple scattering can present the local thickness of certain structures. Although simple Phong shading is fast, it only provide limited realism and may not reflect clear structures, depth orders or thickness of complex volume datasets. In recent years, advanced real-time illumination techniques have been received much attention due to the fast growing computational power of modern GPUs. These techniques can achieve better shadings such as ambient occlusion, soft shadows and scattering in real-time. However, it is still challenging to achieve multiple volumetric effects at the same time under various light sources without losing efficiency or rendering quality.

The light transport within volume usually involves emission, absorption and scattering. Such a complicated light propagation produces various global illumination effects, including single/multiple scattering and volume shadow. Although the Monte Carlo ray tracing can accurately render these volumetric effects, its prohibitively high computation expense leads itself to be impractical. Kajiya and Von Herzen [Kajiya and Von Herzen 1984] first proposed to separate the rendering procedure into two steps. In the first step, the source radiance at each voxel is estimated, and the second step is to march along the view rays to gather the source radiance. Although this method is more efficient than the Monte Carlo ray tracing, due to the dense sampling of source radiance, the radiance estimation of each voxel requires substantial computations. Volume photon mapping [Jensen and Christensen 1998] accelerates the radiance estimation but it is still hard to achieve real-time frame rate even with the assistant of a GPU. Most real-time techniques try to reduce the cost of voxel radiance estimation by using simplified approximations (e.g. [Schlegel et al. 2011]). Although such methods can produce smooth shadows, the shading quality and realism are limited without accurate and comprehensive light simulations.

In this paper, our contribution is a high quality real-time volume illumination technique which is specially designed for interactive

*e-mail:ybzhang@ucdavis.edu
†e-mail:ma@cs.ucdavis.edu

volume visualization where the mapping from data value to color and opacity at each voxel can be dynamic. Our technique models the global light propagation, absorption and multiple scattering as a unified partial differential equation (PDE). This equation can be simply solved using standard and highly-parallelized grid-based numerical methods. Our method separates the illumination modeling and light simulation, which offers the flexibility to choose different numerical schemes for solving the model equation. It is straightforward to implement it on a GPU to achieve both high quality shading and fast rendering. Its efficiency also ensures that we can interactively change the light sources and transfer functions that affect the global light transportation.

2 Related Work

The volume illumination involves solving the radiative transfer equation [Chandrasekhar 1960]. As mentioned already, Kajiya and Von Herzen [Kajiya and Von Herzen 1984] present a two-pass ray tracing method. Afterwards, several methods have been proposed to compute numerical solution of the radiative transfer equation by using either ray tracing [Levoy 1990; Lafortune and Willems 1996] or radiosity [Rushmeier and Torrance 1987]. These solutions can render various volume illumination effects with arbitrary phase functions and heterogeneous media. However, the computational cost is tremendous and the rendering can only be done offline. To improve the performance, Stam [Stam 1995] approximates the time-consuming multiple scattering effects using diffusion process, which can be simulated by either a multi-grid scheme or a finite-element blob method. Nevertheless, the speed is still far from interactive.

To render volume scattering, volumetric photon mapping (VPM) method [Jensen and Christensen 1998], an extension to standard photon mapping [Christensen et al. 2001], is proposed. In its first stage, the VPM method traces photons throughout the volume and stores them into hierarchical data structure, like kd-tree. Afterwards, the ray marching is performed and at each step the prestored photons are gathered for final radiance. This final radiance gathering step can also be computed more efficiently by photon splatting [Boudet et al. 2005; Jarosz et al. 2008]. The difficulty which prevents VPM from achieving interactive performance is the creation and maintenance of hierarchical structure. Even with the GPU-based kd-tree [Zhou et al. 2008b], rendering high-frequency volume effects with tens of millions of photons is still a non-trivial task. [Ihrke et al. 2007] present a method for efficient light transport in refractive objects but changing lighting positions still requires seconds of update time. [Kaplanyan and Dachsbacher 2010] introduced light propagation volumes for indirect lighting and its extension [Billeter et al. 2012] can deal with scattering effects in isotropic and homogeneous media.

Precomputated radiance transfer (PRT) based methods such as [Sloan et al. 2002] has the ability to render high-quality volume effects under complex lighting environments with very low cost at runtime. However, its strength is to handle complex distant lighting which is less important for volume visualization, and it is not efficient and accurate enough for approximating local and internal light sources. Although its preprocessing time and large intermediate storage cost is affordable for static volume datasets, it may not be efficient for time-varying datasets and in-situ rendering. Zhou et al. [Zhou et al. 2008a] proposed a real-time smoke rendering technique where the volume density is approximated using a set of radial basis functions (RBFs). The lighting can be solved efficiently on the RBFs but this technique is not designed for general volume data and may not scale well as the number of RBFs increases.

In recent years, several interactive techniques have been proposed to enhance the volume rendering for visualization where certain effects are desired including shadows and scattering. For example, GPU-based screen space ambient occlusion (SSAO) [Shanmugam and Arikan 2007] is a fast technique to render occlusion effects by sampling the depth buffer in screen space. It can be also applied to volume rendering [Díaz et al. 2010]. Although efficient, SSAO may not produce desired results in certain cases where depth buffer cannot offer enough information such as topology and opacity. Object space occlusion, for example [Kronander et al. 2012], offers better quality while the cost of visibility estimation is still high. Summed area table (SAT) based techniques [Díaz et al. 2010; Schlegel et al. 2011] provide an efficient way to integrate occlusion/distinction in certain volume regions and can render global shadows. But the simplified approximation cannot produce good multiple scattering effects such that the overall shading quality is limited.

3 Overview

The design of our volume rendering method is made to meet several goals:

- No preprocessing is required such that it is usable for both static and time-varying volume datasets as well as online rendering.
- The rendering speed should be real-time to ensure interactive transfer function editing for dynamic mapping from data value to color and opacity .
- It should be optimized for multiple common light sources including distant(directional) lights and point lights, which are useful for volume visualization, rather than arbitrarily complex environment lights.
- It can produce realistic high-quality shading including shadow and multiple scattering inside translucent and heterogeneous media.
- The auxiliary data structures should be similar to the volume datasets to facilitate fast light computation and sampling.

In order to meet these goals, we first divide the light simulation into N steps where N is the number of light sources. For each light source, we solve a convection equation which models the light propagation and absorption(occlusion) inside the volume domain to obtain a steady state energy distribution. The resultant energy fields are accumulated into a single light volume. Then a diffusion equation is solved for multiple scattering within the volume. Finally, we use the standard volume ray casting method to evaluate the ray integrals where the light volume is used to shade the sampling points along each ray. Fig. 2 shows a general workflow of our volume rendering technique.

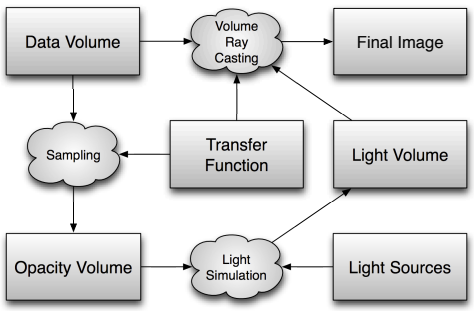


Figure 2: *The general workflow of our volume renderer.*

4 Illumination Model

Although Monte-Carlo ray tracing is the most general way to render volume data, the two-step rendering method [Kajiya and Von Herzen 1984] offers a practical strategy for efficient volume rendering where the voxel radiance is estimated at the first step and is gathered at the second ray tracing step. This framework is GPU-friendly because the second step can be achieved using s standard GPU-based volume ray casting method and the per voxel radiance can also be estimated in parallel. Many recent volume rendering techniques are also based on such framework (e.g. [Schlegel et al. 2011; Kronander et al. 2012]). Therefore, we also follow this strategy to derive a novel PDE-based volume illumination method.

4.1 Light Propagation

Our technique separately estimates the direct illumination for individual light sources. For each simple light source, the basic assumption we made is that the light propagation direction at each point is unique for direct lighting without scattering or reflection/refraction. This is obvious for simple light sources such as distant light, point light or cone light. And this assumption enables using a single value to encode light energy at each spatial location. Therefore, the direct lighting can be modeled as a time-varying light flow field. Consider an empty domain V, the rate of energy change inside V is

$$\frac{\partial}{\partial t}\int_V \rho(\mathbf{x})d\mathbf{x} = -\int_{\partial V} c\mathbf{u}(\mathbf{x})\cdot\mathbf{n}(\mathbf{x})\rho(\mathbf{x})d\mathbf{x} \tag{1}$$

where ρ is the energy density at $\mathbf{x}$, c is the speed of light, $\mathbf{u}(\mathbf{x})$ is the unit light propagation direction at $\mathbf{x}$, ∂V is the boundary of V, and $\mathbf{n}(\mathbf{x})$ is the outward unit normal on ∂V. Here $\mathbf{u}$ is light source dependent and it is discussed in later sections. The energy change rate inside V depends on how much energy comes across the boundary surface of V which can be evaluated using the right hand side of Eq. 1. If the volume of V is extremely small, we can derive the differential equation using Stokes' theorem

$$\frac{\partial}{\partial t}\rho(\mathbf{x}) = -c\mathbf{u}(\mathbf{x})\cdot\nabla\rho(\mathbf{x}) \tag{2}$$

Eq. 2 is a standard convection equation [Bejan 1984] which is solvable if a compatible boundary condition is applied.

4.2 Absorption

Light absorption is an important feature for volume visualization because materials with absorption properties can cast shadows onto other objects. If the domain V is not empty where light is partially absorbed by semi-transparent materials, we can modify Eq. 2 in order to take into account absorption

$$\frac{\partial}{\partial t}\rho(\mathbf{x}) = -c\mathbf{u}(\mathbf{x})\cdot\nabla\rho(\mathbf{x}) - \sigma_a(\mathbf{x})\rho(\mathbf{x}) \tag{3}$$

where $0 \le \sigma_a(\mathbf{x}) \le 1$ is the absorption coefficient at $\mathbf{x}$. $\sigma_a(\mathbf{x})$ can be obtained from the opacity volume, which depends on the transfer function of the original volume data. A $\sigma_a(\mathbf{x}) = 0$ means light passes through $\mathbf{x}$ freely while a $\sigma_a(\mathbf{x}) = 1$ indicates light is completely blocked at $\mathbf{x}$.

4.3 Scattering

Multiple scattering is another desired feature in volume visualization which enhances the visual perception of local thickness of volumetric objects. While accurately simulating the multiple scattering requires heavy computational cost, over-simplified approximation cannot produce visually plausible results. As stated in [Stam 1995], multiple scattering can be approximated by a diffusion process in the limit of an optically thick medium. This offers a practical way to model the scattering effect. Therefore, we integrate a diffusion term into Eq. 3

$$\frac{\partial}{\partial t}\rho(\mathbf{x}) = -c\mathbf{u}(\mathbf{x})\cdot\nabla\rho(\mathbf{x}) - \sigma_a(\mathbf{x})\rho(\mathbf{x}) + \nabla\cdot(D\nabla\rho(\mathbf{x})) \tag{4}$$

where D is the diffusion tensor. For volume visualization, we assume the scattering is isotropic and let the diffusion tensor $D = \sigma_s I$ where σ_s is the scattering coefficient, because the contribution of anisotropic scattering to the visual perception is not obvious. Therefore Eq. 4 can be simplified as

$$\frac{\partial}{\partial t}\rho(\mathbf{x}) = -c\mathbf{u}(\mathbf{x})\cdot\nabla\rho(\mathbf{x}) - \sigma_a(\mathbf{x})\rho(\mathbf{x}) + \sigma_s\nabla^2\rho(\mathbf{x}) \tag{5}$$

Eq. 5 is a convection-diffusion equation that models the light propagation, absorption and scattering within the entire volume domain Ω. However, it still needs proper initial and boundary conditions in order to be solved numerically. These conditions depend on light sources and are discussed in the following subsections.

4.4 Distant Lights

Distant parallel lighting is commonly used in volume visualization where the direction of the light flow $\mathbf{u}$ is constant. Assume the light direction is $\mathbf{u}_D$ and the energy density of the distant light is ρ_D, the boundary condition for Eq. 5 is a hybrid Dirichlet-Neumann condition [Cheng and Cheng 2005]

$$\rho(\mathbf{x})|_{\partial\Omega} = \rho_D, \quad \text{if } \mathbf{n}(\mathbf{x})\cdot\mathbf{u}_D < 0, \tag{6}$$

$$\left.\frac{\partial\rho(\mathbf{x})}{\partial\mathbf{n}}\right|_{\partial\Omega} = 0, \quad \text{otherwise.} \tag{7}$$

where $\mathbf{x} \in \partial\Omega$ is the boundary of the entire volume domain Ω and $\mathbf{n}(\mathbf{x})$ is the unit outward normal at $\mathbf{x}$. Eq. 6 specifies the boundary condition for the surface region that is facing the light. The Neumann boundary condition used in Eq. 7 means the light leaves the volume boundary surface without any occlusion. Fig. 3 shows the relation between the light source and the boundary condition.

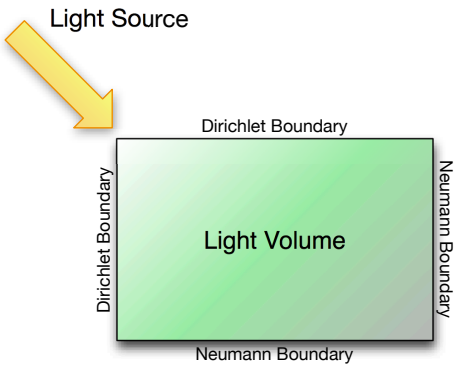


Figure 3: *A 2D illustration of a hybrid Dirichlet-Neumann boundary condition. The Dirichlet boundary condition is applied at the volume boundary whose surface normals are facing the light source. The boundary values are set to the light intensity at the surface. The Neumann boundary condition is applied at the rest part of the volume boundary where the derivatives of light intensity along the surface normals are set to zero.*

4.5 Point Lights

Point lights are also useful for volume visualization. In addition to using point light sources as external light sources, we may also put point lights inside the volume domain such that certain highly occluded internal structures can be illuminated and highlighted. Therefore, the boundary condition for a point light source can be classified as two different cases. Assume $\mathbf{x}_P$ is the position of the light source. If $\mathbf{x}_P$ is outside the volume region (i.e. $\mathbf{x}_P \notin \Omega$), we can use the following boundary condition

$$\rho(\mathbf{x})|_{\partial\Omega} = G(\mathbf{x},\mathbf{x}_P)\rho_P, \quad \text{if } \mathbf{n}(\mathbf{x})\cdot(\mathbf{x}-\mathbf{x}_P) < 0, \tag{8}$$

$$\left.\frac{\partial\rho(\mathbf{x})}{\partial\mathbf{n}}\right|_{\partial\Omega} = 0, \quad \text{otherwise.} \tag{9}$$

where $G(\mathbf{x},\mathbf{x}_P)$ is a falloff function for point light sources. In our implementation, we use a simple Gaussian function for G. If $\mathbf{x}_P$ is inside the volume region (i.e. $\mathbf{x}_P \in \Omega$), a pure Neumann boundary condition can be used

$$\left.\frac{\partial\rho(\mathbf{x})}{\partial\mathbf{n}}\right|_{\partial\Omega} = 0, \quad \forall\mathbf{x}\in\partial\Omega. \tag{10}$$

We also need to specify an interior boundary condition at the position of the light source

$$\rho(\mathbf{x}_P) = \rho_P. \tag{11}$$

In general, Eq. 5 to 11 form a complete system for volume light propagation under a distant or point light source. The existence and uniqueness of the steady state solution to the system ensures that we can solve it numerically. The detailed numerical schemes we used are discussed in the next section.

5 Solving the Model Equation

Solving Eq. 5 consist of several steps. We first split the equation into a convection step and a diffusion step. The convection step is solved for each light source using a first-order upwind scheme [Courant et al. 1952]. The results are accumulated into a single energy volume and we then solve the diffusion step for scattering effects. Details are discussed in this section.

5.1 Splitting

We use finite difference method to solve Eq. 5. In order to minimize the computational cost, instead of solving the entire equation for each light source, we split Eq. 5 into a convection equation

$$\frac{\partial}{\partial t}\rho(\mathbf{x}) = -c\mathbf{u}(\mathbf{x})\cdot\nabla\rho(\mathbf{x}) - \sigma_a(\mathbf{x})\rho(\mathbf{x}) \tag{12}$$

and a diffusion equation

$$\frac{\partial}{\partial t}\rho(\mathbf{x}) = \sigma_s\nabla^2\rho(\mathbf{x}) \tag{13}$$

where Eq. 13 can be made independent of light sources with a pure Neumann boundary condition. Therefore, we can first solve Eq. 12 for each light source and combine the solution together. Then we only need to solve a single diffusion equation instead of multiple ones. This can significantly reduce the computational cost when there are multiple light sources, because solving Eq. 13 usually requires certain iterative linear solver such as Gauss-Seidel iteration or conjugate gradient method which has much higher cost than solving Eq. 12.

5.2 Volume Discretization

Before solving Eq. 12 and 13, we have to discretize the volume domain. For a regular grid volume dataset, we can use a similar grid for light simulation. The grid resolution Δx can be the same as the volume dataset or lower. Certain amount of down sampling can increase the performance of light simulation while maintain the shading quality. The energy density $\rho(\mathbf{x})$ is then discretized into $\rho_{i,j,k} = \rho(\mathbf{x}_{i,j,k})$ where $0 \le i < N_x$, $0 \le j < N_y$, $0 \le k < N_z$ and $\mathbf{x}_{i,j,k} = (i\Delta x, j\Delta x, k\Delta x)$. The discretizations of other field quantities are similar to the energy density.

5.3 Light Volume Initialization

After volume discretization, we have to initialize the solution $\rho_{i,j,k}$. For solving Eq. 12, we only need to set the boundary values according to Eq. 6-10 and set the interior values to zeros except in the case of Eq. 11 which is discussed in this subsection. When initializing the energy density for an internal point light source, we set the voxels which are close to the center of the light source to the original light intensity

$$\rho_{i,j,k} = G(\mathbf{x}_{i,j,k},\mathbf{x}_P)\rho_P, \tag{14}$$

where

$$\left|\mathbf{x}_{i,j,k} - \mathbf{x}_P\right| < 2\Delta x. \tag{15}$$

For internal point lights, the region that satisfies Eq. 15 will not be updated during the light simulation and is treated as an internal boundary condition.

5.4 The Upwind Scheme for Convection

To solve Eq. 12, we use a first-order upwind scheme which has regular memory access patterns on GPUs. Upwind schemes are originated from [Courant et al. 1952], where the finite difference stencil points are chosen based on the local wave direction. The scheme uses stencil points along the inverse wave direction in order to achieve stable results when the CFL condition $C = c\Delta t/\Delta x \le 1$ is satisfied. In our implementation, we let $c\Delta t = \Delta x/2$ such that $C = c\Delta t/\Delta x = 1/2$. Assume we have the solution at time t, say $\rho_{i,j,k}(t)$, the solution at the next time step can be evaluated through

$$\rho_{i,j,k}(t+\Delta t) = (1-\sigma_a(\mathbf{x}))\left(\rho_{i,j,k}(t) - \frac{1}{2}\Delta x\Delta\rho_{i,j,k}(t)\right), \tag{16}$$

where

$$\Delta\rho_{i,j,k}(t) = u_{i,j,k}\left(\rho_{i,j,k}\right)_x + v_{i,j,k}\left(\rho_{i,j,k}\right)_y + w_{i,j,k}\left(\rho_{i,j,k}\right)_z. \tag{17}$$

Here $(u_{i,j,k}, v_{i,j,k}, w_{i,j,k})$ is the light direction at the voxel $\Omega_{i,j,k}$ and $\left(\rho_{i,j,k}\right)_x$, $\left(\rho_{i,j,k}\right)_y$, $\left(\rho_{i,j,k}\right)_z$ are the three components of the density gradient which depend on the light direction

$$\left(\rho_{i,j,k}\right)_x = \begin{cases} \frac{\rho_{i+1,j,k}(t)-\rho_{i,j,k}(t)}{\Delta x}, & \text{if } u_{i,j,k} < 0, \\ \frac{\rho_{i,j,k}(t)-\rho_{i-1,j,k}(t)}{\Delta x}, & \text{if } u_{i,j,k} > 0. \end{cases} \tag{18}$$

$$\left(\rho_{i,j,k}\right)_y = \begin{cases} \frac{\rho_{i,j+1,k}(t)-\rho_{i,j,k}(t)}{\Delta x}, & \text{if } v_{i,j,k} < 0, \\ \frac{\rho_{i,j,k}(t)-\rho_{i,j-1,k}(t)}{\Delta x}, & \text{if } v_{i,j,k} > 0. \end{cases} \tag{19}$$

$$\left(\rho_{i,j,k}\right)_z = \begin{cases} \frac{\rho_{i,j,k+1}(t)-\rho_{i,j,k}(t)}{\Delta x}, & \text{if } w_{i,j,k} < 0, \\ \frac{\rho_{i,j,k}(t)-\rho_{i,j,k-1}(t)}{\Delta x}, & \text{if } w_{i,j,k} > 0. \end{cases} \tag{20}$$

16 is evaluated repeatedly until a steady state solution of Eq. 5 is obtained which satisfies

$$-c\mathbf{u}(\mathbf{x})\cdot\nabla\rho(\mathbf{x}) - \sigma_a(\mathbf{x})\rho(\mathbf{x}) = 0 \tag{21}$$

together with the light source dependent boundary conditions discussed in Section 4.4 and 4.5. To achieve maximum convergence speed, we update the energy volume layer by layer using 16 along the light propagation direction. Usually 4-5 iterations are required for each layer in order to achieve good results.

5.5 Light Field Composition

Before solving Eq. 13 for scattering effects, we can composite the solutions of Eq. 12 for all light sources. This can be done by simply combining all the energy density fields together with corresponding light color into a single light color field

$$L(\mathbf{x}) = \sum_{k=1}^{N} (\rho_k(\mathbf{x}) L_k) \tag{22}$$

where ρ_k is the solution of Eq. 12 under the kth light source, L_k is the color of the kth light source and L is the composed light color field.

5.6 The Diffusion Process

We now simulate the scattering effect by solving Eq. 13. Because the energy density fields are already combined in the previous step as discussed in Section 5.5, we need to modify Eq. 13 to a vector form

$$\frac{\partial}{\partial t} L(\mathbf{x}) = \hat{\sigma}_s \nabla^2 L(\mathbf{x}) \tag{23}$$

where $\hat{\sigma}_s$ consist of three scattering coefficients for different color channels. Although an accurate multigrid solver can be used, Eq. 23 is solved implicitly using the Gauss-Seidel iteration method [Jeffreys and Jeffreys 1966] which has less memory usage. The solution at $t = 1$ is used as the final illumination volume.

6 Rendering

After we get the final illumination volume, a standard volume ray casting method is adopted to render the original volume data. During the volume ray integral evaluation, only one extra texture sampling operation is required in order to lookup the illumination value $L(\mathbf{x})$ at each sample point. This value is then combined with the Phong shading result to evaluate the final light color at the sample point

$$L_{final}(\mathbf{x}) = L(\mathbf{x}) \left(I_a(\mathbf{x}) + I_d(\mathbf{x}) + I_s(\mathbf{x})\right) \tag{24}$$

where I_a, I_d and I_s are the intensity of ambient, diffuse and specular lighting, respectively. $L_{final}(\mathbf{x})$ is then used to shade the material color which depends on the original data and the user specified transfer function.

7 Implementation

The general procedure of our volume rendering technique is listed in Algorithm 1. The user interface of our renderer enables users to edit transfer function, light sources and rendering parameters in real-time. For the light simulation, the resolution of the light volume can also be adjusted by users. Normally a light volume with a half resolution of the original volume data can already produce visually plausible results. For certain cases where high computational cost is required (e.g. many light sources, high resolution light volumes) and real-time frame rate is crucial, we can also compute the light volume in multiple frames due to the iterative structure of our method.

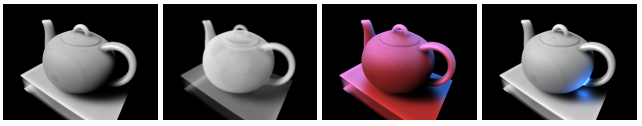
Algorithm 1 Grid-based Volume Illumination

```
for all frame do
    if transfer function is updated then
        update the opacity volume
    for all light sources do
        if the opacity volume or the current light is changed then
            initialize the light volume
            simulate light absorption
    if the opacity volume or any light is changed then
        compose the light volume
        simulate light scattering
    render the current frame using ray casting
```

We use NVIDIA CUDA to simulate the light propagation and use OpenGL to render the final image. At first, the user specified transfer function are copied into the GPU memory. Then the opacity volume is calculated in parallel based on the transfer function and the resampled data volume which has the same resolution as the light volume. The initialization of light volume, the upwind scheme for convection and the diffusion iteration are also implemented using parallel kernel functions. The final light volume is copied to a OpenGL 3D texture and is sampled in a GLSL volume ray casting program. Because all the computations except the final ray casting are per voxel operations, it is extremely easy to implement using parallel GPU programs. The rendering quality and performance of our method are discussed in the next section.

8 Results

8.1 Lights and Soft Shadows



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

Figure 4: *Teapot rendering with different opacity values and light sources.*

We did several simple experiments to validate our illumination method. Fig. 4(a) and 4(b) shows a teapot rendering with different opacity values. It is shown that objects with higher opacity values can completely occlude the incoming light and cast dark shadows while semi-transparent objects just partially absorb the light energy and cast lighter shadows. We also render the teapot under different light sources. Fig. 4(c) and 4(d) shows the difference between distant and point light sources. For volume visualization, single light source may not fully illuminate the volume data due to the occlusion and dark shadows. Therefore, multiple distant light sources are useful for better illumination and distinct light colors may also help users to identify different regions of the of volume data. Fig. 5 is an example of using distant light sources. On the other hand, point light sources are also useful for volume visualization because it can be used to highlight certain part of the volume data. An example is shown in Fig. 1(a) where a mechanical part is rendered under four light sources. It is noticeable that the shadows generated by our illumination technique are soft shadows rather than hard shadows. The reason is that the upwind scheme we used in Section 5.4 is a first order scheme. Although stable, its numerical dissipation will smear the solution of the convection equation. Fortunately, this is a

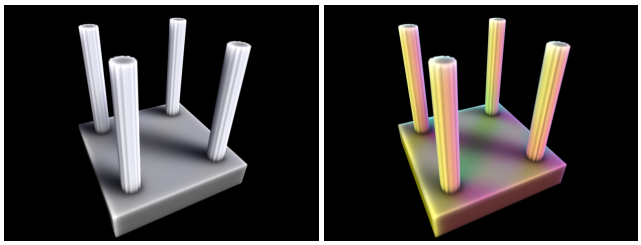
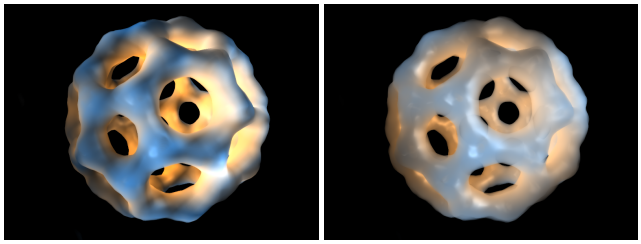


Figure 5: *Pillar rendering under distant lights. The left image shows the rendering result under a single distant light source where soft shadows are produced. The right image shows the rendering result under four distant light sources with different colors (white, red, green, blue) where colored soft shadows are produced.*

desired feature because soft shadows are more natural in real world. And the numerical dissipation can also be reduced by using higher resolution light volumes or high order upwind schemes.

8.2 Scattering Test

The scattering effect is tested by comparing the rendering results by using different scattering coefficients. In Fig. 6, a buckyball is rendered with $\sigma_s = 0$ and $\sigma_s = 10$. It is shown that a translucent



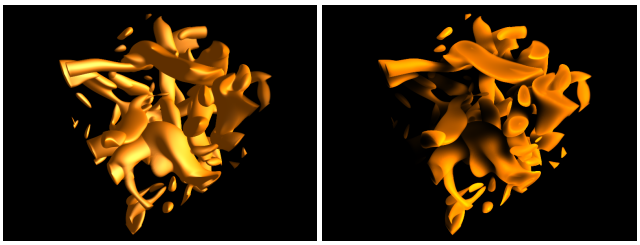
(a) $\sigma_s = 0$ (b) $\sigma_s = 10$

Figure 6: *Experiment with scattering simulation. A buckyball is rendered with a blue distant light and an internal orange point light. The left image shows the rendering result without scattering and the right one shows the rendering result with lower opacity and strong scattering effects* $\sigma_s = 10$.

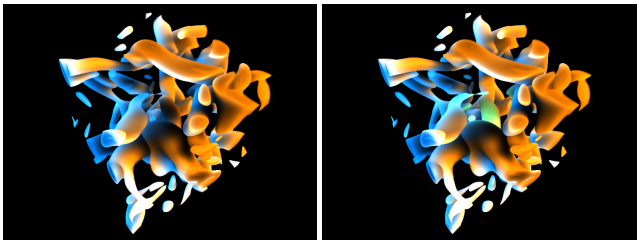
object rendered with a lower opacity and a higher σ_s value can have a highly realistic look. This example also demonstrates that the internal structure is illuminated using an orange point light. Fig. 7 is another example which shows how the σ_s value can affect the rendering results. Due to the multiple scattering, the dark smoke regions can be illuminated which offers a more realistic look. This example also shows that we can use the scattering coefficient to control the softness of the shadow.

8.3 More Examples

For highly occluded data such as Fig. 8, soft shadows can improve the visual perception of depth orders compared to simple Phong shading. In this example, we first render the vortex using the local illumination. The we use our illumination method to render the vortex data using an orange distant light and the result show detailed soft shadows. In order to illuminate the other side of the data, we use another distant light with blue color. In the bottom-right image of Fig. 8, we also added a green point light to illuminate the central area and the global spatial relations of the vortex data are clearly visualized. Fig. 9 is another example of how global illumi-



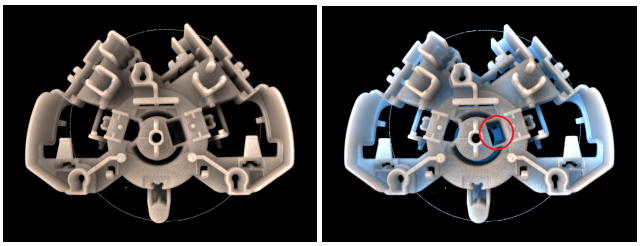
(a) Phong Shading (b) Distant Light



(c) Two Distant Lights (d) Distant+Point Lights

Figure 8: *A highly occluded vortex dataset rendered with different lighting environments.*

nation can help volume visualization where certain overshadowed areas are illuminated through multiple scattering. Fig. 1(c) shows a



(a) Front Lighting (b) Front+Back Lighting

Figure 9: *Mechanical part rendered with different light sources. The back lighting with multiple scattering effect highlighted the area which is overshadowed due to the occlusion.*

more complex example where the dataset contains a machine room together with a flow pressure field. In this example, point lights are used to highlight certain part of the data. Fig. 10 shows more examples of our illumination method where all the volume datasets are rendered with realistic shadings. The depth orders as well as local thickness of structures are clearly presented.

8.4 Performance

The performance of all the rendering examples is listed in Table 1. The volume data size, the number of light sources, the time used for light simulation and the volume rendering time per frame are listed. The light simulation is only needed after the lighting parameters or the opacity map has been changed. Therefore the frame rate is slower when users are adjusting the parameters that affect the global light transportation. Otherwise, the rendering time is almost the same as the classical volume ray casting with the Phong model. We also measure the light simulation time using different light volume resolutions. Table 2 shows that in order to achieve fast light simulation, a light volume with a resolution less than $128 \times 128 \times 128$ should be used. Compared with previous work, our



(a) $\sigma_s = 0$ (b) $\sigma_s = 1$ (c) $\sigma_s = 10$

Figure 7: *A smoke dataset is rendered under two distant light sources using different scattering coefficients. Left:* $\sigma_s = 0$*; Middle:* $\sigma_s = 1$*; Right:* $\sigma_s = 10$*. The difference is obvious at the lower left corner of the smoke plume where the density (opacity) is high.*

Dataset	Data Volume Size	Light Volume Size	No. Lights	Light Simulation	Rendering/Frame
Parts	$504 \times 504 \times 225$	$126 \times 126 \times 57$	4	83ms	34ms
Vortex (teaser)	$256 \times 256 \times 256$	$128 \times 128 \times 128$	5	91ms	27ms
Machine Room	$417 \times 345 \times 60$	$209 \times 173 \times 30$	4	86ms	21ms
Teapot	$256 \times 256 \times 178$	$128 \times 128 \times 89$	2	54ms	19ms
Pillars	$100 \times 100 \times 100$	$50 \times 50 \times 50$	4	49ms	17ms
Buckyball	$64 \times 64 \times 64$	$32 \times 32 \times 32$	2	22ms	15ms
Vortex	$128 \times 128 \times 128$	$128 \times 128 \times 128$	3	79ms	17ms
Smoke	$300 \times 300 \times 150$	$150 \times 150 \times 75$	2	54ms	25ms
Bonsai	$256 \times 256 \times 256$	$128 \times 128 \times 128$	1	54ms	21ms
Blood Vessels	$256 \times 256 \times 256$	$128 \times 128 \times 128$	1	52ms	21ms
Foot	$128 \times 128 \times 128$	$64 \times 64 \times 64$	1	23ms	17ms

Table 1: *The performance measurement of our rendering technique. The time is measured on a laptop with an Intel 1.7GHz CPU and an NVIDIA Geforce 460M GPU.*

Light Volume Size	Absorption	Scattering
$16 \times 16 \times 16$	6ms	3ms
$32 \times 32 \times 32$	8ms	4ms
$64 \times 64 \times 64$	11ms	7ms
$128 \times 128 \times 128$	34ms	18ms
$256 \times 256 \times 256$	182ms	57ms

Table 2: *Measurement of simulation time for different light volume resolutions. The aneurism dataset is used.*

method is not as fast as the high-quality precomputation based techniques, where changing lighting environments does not increase the rendering cost, and the rendering performance is dominated by the volume ray casting itself. However, our technique does not need precomputation and can produce rendering results with comparable high quality, which is more practical in certain cases.

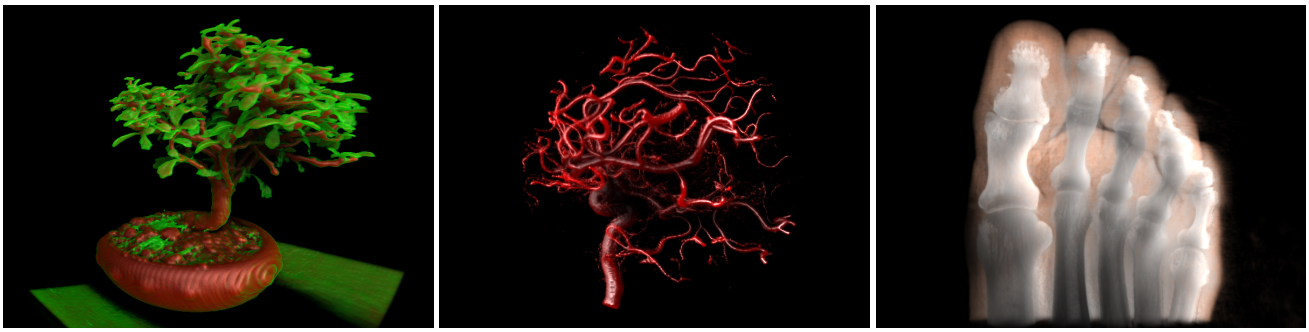
9 Conclusion

In this paper, we present a grid-based illumination method for volume visualization where a convection-diffusion equation is used to model the light propagation, absorption and scattering of optically thick medium within the volume data. The illumination model can be efficiently solved using classical numerical methods and it also produces high quality realistic rendering results which improve the visual perception on the volume data. One limitation of our current implementation is that we store the whole volume data in the GPU memory. Therefore the volume size is constrained by the total GPU memory. But it is straightforward to implement our method on clusters due to the parallel algorithm we used. Another issue is that the light simulation time could be much longer on less powerful GPUs. Fortunately, our illumination technique is iterative such that we can execute a fixed number of numerical iterations at each frame. In this way, the real-time frame rate can be maintained, which is crucial for interactive visualization. Our model also assumes the volume is optically thick. Although certain datasets do not satisfy this condition, the rendering results still look plausible for visualization purpose.

In the future, we will try to optimize our method for extremely large volume data and also time-varying data. Certain hierarchical data structures should be designed for extremely large volume data. For time-varying data, we can make use of the previous simulated light volumes as an initial guess of the new light volumes such that the number of iterations for solving the convection equation can be reduced due to the continuity of the time-varying data. We will also support more light types. For example, cone light can be implemented by simply limiting the emission angle of a point light. In addition, our illumination model can be applied to volume datasets on irregular meshes such as tetrahedron meshes without changing the model equation. The same equation can be solved using standard numerical methods such as finite element method, which is good for handling irregular meshes.

Acknowledgements

This research was sponsored in part by the U.S. National Science Foundation through grants OCI 0905008, OCI 0850566, and OCI 0749227, and also by the U.S. Department of Energy through grants DE-FC02-06ER25777 and DE-FC02-12ER26072, program manager Lucy Nowell.



(a) Bonsai (b) Blood Vessels (c) Foot

Figure 10: *More rendering examples from CT scan data. The depth orders of leaves (left) and vessels (middle) which may not be easily distinguished are clearly presented using our method. The local thickness of bones (right) is also perceivable using back lighting.*

References

BEJAN, A. 1984. *Convection heat transfer*. John Wiley and Sons Inc., New York, NY.

BILLETER, M., SINTORN, E., AND ASSARSSON, U. 2012. Real-time multiple scattering using light propagation volumes. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ACM, New York, NY, USA, I3D '12, 119–126.

BOUDET, A., PITOT, P., PRATMARTY, D., AND PAULIN, M. 2005. Photon splatting for participating media. In *Proc. of GRAPHITE*, 197–204.

CHANDRASEKHAR, S. 1960. *Radiative Transfer*. Dover Publications, New York.

CHENG, A., AND CHENG, D. 2005. Heritage and early history of the boundary element method. *Engineering Analysis with Boundary Elements 29*, 3, 268–302.

CHRISTENSEN, P., JENSEN, H., AND SUYKENS, F. 2001. A practical guide to global illumination using photon mapping. In *SIGGRAPH Course Notes,*, 60–90.

COURANT, R., ISAACSON, E., AND REES, M. 1952. On the solution of nonlinear hyperbolic differential equations by finite differences. *Communications on Pure and Applied Mathematics 5*, 3, 243–255.

DÍAZ, J., VÁZQUEZ, P., NAVAZO, I., AND DUGUET, F. 2010. Real-time ambient occlusion and halos with summed area tables. *Computers & Graphics 34*, 4, 337–350.

IHRKE, I., ZIEGLER, G., TEVS, A., THEOBALT, C., MAGNOR, M., AND SEIDEL, H.-P. 2007. Eikonal rendering: efficient light transport in refractive objects. *ACM Trans. Graph. 26*, 3 (July).

JAROSZ, W., ZWICKER, M., AND JENSEN, H. W. 2008. The beam radiance estimate for volumetric photon mapping. *Computer Graphics Forum 27*, 2 (4), 557–566.

JEFFREYS, S., AND JEFFREYS, B. 1966. *Methods of mathematical physics*. Cambridge University Press.

JENSEN, H. W., AND CHRISTENSEN, P. H. 1998. Efficient simulation of light transport in scences with participating media using photon maps. In *Proc. of SIGGRAPH*, ACM, 311–320.

KAJIYA, J. T., AND VON HERZEN, B. P. 1984. Ray tracing volume densities. *SIGGRAPH Comput. Graph. 18*, 165–174.

KAPLANYAN, A., AND DACHSBACHER, C. 2010. Cascaded light propagation volumes for real-time indirect illumination. In *Proc. of I3D*, ACM, 99–107.

KRONANDER, J., JONSSON, D., LOW, J., LJUNG, P., YNNERMAN, A., AND UNGER, J. 2012. Efficient visibility encoding for dynamic illumination in direct volume rendering. *Visualization and Computer Graphics, IEEE Transactions on 18*, 3 (march), 447 –462.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1996. Rendering participating media with bidirectional path tracing. In *Proc. of EGWR*, 91–100.

LEVOY, M. 1990. Efficient ray tracing of volume data. *ACM Trans. Graph. 9*, 245–261.

RUSHMEIER, H. E., AND TORRANCE, K. E. 1987. The zonal method for calculating light intensities in the presence of a participating medium. *SIGGRAPH Comput. Graph. 21*, 293–302.

SCHLEGEL, P., MAKHINYA, M., AND PAJAROLA, R. 2011. Extinction-based shading and illumination in gpu volume ray-casting. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12 (dec.), 1795 –1802.

SHANMUGAM, P., AND ARIKAN, O. 2007. Hardware accelerated ambient occlusion techniques on gpus. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, ACM, New York, NY, USA, I3D '07, 73–80.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21*, 527–536.

STAM, J. 1995. Multiple scattering as a diffusion process. In *Proc. of EGWR*, 41–50.

ZHOU, K., REN, Z., LIN, S., BAO, H., GUO, B., AND SHUM, H.-Y. 2008. Real-time smoke rendering using compensated ray marching. *ACM Trans. Graph. 27*, 36:1–36:12.

ZHOU, K., HOU, Q., WANG, R., AND GUO, B. 2008. Real-time kd-tree construction on graphics hardware. *ACM Trans. Graph. 27*, 126:1–126:11.