Eurographics Symposium on Parallel Graphics and Visualization (2009)
J. Comba, K. Debattista, and D. Weiskopf (Editors)

Parallel Solution to the Radiative Transport

László Szirmay-Kalos[1], Gábor Liktor[1], Tamás Umenhoffer[1], Balázs Tóth[1], Shree Kumar[2] and Glenn Lupton[3]

[1]Budapest University of Technology and Economics, Hungary
[2]Hewlett-Packard, India, [3]Hewlett-Packard, USA

Abstract

This paper presents a fast parallel method to compute the solution of the radiative transport equation in inhomogeneous participating media. The efficiency of the method comes from different factors. First, we use a novel approximation scheme to find a good guess for both the direct and the scattered component. This scheme is based on the analytic solution for homogeneous media, which is modulated by the local material properties. Then, the initial approximation is refined iteratively. The iterative refinement is executed on a face centered cubic grid, which is decomposed to blocks according to the available simulation nodes. The implementation uses CUDA and runs on a cluster of GPUs. We also show how the communication bottleneck can be avoided by not exchanging the boundary conditions in every iteration step.

1. Introduction

The multiple-scattering simulation in participating media is one of the most challenging problems in computer graphics, radiotherapy, PET/SPECT reconstruction, etc., where accurate estimates of both the direct and indirect terms are needed not only to produce nice images, but also to reconstruct volume data from projected measurements or to make decisions on the placement of the radiation source in the body during radiotherapy treatment. As these applications rely on intensive man-machine communication, the system is expected to respond to user actions interactively and to deliver simulation results in seconds rather than in hours, which is typical in CPU based solutions.

Such simulation should solve the *radiative transport equation* that expresses the change of radiance $L(\vec{x},\vec{\omega})$ at point $\vec{x}$ and in direction $\vec{\omega}$:

$$\vec{\omega}\cdot\vec{\nabla}L = \left.\frac{\mathrm{d}L(\vec{x}+\vec{\omega}s,\vec{\omega})}{\mathrm{d}s}\right|_{s=0} =$$

$$-\sigma_t(\vec{x})L(\vec{x},\vec{\omega})+\sigma_s(\vec{x})\int_{\Omega'} L(\vec{x},\vec{\omega}')P(\vec{\omega}',\vec{\omega})\mathrm{d}\omega', \qquad (1)$$

where σ_t is the *extinction coefficient* describing the probability of collision in a unit distance. When collision happens, the photon is either scattered or absorbed, so the extinction coefficient is broken down to *scattering coefficient* σ_s and *absorption coefficient* σ_a:

$$\sigma_t(\vec{x}) = \sigma_a(\vec{x})+\sigma_s(\vec{x}).$$

The probability of reflection given that collision happened is called the *albedo* of the material:

$$a = \frac{\sigma_s}{\sigma_t}.$$

If reflection happens, the probability density of the reflected direction is defined by *phase function* $P(\vec{\omega}',\vec{\omega})$. The extent of anisotropy is usually expressed by the mean cosine of the phase function:

$$g = \int_{\Omega'} (\vec{\omega}'\cdot\vec{\omega})P(\vec{\omega}'\cdot\vec{\omega})\mathrm{d}\omega'.$$

In homogeneous media volume properties σ_t, σ_s, and $P(\vec{\omega}',\vec{\omega})$ do not depend on position $\vec{x}$. In inhomogeneous media these properties depend on the actual position.

In case of measured data, material properties are usually stored in a 3D voxel grid, and are assumed to be constant or linear between voxel centers. If the diameter of the region represented by a voxel is Δ, then the total extinction is worth representing by a new parameter that is called the *opacity* and is denoted by α:

$$\alpha = 1-e^{-\sigma_t\Delta} \approx \sigma_t\Delta. \qquad (2)$$

Radiance $L(\vec{x},\vec{\omega})$ is often expressed as a sum of two terms, the *direct term* L_d that represents unscattered light, and the

media term L_m that stands for the light component that scattered at least once:

$$L(\vec{x},\vec{\omega}) = L_d(\vec{x},\vec{\omega}) + L_m(\vec{x},\vec{\omega}).$$

The direct term is reduced by out-scattering:

$$\frac{\mathrm{d}L_d}{\mathrm{d}s} = -\sigma_t(\vec{x})L_d(\vec{x},\vec{\omega}).$$

The media term is not only reduced by out-scattering, but also increased by in-scattering:

$$\frac{\mathrm{d}L_m}{\mathrm{d}s} = -\sigma_t L_m + \sigma_s \int_{\Omega'} (L_d + L_m)\, P(\vec{\omega}',\vec{\omega})\mathrm{d}\omega'.$$

Note that this equation can be re-written by considering the reflection of the direct term as a *volumetric source*:

$$\frac{\mathrm{d}L_m}{\mathrm{d}s} = -\sigma_t(\vec{x})L_m(\vec{x},\vec{\omega}) +$$

$$\sigma_s(\vec{x}) \int_{\Omega'} L_m(\vec{x},\vec{\omega})P(\vec{\omega}',\vec{\omega})\mathrm{d}\omega' + \sigma_s(\vec{x})Q(\vec{x},\vec{\omega}), \qquad (3)$$

where the source intensity is

$$Q(\vec{x},\vec{\omega}) = \int_{\Omega'} L_d(\vec{x},\vec{\omega}')P(\vec{\omega}',\vec{\omega})\mathrm{d}\omega'.$$

The volumetric source represents the coupling between the direct and media terms.

Although the direct term can be expressed as an integral even in inhomogeneous media, the evaluation of this integral requires ray marching and numerical quadrature. Having obtained the direct term and transformed it to the volumetric source, the media term needs to be computed.

Cerezo et al. [CPP*05] classified algorithms solving the transport equation as analytic, stochastic, and iterative.

Analytic techniques rely on simplifying assumptions, such as that the volume is homogeneous, and usually consider only the single scattering case [Bli82, SRNN05]. Jensen et al. [JMLH01] attacked the subsurface light transport by assuming that the space is partitioned into two half spaces with homogeneous material and developed the dipole model. Narasimhan and Nayar [NN03] proposed a multiple scattering model for optically thick homogeneous media and isotropic light source.

Stochastic methods apply Monte Carlo integration to solve the transport problem [KH84, JC98]. These methods are the most accurate but are far too slow in interactive applications.

Iterative techniques need to represent the current radiance estimate that is refined in each step [DMK00]. The radiance function is specified either by *finite-elements*, using, for example, the zonal method [RT87], spherical harmonics [KH84], radial basis functions [ZRL*08], metaballs, etc. or exploiting the particle system representation [SKSU05].

Stam [Sta95] introduced *diffusion theory* to compute energy transport. Here the angular dependence of the radiance is approximated by a two-term expansion:

$$L(\vec{x},\vec{\omega}) \approx \tilde{L}(\vec{x},\vec{\omega}) = \frac{1}{4\pi}\phi(\vec{x}) + \frac{3}{4\pi}\vec{E}(\vec{x})\cdot\vec{\omega}.$$

By enforcing the equality of the directional integrals of L and $\tilde{L}$, we get the following equation for *fluence* $\phi(\vec{x})$:

$$\phi(\vec{x}) = \int_{\Omega} L(\vec{x},\vec{\omega})\mathrm{d}\omega.$$

Requiring $\int_{\Omega} L(\vec{x},\vec{\omega})\vec{\omega}\, d\omega = \int_{\Omega} \tilde{L}(\vec{x},\vec{\omega})\vec{\omega}\, d\omega$, we obtain *vector irradiance* $\vec{E}(\vec{x})$ as

$$\vec{E}(\vec{x}) = \int_{\Omega} L(\vec{x},\vec{\omega})\vec{\omega}\mathrm{d}\omega.$$

Substituting this two-term expansion into the radiative transport equation and averaging it for all directions, we obtain the following diffusion equations:

$$\vec{\nabla}\phi(\vec{x}) = -3\sigma_t'\vec{E}(\vec{x}), \quad \vec{\nabla}\cdot\vec{E}(\vec{x}) = -\sigma_a\phi(\vec{x}). \qquad (4)$$

where $\sigma_t' = \sigma_t - \sigma_s g$ is the *reduced extinction coefficient*.

In [Sta95] the diffusion equation is solved by either a multi-grid scheme or a finite-element blob method. Geist et al. [GRWS04] computed multiple scattering as a diffusion process, using a lattice-Boltzmann solution method.

In order to speed up the solutions to interactive rates, the transport problem is often simplified and the solution is implemented on the GPU. The *translucent rendering* approach [KPHE02] involves multiple scattering simulation, but considers only multiple approximate forward scattering and single backward scattering. This method aims at nice images instead of physical accuracy. Physically based global illumination methods, like the photon map, have also been used to solve the multiple scattering problem [QXFN07]. To improve speed, light paths were sampled on a finite grid.

The high computational burden of multiple scattering simulation has been attacked by parallel methods both in surface [ACD08] and volume rendering [SMW*04]. Parallel volume rendering methods considered the visualization of very large datasets, while interactive multiple scattering simulation has not been in focus yet. Stochastic methods scale well on parallel systems, so they would be primary candidates for parallel machines, but their convergence rate is still too slow. Iterative techniques, on the other hand, converge faster but require data exchanges between the nodes, which makes scalability sub-linear.

This paper presents a fast parallel solution for the radiative transport equation (Figure 1). We have taken the iteration approach because of its better convergence. This posed challenges for the parallel implementation because we should attack the sub-linear scalability and the communication bottleneck. Our approach is based on two observations. Iteration

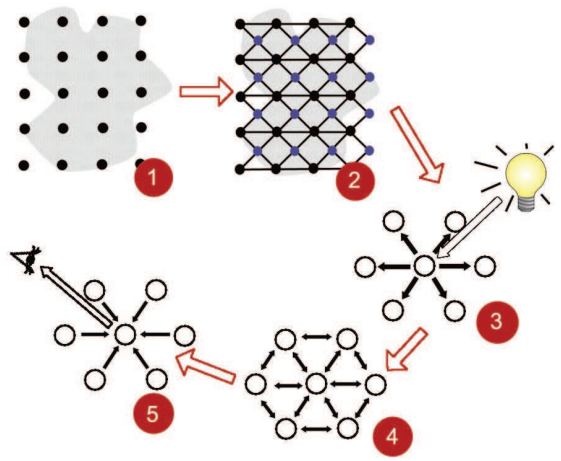


Figure 1: *The outline of the algorithm. 1: The volume is defined on a grid. 2: An illumination network is established between the grid points. 3: Single scattering and estimated multiple scattering are distributed from each light source. 4: The final results are obtained by iteration which corrects the errors of the estimation. 5: The image is rendered by standard alpha blending.*

is slow because it requires many "warming up" steps to distribute the power of sources to far regions. Thus, if we can find an easy way to approximate the solution, then iteration should only refine the initial approximation, which could be done in significantly fewer steps. On the other hand, iteration requires the exchange of data from all computing nodes in each step, which leads to a communication bottleneck. We propose an iteration scheme when the data are exchanged less frequently. This slows down the convergence of the iteration, so computing nodes should work longer, but reduces the communication load.

We shall assume that the primary source of illumination is a single point light source in the origin of our coordinate system. More complex light sources can be modeled by translation and superposition. We use a simple and fast technique to initially distribute the light in the medium. The distribution is governed by the diffusion theory, where the single pass approximate solution is made possible by assumptions that the medium is locally homogeneous and spherically symmetric. The solution is approximate but can be obtained at the same cost as the direct term. Having obtained the initial approximation, the final solution is computed by iteration on a GPU cluster.

This paper is organized as follows. Section 2 explains the iteration solution, the importance of having a good initial approximation, and the challenges of parallel execution. Section 3 discusses the computation of the direct term. Section 4 presents the initial estimation of the radiance. Section 5 deals with the iterative refinement. Section 6 presents our distributed implementation and Section 7 summarizes the results.

2. Iteration solution of the radiative transport equation

The transport equation is an integro-differential equation. Integrating both sides, the equation can be turned to an integral equation of the following form:

$$L = \mathcal{T}L + Q^e$$

where $\mathcal{T}$ is a linear integral operator and Q^e is the source term provided by the boundary conditions. Applying finite-element techniques, the continuous radiance function is represented by finite data, which turns the integrated transport equation to a system of linear equations:

$$\mathbf{L} = \mathbf{T} \cdot \mathbf{L} + \mathbf{Q}^e,$$

where vector $\mathbf{L}$ is the radiance of the sample locations and directions, $\mathbf{Q}^e$ is the vector of source terms and boundary conditions, and $\mathbf{T}$ is the transport matrix.

Iteration obtains the solution as the limiting value of the following iteration sequence

$$\mathbf{L}_n = \mathbf{T} \cdot \mathbf{L}_{n-1} + \mathbf{Q}^e$$

so if this scheme is convergent, then the solution can be obtained by starting with an arbitrary radiance distribution $\mathbf{L}_0$ and iteratively repeating operator $\mathbf{T}$. The convergence is guaranteed if $\mathbf{T}$ is a contraction, i.e. for the norm of this matrix, we have

$$\|\mathbf{T} \cdot \mathbf{L}\| < \lambda \|\mathbf{L}\|, \quad \lambda < 1,$$

which is the case if the albedo is less than 1.

The error at a particular step n can be found by subtracting the solution $\mathbf{L}$ from the iteration scheme, and applying the definition of the contraction iteratively:

$$\|\mathbf{L}_n - \mathbf{L}\| = \|\mathbf{T} \cdot (\mathbf{L}_{n-1} - \mathbf{L})\| < \lambda \|\mathbf{L}_{n-1} - \mathbf{L}\| < \lambda^n \|\mathbf{L}_0 - \mathbf{L}\|.$$

Note that the error is proportional to the norm of the difference of the initial guess $\mathbf{L}_0$ and the final solution $\mathbf{L}$. Thus, having a good initial guess that is not far from the solution, the error after n iteration steps can be significantly reduced.

2.1. Iteration on parallel machines

In order to execute the iteration on a parallel machine, the radiance vector $\mathbf{L}_n$ is broken to parts and each computing node is responsible for the update of its own part. However, the new value of a part also depends on other parts, which would necessitate state exchanges between the nodes in every iteration. This would quickly make the communication the bottleneck of the parallel computation.

This problem can be attacked by not exchanging the current state in every iteration cycle. Suppose, for example, that

we exchange data just in every second iteration cycle. When the data is exchanged before executing the matrix-vector multiplication, the iteration looks like the original formula:

$$\mathbf{L}_n = \mathbf{T} \cdot \mathbf{L}_{n-1} + \mathbf{Q}^e.$$

However, when the data is not exchanged, a part of the matrix is multiplied by the radiance estimate of the older iteration. Let us denote the matrix by $\mathbf{T}^*$ that is similar to $\mathbf{T}$ where the own part is multiplied and zero elsewhere. With this notation, the cycle without previous data exchange is:

$$\mathbf{L}_n = \mathbf{T}^* \cdot \mathbf{L}_{n-1} + (\mathbf{T} - \mathbf{T}^*) \cdot \mathbf{L}_{n-2} + \mathbf{Q}^e.$$

Putting the two equations together, the execution of an iteration without state changes and then an iteration with state changes would result in:

$$\mathbf{L}_n = \mathbf{T}^2 \cdot \mathbf{L}_{n-2} + \mathbf{T} \cdot \mathbf{Q}^e + \mathbf{Q}^e + \mathbf{T} \cdot (\mathbf{T} - \mathbf{T}^*) \cdot (\mathbf{L}_{n-3} - \mathbf{L}_{n-2}).$$

Note that if this scheme is convergent, then $\mathbf{L}_n$, $\mathbf{L}_{n-2}$, and $\mathbf{L}_{n-3}$ should converge to the same vector $\mathbf{L}$, thus the limiting value satisfies the following equation:

$$\mathbf{L} = \mathbf{T}^2 \cdot \mathbf{L} + \mathbf{T} \cdot \mathbf{Q}^e + \mathbf{Q}^e.$$

This equation is equivalent to the original equation, which can be proven if the right side's $\mathbf{L}$ is substituted by the complete right side:

$$\mathbf{L} = \mathbf{T} \cdot \mathbf{L} + \mathbf{Q}^e = \mathbf{T} \cdot (\mathbf{T} \cdot \mathbf{L} + \mathbf{Q}^e) + \mathbf{Q}^e.$$

The price of not exchanging the data in every iteration step is the additional error term $\mathbf{T} \cdot (\mathbf{T} - \mathbf{T}^*) \cdot (\mathbf{L}_{n-3} - \mathbf{L}_{n-2})$. This error term converges to zero, but slows down the iteration process especially at the beginning of the iteration.

Using the same argument, we can prove a similar statement for cases when the data is exchanged just in every third, fourth, etc. cycles. The number of iterations done by the nodes between data exchanges should be specified by finding an optimal compromise, which depends on the relative computation and communications speeds.

3. Distribution of the direct term

The direct term is reduced by out-scattering. As the source is in the origin, the direct term is non-zero only for the direction from the origin to the considered point. Let us consider a point at distance r on a beam started at the source and having solid angle $\Delta\Omega$, and step on this beam by $\mathrm{d}r$. As a photon collides with the medium with probability $\sigma_t(r)\mathrm{d}r$ during the step, the *radiant intensity* (i.e. the power per solid angle) $\Phi(r)$ at distance r satisfies the following equation

$$\Phi(r+\mathrm{d}r) = \Phi(r) - \sigma_t(r)\mathrm{d}r\Phi(r) \implies \frac{\mathrm{d}\Phi(r)}{\mathrm{d}r} = -\sigma_t(r)\Phi(r). \tag{5}$$

If the radiant intensity is Φ_0 at the source, then the solution of this equation is

$$\Phi(r) = \Phi_0 e^{-\int_0^r \sigma_t(s)\mathrm{d}s}.$$

The *radiance* is the power per differential solid angle and differential area. In our beam the power is the product of radiant intensity $\Phi(r)$ and solid angle $\Delta\Omega$. On the other hand, the solid angle in which the source is visible equals to zero, which introduces a Dirac delta in the radiance formula. The area at distance r grows as $\Delta A = \Delta\Omega r^2$. Thus, the radiance of the direct term is

$$L_d(\vec{x}, \vec{\omega}) = \frac{\Phi(r)\Delta\Omega}{\Delta\Omega r^2}\delta(\vec{\omega} - \vec{\omega}_{\vec{x}}) = \frac{\Phi(r)}{r^2}\delta(\vec{\omega} - \vec{\omega}_{\vec{x}}), \tag{6}$$

where $r = |\vec{x}|$ is the distance and $\vec{\omega}_{\vec{x}} = \vec{x}/|\vec{x}|$ is the direction of the point from the source.

4. Initial distribution of the estimated radiance

Let us consider just a single beam starting at the origin where the point source is. When a beam is processed, we shall assume that other beams face to the same material characteristics, i.e. we assume that the scene is *spherically symmetric*. Consequently, the solution should also have spherical symmetry.

In case of spherical symmetry, the radiance of the inspected beam at point $\vec{x}$ and in direction $\vec{\omega}$ may depend just on distance $r = |\vec{x}|$ from the origin and on angle θ between direction $\vec{\omega}$ and the direction of point $\vec{x}$. This allows parametrization $L(r, \theta)$ instead of $L(\vec{x}, \vec{\omega})$. The fluence depends just on distance r and vector irradiance $\vec{E}(\vec{x})$ has the direction of the given point, that is $\vec{E}(\vec{x}) = E(r)\vec{\omega}_{\vec{x}}$.

Expressing the divergence operator in spherical coordinates, we get:

$$\vec{\nabla} \cdot \vec{E}(\vec{x}) = \vec{\nabla} \cdot (E(r)\vec{\omega}_{\vec{x}}) = \frac{1}{r^2}\frac{\partial(r^2 E(r))}{\partial r}.$$

Thus, the scalar versions of the diffusion equations are:

$$\frac{\mathrm{d}\phi(r)}{\mathrm{d}r} = -3\sigma_t' E(r), \quad \frac{1}{r^2}\frac{\mathrm{d}(r^2 E(r))}{\mathrm{d}r} = -\sigma_a \phi(r). \tag{7}$$

For homogeneous infinite material, the differential equation can be solved analytically:

$$\phi_0^h(r) = \frac{3\sigma_t'\Phi_0}{r}e^{-\sigma_e r}, \quad E_1^h(r) = \frac{\Phi_0}{r^2}e^{-\sigma_e r}(\sigma_e r + 1). \tag{8}$$

where $\sigma_e = \sqrt{3\sigma_a\sigma_t'}$ is the *effective transport coefficient*.

One option for the initial radiance estimation would be the application of the homogeneous solution assuming that the volume everywhere has similar material properties obtained as the average of the real values. However, this approach would provide poor estimates in regions having very different scattering parameters, that is in strongly inhomogeneous materials. Thus, we use the homogeneous solution only in the neighborhood of the source and farther away differential equation 7 is iterated. The volume is processed by ray marching initiated at the source. The radiance of these rays forming a bundle is initialized with the homogeneous solution.

As ray marching proceeds taking steps Δ increasing distance r, material properties σ_t, σ_s, and g are fetched at the sample location, and state variables $\phi(r)$, and $E(r)$ are updated according to the numerical quadrature solving equation 7, resulting in the following iteration formulae:

$$\begin{aligned} \phi(r+\Delta) &= \phi(r) - 3\sigma_t' E(r)\Delta, \\ E(r+\Delta) &= \frac{r^2}{(r+\Delta)^2}\left(E(r) - \sigma_a \phi(r)\Delta\right). \end{aligned} \tag{9}$$

5. Refinement of the initial solution by iteration

At the end of the approximate radiance distribution we have good estimates for the direct term L_d and volumetric source

$$Q(\vec{x},\vec{\omega}) = \int_{\Omega'} L_d(\vec{x},\vec{\omega}')P(\vec{\omega}',\vec{\omega})\mathrm{d}\omega' = \frac{\Phi(r)}{r^2}P(\vec{\omega}_{\vec{x}},\vec{\omega}),$$

and probably less accurate estimates for the total radiance

$$L(\vec{x},\vec{\omega}) \approx \frac{1}{4\pi}\phi(\vec{x}) + \frac{3}{4\pi}\vec{E}(\vec{x})\cdot\vec{\omega}.$$

Thus, we can accept direct term L_d, but the media term $L_m = L - L_d$ needs further refinement. We use an iteration scheme to make the media term more accurate, which is based on equation 3, but considers only the voxel centers. The *incoming medium radiance* arriving at voxel p from direction $\vec{\omega}$ is denoted by $I_m^{(p)}(\vec{\omega})$. Similarly, the *outgoing medium radiance* is denoted by $L_m^{(p)}(\vec{\omega})$. Using these notations, the discretized version of equation 3 at voxel p is:

$$L_m^{(p)}(\vec{\omega}) = (1-\alpha^{(p)})I_m^{(p)}(\vec{\omega}) + \alpha^{(p)}a^{(p)}\int_{\Omega'} I_m^{(p)}(\vec{\omega}')P^{(p)}(\vec{\omega}',\vec{\omega})\mathrm{d}\omega' + \alpha^{(p)}a^{(p)}Q^{(p)}(\vec{\omega}) \tag{10}$$

since $\sigma_t\Delta \approx \alpha$ and $\sigma_s\Delta \approx \alpha a$.

The incoming radiance of a voxel is equal to the outgoing radiance of another voxel that is the neighbor in the given direction, or it is set explicitly by the boundary conditions. Since in the discretized model a voxel has just finite number of neighbors, the in-scattering integral can also be replaced by a finite sum:

$$\int_{\Omega'} I^{(p)}(\vec{\omega}')P^{(p)}(\vec{\omega}',\vec{\omega})\mathrm{d}\omega' \approx \frac{4\pi}{D}\sum_{d=1}^{D} I^{(p)}(\vec{\omega}_d')P^{(p)}(\vec{\omega}_d',\vec{\omega}).$$

where D is the number of neighbors, which are in directions $\vec{\omega}_1',\ldots,\vec{\omega}_D'$ with respect to the given voxel. The number of neighbors depends on the structure of the grid. In a conventional *Cartesian Cubic* grid, a grid point has 6 neighbors. In a so called *Body Centered Cubic grid* [Cse05] a voxel has 8 neighboring voxels that share a face, which still seems to be too small to approximate a directional integral. Thus, it is better to use a *Face Centered Cubic grid* (FCC grid) [QXFN07], where each voxel has $D = 12$ neighbors (Figure 2).

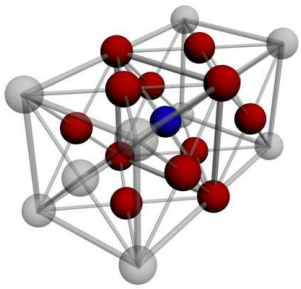


Figure 2: *Structure of the Face Centered Cubic Grid. Grid points are the voxel corners, voxel centers, and the centers of the voxel faces. Here every grid point has 12 neighbors, all at the same distance.*

Note that unknown radiance values appear both on the left and the right sides of the discretized transport equation. If we have an estimate of radiance values (and consequently, of incoming radiance values), then these values can be inserted into the formula of the right side and a new estimate of the radiance can be provided. Iteration keeps repeating this step. If the process is convergent, then in the limiting case the formula would not alter the radiance values, which are therefore the roots of the equation.

6. Implementation

The system has been implemented on a 5 node HP Scalable Visualization Array (SVA), where each node is equipped with an NVIDIA GeForce 8800 GTX GPU, programmed under CUDA. The nodes are interconnected by Infiniband.

In order to execute ray marching parallely for all rays during initial radiance distribution, the volume is resampled to a new grid that is parameterized with spherical coordinates. A voxel of the new grid with (u,v,w) coordinates represents fluence ϕ and vector irradiance E of point

$$\vec{x} = R(w\cos\psi\sin\theta, w\sin\psi\sin\theta, w\cos\theta),$$

$$\text{where} \quad \psi = 2\pi u, \quad \theta = \arccos(1-2v),$$

and R is the radius of the volume. Note that this parametrization provides uniform sampling in the directional domain. A (u,v) pair encodes a ray, while w encodes the distance from the origin. This texture is processed w-layer by w-layer, i.e. stepping the radius r simultaneously for all rays. In a single step the GPU updates the fluence and the vector irradiance according to equation 9.

The initial radiance distribution is not parallelized, but we trace all rays in all nodes. However, as we terminate rays leaving the subvolume associated with a node, it can also benefit from the addition of more nodes. Iteration, visual-

ization, and image compositing are executed in a distributed way.

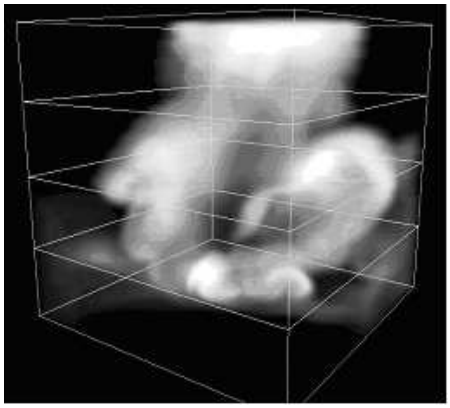


Figure 3: *Decomposition of the volume to subvolumes*

The tasks are distributed by subdividing the volume along one axis and each node is responsible for both the radiative transport simulation and the rendering of its associated subvolume (Figure 3). The images rendered by the nodes are composited by the ParaComp library [Par07].

During iterational refinement separate kernels are executed on the GPU for each computational step. The radiance distribution for one wavelength in the FCC grid is represented with 12 floating point arrays — one for each discrete direction in the grid. The FCC sites can be mapped into a standard 3D array by using proper indexing, where each value means the outgoing radiance from a given grid site in one direction. The volumetric source values remain constant during the iteration, so we store them in separate 3D textures. The iteration kernel updates the state of the grid by reading the emissions and the incoming radiances from the neighboring grid sites. The output of an iteration step is the input of the following one, so we copy the results back to the input textures after each kernel execution. In order to improve performance, we introduced a sensitivity constant which is a lower bound to the sum of the incoming radiances for each point. We evaluate the iteration formula only where the radiance value is greater than this constant. This method is efficient if there are larger parts of the volume without significant irradiance.

In addition to executing the iteration in the individual subvolumes, we need to implement the radiance transport between the neighboring volume parts. The simulation areas overlap so that the radiance values can be seamlessly passed from one subvolume to the other. MPI communication between the nodes is used to exchange the solutions at the boundary layers. It is important to notice that each node needs to pass only 4 arrays to its appropriate neighbor as the FCC grid has 4 outgoing and 4 incoming directions for each axis-aligned boundaries.

7. Results

First, we have examined the effect of the initial radiance approximation. Figure 4 shows the error curves of the iteration obtained when the radiance is initialized by the direct term only and when the media term approximation is also used. Note that the application of the media term approximation halved the number of iteration steps required to obtain a given accuracy.

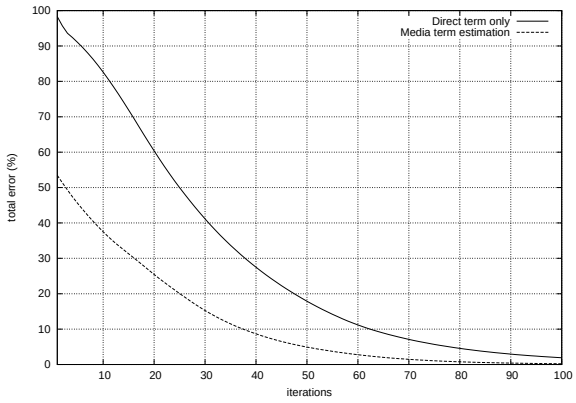


Figure 4: *Error curves of the iteration when the radiance is initialized to the single-scattering term and when the radiance is initialized to the media term approximation. Note that in the latter case, roughly only 50% of the iteration steps are needed to obtain the same accuracy.*

The evolution of the iteration can also be followed in Figures 5 and 6. Note that if we initialize the iteration with the direct term, we need about 100 iteration steps to eliminate any further visual change in the image (the error goes below 2%). However, when the radiance is initialized to the approximated media term, we obtain the same result executing only 60 iterations.

Finally, we tested the scalability of our parallel implementation. The volume is decomposed to 4 blocks along axis z (Figure 3), and the transfer of each block is computed on a separate node equipped with its own GPU. Table 1 summarizes the time data measured when a classical iteration scheme is executed that exchanges the boundary layers of the blocks in each iteration step, and when they are exchanged just after every fifth iteration step. We can observe that the visualization scales well with the introduction of new nodes but iteration time improves just moderately when boundary conditions are exchanged in each iteration step, because of the communication bottleneck. This bottleneck can be eliminated by exchanging the boundary conditions less frequently, which slightly reduces the speed of convergence, so we trade communication overhead for GPU computation power. We observed that the error caused by exchanging the boundary conditions just after every fifth iteration cycle can

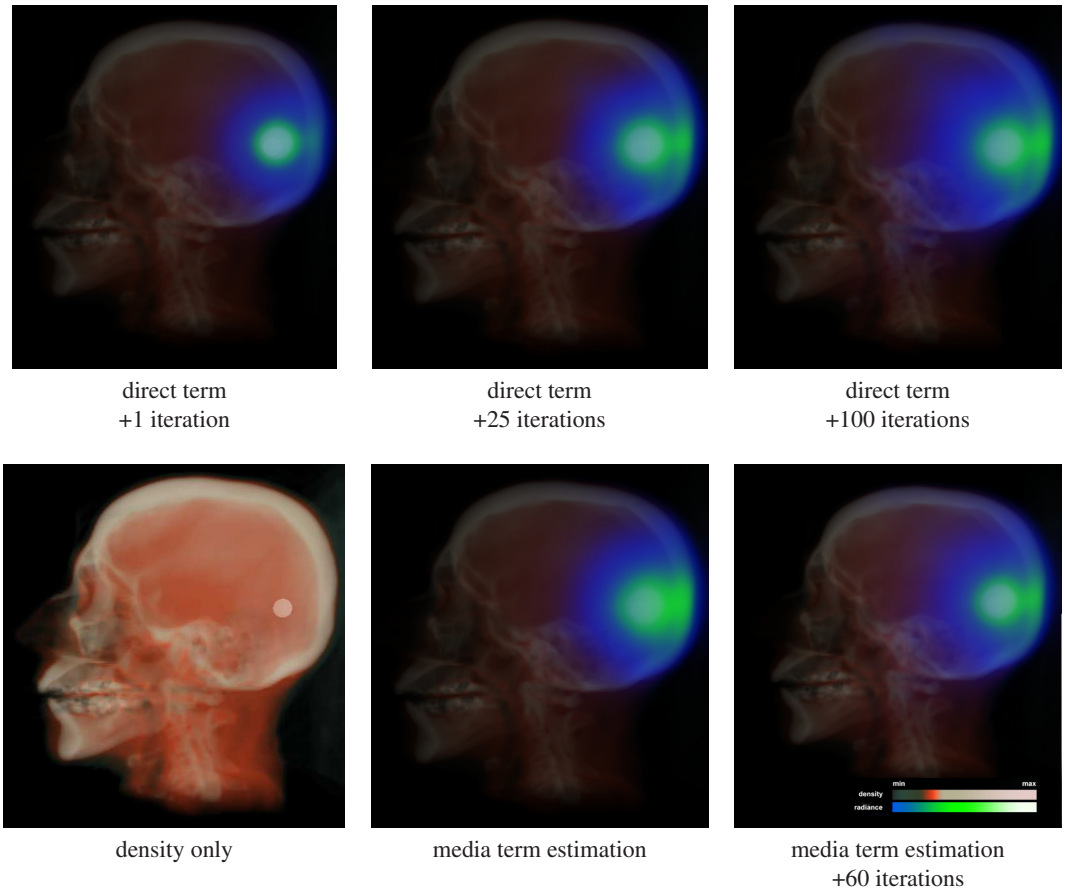


Figure 5: *Evolution of the iteration when the radiance is initialized to the direct term and to the estimated media term, respectively. The radiance is color coded to emphasize the differences and is superimposed on the density field.*

be compensated by about 5% more cycles, which is a good tradeoff. Note that when we exchanged the boundary conditions just after every fifth iteration cycle, the iteration speed scaled well with the introduction of newer nodes.

8. Conclusions

This paper proposed an effective method to solve the radiative transport equation in heterogeneous participating media on a cluster of GPUs. The transport equation is solved on an FCC grid by iteration. The iterative algorithm has been significantly improved by finding a good initial guess for the radiance and modifying the parallel implementation to reduce the frequency of data exchanges. Without these, the very high performance of GPUs would make the communication become the bottleneck. This concept of iterating more on the nodes without exchanges gives us a versatile tool to address the scalability issue. We have tested the approach on a cluster of GPUs, but it is equally applicable to multiple GPU cards inserted in the same desktop since they also share the problem of the communication bottleneck.

Acknowledgement

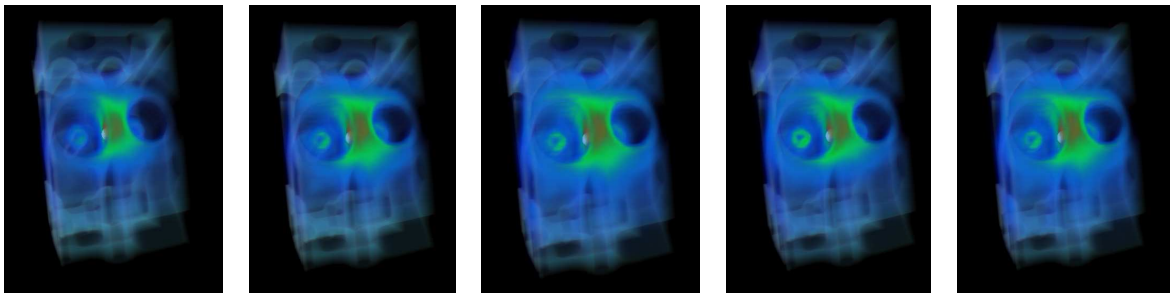
This work has been supported by the National Office for Research and Technology, Hewlett-Packard, OTKA K-719922 (Hungary), and by the Terratomo project.

References

[ACD08] AGGARWAL V., CHALMERS A., DEBATTISTA K.: High-Fidelity Rendering of Animations on the Grid: A Case Study. Favre J. M., Ma K.-L., (Eds.), Eurographics Association, pp. 41–48.

[Bli82] BLINN J. F.: Light reflection functions for simulation of clouds and dusty surfaces. In *SIGGRAPH '82 Proceedings* (1982), pp. 21–29.

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



direct term | direct term +1 iteration | direct term +25 iterations | media term approximation | converged result

Figure 6: *Evolution of the iteration when the radiance is initialized to the direct term and to the estimated media term, respectively. The radiance is color coded to emphasize the differences and is superimposed on the density field.*

No	Freq	Initial	Iteration	Visual	Total
2	1	27 ms	19 + 23 ms	35 ms	2.6 s
3	1	19 ms	12 + 25 ms	30 ms	2.2 s
4	1	17 ms	8 + 29 ms	25 ms	2.1 s
2	1/5	27 ms	19 + 10 ms	35 ms	1.8 s
3	1/5	19 ms	12 + 10 ms	30 ms	1.4 s
4	1/5	17 ms	8 + 11 ms	25 ms	1.1 s

Table 1: *Performance figures with respect to the number of nodes ("No") and the frequency of boundary conditions exchanges ("Freq"). The volume is a* $128 \times 128 \times 64$ *resolution grid. The resolution of the screen is* 600×600*. "Initial" time is needed by the initial radiance distribution, "Iteration" is the sum of the time of a single iteration cycle and the time required by the exchanges of the boundary conditions and texture ping-pong after the iteration cycle, "Visual" is needed by the final ray casting and compositing the partial images, and "Total" is the total simulation/rendering time needed to reduce the error below 2% (60 iterations if boundary conditions are exchanged after each iteration and 63 iterations if boundary conditions are exchanged after every 5 iterations).*

[Cse05] CSÉBFALVI B.: Prefiltered Gaussian reconstruction for high-quality rendering of volumetric data sampled on a body-centered cubic grid. In *VIS '05: Visualization, 2005* (2005), IEEE Computer Society, pp. 311–318.

[DMK00] DACHILLE F., MUELLER K., KAUFMAN A.: Volumetric global illumination and reconstruction via energy back-projection. In *Symposium on Volume Rendering* (2000).

[GRWS04] GEIST R., RASCHE K., WESTALL J., SCHALKOFF R.: Lattice-boltzmann lighting. In *Eurographics Symposium on Rendering* (2004).

[JC98] JENSEN H. W., CHRISTENSEN P. H.: Efficient simulation of light transport in scenes with participating media using photon maps. *SIGGRAPH '98 Proceedings* (1998), 311–320.

[JMLH01] JENSEN H. W., MARSCHNER S., LEVOY M., HANRAHAN P.: A practical model for subsurface light transport. *SIGGRAPH 2001 Proceedings* (2001).

[KH84] KAJIYA J., HERZEN B. V.: Ray tracing volume densities. In *SIGGRAPH '84 Proceedings* (1984), pp. 165–174.

[KPHE02] KNISS J., PREMOZE S., HANSEN C., EBERT D.: Interactive translucent volume rendering and procedural modeling. In *VIS '02: Proceedings of the conference on Visualization '02* (2002), IEEE Computer Society, pp. 109–116.

[NN03] NARASIMHAN S. G., NAYAR S. K.: Shedding light on the weather. In *CVPR 03* (2003), pp. 665–672.

[Par07] PARACOMP: *HP Scalable Visualization Array Version 2.1*. Tech. rep., HP, 2007. http://docs.hp.com/en/A-SVAPC-2C/A-SVAPC-2C.pdf.

[QXFN07] QIU F., XU F., FAN Z., NEOPHYTOS N.: Lattice-based volumetric global illumination. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1576–1583. Fellow-Arie Kaufman and Senior Member-Klaus Mueller.

[RT87] RUSHMEIER H. E., TORRANCE K. E.: The zonal method for calculating light intensities in the presence of a participating medium. In *SIGGRAPH 87* (1987), pp. 293–302.

[SKSU05] SZIRMAY-KALOS L., SBERT M., UMENHOFFER T.: Real-time multiple scattering in participating media with illumination networks. In *Eurographics Symposium on Rendering* (2005), pp. 277–282.

[SMW*04] STRENGERT M., MAGALLÓN M., WEISKOPF D., GUTHE S., ERTL T.: Hierarchical Visualization and Compression of Large Volume Datasets Using GPU Clusters . Bartz D., Raffin B., Shen H.-W., (Eds.), Eurographics Association, pp. 41–48.

[SRNN05] SUN B., RAMAMOORTHI R., NARASIMHAN S. G., NAYAR S. K.: A practical analytic single scattering model for real time rendering. *ACM Trans. Graph. 24*, 3 (2005), 1040–1049.

[Sta95] STAM J.: Multiple scattering as a diffusion process. In *Eurographics Rendering Workshop* (1995), pp. 41–50.

[ZRL*08] ZHOU K., REN Z., LIN S., BAO H., GUO B., SHUM H.-Y.: Real-time smoke rendering using compensated ray marching. *ACM Trans. Graph. 27*, 3 (2008), 36.