Heterogeneous Subsurface Scattering Using the Finite Element Method

Adam Arbree, Bruce Walter, and Kavita Bala, *Member*, *IEEE*

Abstract—Materials with visually important heterogeneous subsurface scattering, including marble, skin, leaves, and minerals are common in the real world. However, general, accurate, and efficient rendering of these materials is an open problem. In this paper, we describe a finite element (FE) solution of the heterogeneous diffusion equation (DE) that solves this problem. Our algorithm is the first to use the FE method to solve the difficult problem of heterogeneous subsurface rendering. To create our algorithm, we make two contributions. First, we correct previous work and derive an accurate and complete heterogeneous diffusion formulation with two key elements: the diffusive source boundary condition (DSBC)—an accurate model of the reduced intensity (RI) source—and its associated render query function. Second, we solve this formulation accurately and efficiently using the FE method. With these contributions, we can render subsurface scattering with a simple four step algorithm. To demonstrate that our algorithm is simultaneously general, accurate, and efficient, we test its performance on a series of difficult scenes. For a wide range of materials and geometry, it produces, in minutes, images that match path traced references, that required hours.

Index Terms—Three-dimensional graphics and realism, color, shading, shadowing, texture, miscellaneous, subsurface scattering, partial differential equations, finite element methods.

1 Introduction

The subsurface scattering of light creates the distinctive appearance of many ubiquitous materials, such as marble, skin, minerals, and leaves. However, because of the complexity of the scattering within these materials, efficient and accurate subsurface rendering is challenging, and previous approaches have limitations. Monte Carlo (MC) algorithms [21], [27], [39] accurately solve the general subsurface rendering problem but their hours-per-image cost makes them impractical for most applications; algorithms using the dipole diffusion bidirectional surface scattering reflectance distribution function (BSSRDF) [28] are fast, even real-time, but can only model homogeneously scattering materials, and, though capture and re-render systems [12], [17], [19], [38] can quickly generate high-quality images, they can only redisplay captured materials.

Because of these limitations, current research seeks an improved result: an efficient and accurate rendering algorithm for general heterogeneous materials. We achieve this result by using a finite element (FE) algorithm to solve the heterogeneous diffusion equation (DE). However, creating our solution required two contributions to previous work:

- a complete and correct rendering formulation for the heterogeneous diffusion problem; and
- an efficient FE algorithm that solves this problem.

Our contributions strike a balance between performance and accuracy. By solving an approximate scattering model, we ensure efficiency, but, by carefully selecting this approximation and by using the FE method, we preserve accuracy. As a result, in a few minutes, our algorithm produces images (see Fig. 1, top row) that nearly match path-traced references which required hours of computation (see Fig. 1 difference images, top right). Moreover, because our algorithm is a general solution, it can reproduce a wide range of difficult subsurface effects. For example, it can render complex high-frequency aggregates, such as marble (see Fig. 1, bottom left), the sharp edges between regions of smooth, noise-free translucency (see Fig. 1, bottom center), and, even in a challenging scene lit entirely by subsurface illumination, the complex, structured, and high-resolution subsurface detail found in many natural materials (see Fig. 1, bottom right).

Our contributions address two gaps found in previous discussions of heterogeneous subsurface rendering. First, we correctly derive our heterogeneous diffusion formulation with an accurate model of the reduced intensity (RI) source, the diffusive source boundary condition (DSBC), and its corresponding render query function. Because we derive the DSBC and the query function from first principles, we are able to correct the errors found in previous discussions [40], [49] and produce a heterogeneous diffusion formulation suitable for high quality rendering applications. Second, though FE algorithms have been used for a wide range of mathematical problems, our algorithm improves all previous subsurface renderers by introducing the FE method for rendering subsurface scattering.

In particular, our FE algorithm significantly extends the two recent works most closely related to our own. First, we show that our FE solution improves the ambitious and heterogeneous capture, modeling and rendering system recently created by Wang et al. [49]. Though their algorithm

- *A. Arbree is with Autodesk® Corporation, 1 Market St, San Franscisco, CA 94903. E-mail: adam.arbree@autodesk.com.*
- *B. Walter and K. Bala are with Cornell University, Ithaca, NY 14853. E-mail: bjw@graphics.cornell.edu, kb@cs.cornell.edu.*

Manuscript received 13 Jan. 2010; revised 21 Apr. 2010; accepted 30 Apr. 2010; published online 8 Sept. 2010.
Recommended for acceptance by B. Guo.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2010-01-0006.
Digital Object Identifier no. 10.1109/TVCG.2010.117.

Fig. 1. Our FE algorithm for heterogeneous subsurface scattering can reproduce a wide range of materials in arbitrary geometry. Our results (see Dragon results, top row) are nearly identical to path traced references (see black difference images, top right) but render in minutes instead of hours. Our algorithm can (see marble Bunny, left) model complex aggregate materials (see checkerboard Buddha; middle right), capture edges in heterogeneous material properties while still ensuring noise-free translucency in smooth regions, and (see back-lit Geode, right) scale to detailed, high-frequency materials even in the difficult case, when all illumination passes through the medium.

performs excellently for their re-render application, we demonstrate that the quality of their results depends on their capture optimization correcting rendering errors. For noncaptured materials, their finite difference (FD) rendering algorithm is less accurate, and its dependence on PolyGrid [45] meshes places effective limits on the scattering geometry. In contrast, our new algorithm produces high quality images of many materials. Moreover, because our FE solution supports fast, off-the-shelf meshing algorithms and intelligent adaptive mesh refinement, we can achieve these results quickly in a wide range of scattering geometries.

Second, we extend similar, but much more limited, FE results discussed in previous work in medical imaging [40]. Though this work also discusses an FE solution to the diffusion equation, it omits the details necessary to create a useful heterogeneous rendering algorithm. The authors only consider a simplified FE solution in a 2D homogeneous disk and, because of this simple test and the different requirements of their imaging application, they conclude that their formulation cannot be used for a wide range of materials and geometry. However, we find the opposite result for the rendering problem. By using an improved 3D FE solution tailored for rendering, we can achieve high-quality results in a wide range of difficult scenes.

The rest of this paper describes and evaluates our heterogeneous subsurface rendering algorithm. First, we review previous work on subsurface rendering and then, in Section 3, we describe our contributions and algorithm. Afterward, Section 4 reviews diffusion theory to prepare for Sections 5, 6, and 7 that build on this theory to derive our algorithm. Then, in Section 8, we review the details of our implementation, and in Section 9, we compare this implementation to previous solutions. Finally, in Section 10, we conclude with limitations and future work.

2 Previous Work

Due to the breadth of research on subsurface rendering, a complete review lies beyond the scope of this paper. This section briefly reviews most of the previous works in three broad categories: Monte Carlo (MC) algorithms, dipole diffusion algorithms, and capture and re-render systems. Afterward, the two algorithms most closely related to our problem, general heterogeneous subsurface rendering, are reviewed in more detail.

2.1 Monte Carlo Algorithms

By directly simulating basic scattering physics, MC rendering algorithms, including path tracing [21], [27], [39], photon mapping [13], [26], and Metropolis sampling [37], can generate individual photons paths in the scattering material. With sufficiently many paths, they can produce images of arbitrary accuracy; however, even for homogeneous materials, this simulation is generally considered impractical, and heterogeneous materials are yet more expensive (for example, our path-traced references required many CPU/days of computation, see Section 9.4).

2.2 Dipole Diffusion Algorithms

Because of the high cost of MC renderers, practical subsurface algorithms are approximate. Jensen et al. [28] introduced the first such approximation: the dipole diffusion BSSRDF. The dipole diffusion BSSRDF is derived from an analytic scattering approximation [24], [30] that, though fast, limits both the BSSRDF's quality [33] and fundamentally restricts it to homogeneous materials (though an artificial heterogeneous appearance can be created with textures [32], [46]). However, because it is easy to implement and inexpensive, the dipole diffusion BSSRDF has been widely used. It was first used to accelerate MC algorithms [25], [28] and then several papers extended these algorithms to improve their quality [7], [10], [11], [33]. Recently, the dipole diffusion BSSRDF has been used to create real-time algorithms [5], [6], [8], [22], [23], [32], [34], [35], included in precomputed radiance transfer systems [42], [50], and scaled to large scenes, using a Lightcuts based algorithm [2], [48].

2.3 Capture and Re-Render Systems

Capture and re-render systems—both for general subsurface scattering materials [19], [38], [43] and specifically for human skin [12], [17]—compute specialized material models from photographs of a physical scattering object. After this capture computation, the internal model can be warped into new geometry and re-rendered. Because these models are computed from real photographs, the re-rendered images have high quality. However, though these systems can capture both homogeneous and heterogeneous materials, these systems can only redisplay their captured models. They cannot render general heterogeneous materials.

2.4 Heterogeneous Algorithms

To address the limitations of the categories above, we present our general heterogeneous rendering algorithm. This algorithm computes subsurface scattering by numerically solving the DE. Most previous numerical solvers have limitations. Stam [44] first proposed a numerical algorithm, but did not present a general solution; Haber et al. [20] developed numerical solver, but only for homogeneous materials; and, recently, Fattal [16] developed a numerical discrete ordinates method, but for less optically thick materials. Only two works have discussed the general heterogeneous subsurface scattering problem: the finite difference (FD) solver introduced by Wang et al. [49] and the work by Schweiger et al. [40] in the medical imaging field of optical tomography.

2.4.1 Finite Difference Algorithms

As part of an impressive, capture and re-rendering system, Wang et al. [49] developed an interactive, FD algorithm for solving the heterogeneous DE. Their goal was the interactive editing of captured material grids and, for this task, their algorithm is excellent. However, as a general, high quality solution, we have found that their FD algorithm has several limitations (see Section 9.5 for a detailed discussion and a comparison to our results). Foremost, DE formulation errors can result in incorrect solutions. These errors were unnoticed in the original work because the capture optimization corrects them by altering the material parameters.

However, even if we correct their algorithm, using our formulation, two additional limitations remain. First, the FD algorithm requires that a PolyGrid [45] be constructed in the scattering geometry. As noted by the authors, this is a difficult process that can require hand optimization to avoid errors. Second, their FD solution—especially with the additional approximations required to achieve interactive performance—has approximation errors. Most importantly, the deformation of the FD algorithm's PolyGrid cannot be exactly modeled in their solver. Therefore, large deformations result in rendering errors, or, for some materials, the divergence of the iterative solution.

Contemporaneously with the development of this paper, a follow-up work [51] replaced the PolyGrid with a new QuadGraph structure based on a tetrahedral mesh. The QuadGraph improves construction costs and allows the FD mesh to be constructed without user intervention. Additionally, [51] uses our corrected heterogeneous scattering formulation (see our technical report, [3]) to improve quality. However, the QuadGraph solution, like [49], relies on an FD solver for interactive performance, and cannot achieve the high frequency resolution and accuracy of our FE algorithm.

2.4.2 Optical Tomography

Outside of computer graphics, the medical imaging field of optical tomography also develops algorithms for numerically solving the DE. On one hand, optical tomography tackles the same problem addressed by Wang et al. [49]: given a series of photographs of human tissue lit by visible light, determine the scattering properties of the tissue (see [18], [31] for a review of the state of the art). On the other hand, the optical tomography problem is significantly different than the rendering problem. Optical tomography applications are less concerned with efficiency, need solutions in much smaller regions than a whole image and consider only materials within the smaller range of scattering parameters found in human tissue.

However, one work [40] discusses, among several others, a DE problem formulation similar to ours and they consider solving this formulation using the FE method. But, though their discussion introduces topics similar to ours, it also omits the details necessary to create a useful rendering algorithm. They do not explore heterogeneous scattering in arbitrary objects and compute only a simplified finite element solution in a 2D homogeneous disk. Because of this simple test, they cannot determine whether their formulation is accurate or efficient for a wide range of materials and geometry. Finally, because they address a different problem, their equations lack two elements essential in our rendering solution: a boundary Fresnel term and a rendering query function that converts the final solution into radiance.

3 Algorithm

Our algorithm improves upon all previous subsurface renderers. Our results are nearly as accurate as Monte Carlo (MC) images, but much less expensive and, unlike both dipole diffusion algorithms and capture and re-render systems, our algorithm can quickly reproduce a broad range of heterogeneous materials. We achieve these results, using an FE algorithm to solve the heterogeneous DE.

Though the FE method has been used in a wide range of applications spanning many fields, our method is the first to use it to render subsurface scattering. However, because of the specific requirements of the rendering problem, our solution must be carefully formulated to ensure accurate results. Unfortunately, when developing our solution, we found errors in previous diffusion formulations [40], [49] that we correct in the discussion below (see also [3]). In addition, we further extend the 2D FE solution discussed by Schweiger et al. [40], describing a complete 3D heterogeneous FE solution, tailored to the rendering problem. The product of our careful and tailored derivations is a fast, accurate, and general FE subsurface rendering algorithm.

3.1 Overview of Key Results

Our final algorithm is defined by three key results: the diffusive source boundary condition (DSBC), the FE diffusion equation (FEDE), and the render query function.

3.1.1 Result #1: Diffusive Source Boundary Condition

Our algorithm approximates subsurface scattering, using the diffusion equation. This approximation has three components: the DE itself that approximates scattering in the interior of the scattering volume, a boundary condition (BC) that approximates the solution's boundary behavior, and a reduced intensity (RI) source model that approximates the radiance entering the scattering media. Most previous works present these three components in a unified formulation, based on the approximations that derive the dipole diffusion algorithm [28]. However, for general heterogeneous problems, we demonstrate that the reduced intensity source model used in this formulation, the embedded source model [11], is both inaccurate and inefficient. To address these limitations, Section 5 derives our first key result: the diffusive source boundary condition (see (13)). The DSBC combines an accurate boundary condition approximation with an improved RI source model to produce a high-quality heterogeneous subsurface scattering approximation.

3.1.2 Result #2: Finite Element Diffusion Equation

Next, in Section 6, we solve this approximation accurately, using the FE method. This solution has two steps. First, the scattering domain is discretized into a mesh that, by construction, implicitly defines a finite basis for functions in the scattering domain. Second, we discretize the constraints of the diffusion equation and diffusive source boundary condition, using this basis. The result is a large matrix equation whose solution is the coefficient vector of the best approximation, in this basis, of the final subsurface scattering solution. This matrix equation is our second key result—the FEDE (see (18)).

3.1.3 Result #3: Render Query Function

Finally, we describe how to compute exitant subsurface radiance, the quantity needed for rendering, from this solution. Because of the approximations of the DE, DSBC, and their FE solution, this query function must be carefully constructed to ensure accuracy. Section 7 describes these issues and derives our final key result—the render query function (see (21)).

3.2 Finite Element Rendering Algorithm

Using these results, rendering has four steps:

- Step 1. Mesh the scattering volume.
- Step 2. Construct the FEDE for that mesh.
- Step 3. Solve the resulting linear system.
- Step 4. Compute radiance, using the query function.

Moreover, because of the generality of our FE solution, this basic algorithm is not only simple, but adaptable. Though in Section 8, we describe the specific details of our full implementation, without changing its basic structure, it can be altered to fit the problem at hand, or to incorporate existing tools. Many mesh types can be used—allowing fast meshing algorithms (see Section 8.1) and adaptive refinement (see Section 8.5)—without changing the assembly of the FEDE in Step 2 (see Section 8.2 and Fig. 6 for the pseudocode of a general algorithm) and, once the FEDE is assembled, it works with any algorithm.

4 Review of Diffusion Theory

Before we derive our approximation algorithm, we review the exact solution, the volumetric radiance transfer equation, and the three elements of the standard diffusion approximation [24], the DE, the reduced intensity (RI) source model, and the boundary condition (BC), that our algorithm address.

4.1 Volumetric Radiative Transfer Equation

The light scattered within a randomly scattering medium is defined by three functions that specify the probability of photon interactions (see Fig. 2 for the notation and abbreviations). The absorption and scattering coefficients, $\sigma_a(\mathbf{x})$ and $\sigma_s(\mathbf{x})$, respectively, give the the number of photons scattered and absorbed per unit distance, and the phase function $p(\vec{\omega}, \vec{\omega}')$ specifies the probability that a scattered photon leaves $\vec{\omega}$ into $\vec{\omega}'$. Given these three functions, the volumetric radiance transfer equation defines the differential radiance $L(\mathbf{x}, \vec{\omega})$ in the medium leaving $\mathbf{x}$ in direction $\vec{\omega}$ [24].

4.1.1 Volumetric Radiative Transfer Equation

$$(\vec{\omega} \cdot \vec{\nabla}) L(\mathbf{x}, \vec{\omega}) = \sigma_s(\mathbf{x}) \int_{4\pi} p(\vec{\omega}, \vec{\omega}') L(\mathbf{x}, \vec{\omega}') \, d\vec{\omega} - \sigma_t(\mathbf{x}) L(\mathbf{x}, \vec{\omega}) + Q(\mathbf{x}, \vec{\omega}), \tag{1}$$

where $\sigma_t(\mathbf{x}) = \sigma_a(\mathbf{x}) + \sigma_s(\mathbf{x})$ and $Q(\mathbf{x}, \vec{\omega})$ is the light source function.

4.2 The Diffusion Equation

To derive the diffusion equation, one makes the diffusion approximation. The diffusion approximation assumes that the volumetric radiance transfer equation's solution has a smooth angular distribution, and represents it with a linear expansion in the 0^{th} and 1^{st} angular moments of the radiance, the fluence $\phi(\mathbf{x})$, and the vector irradiance $\vec{E}(\mathbf{x})$, respectively.

4.2.1 Diffusion Approximation

$$L(\mathbf{x}, \vec{\omega}) = \frac{1}{4\pi} \phi(\mathbf{x}) + \frac{3}{4\pi} \vec{E}(\mathbf{x}) \cdot \vec{\omega}, \tag{2}$$

Ω	Scattering volume	$\sigma_a(\mathbf{x})$	Absorption coefficient	$\Gamma_s(\mathbf{x})$	Incoming external flux
$\partial\Omega$	Boundary of Ω	$\sigma_s(\mathbf{x})$	Scattering coefficient	$\Gamma_d^{\text{in}}(\mathbf{x})$	Inward solution flux
$L(\mathbf{x},\vec{\omega})$	Radiance leaving $\mathbf{x}$ in $\vec{\omega}$	$\sigma_t(\mathbf{x})$	Total extinction coefficient	$\Gamma_d^{\text{ref}}(\mathbf{x})$	Boundary reflected flux
$L_d(\mathbf{x},\vec{\omega})$	Diffusive radiance	$\sigma_{tr}(\mathbf{x})$	Reduced extinction coefficient	$\mathbb{H}$	Arbitrary function space
$\phi(\mathbf{x})$	Radiant fluence	$\kappa_d(\mathbf{x})$	Diffusion coefficient	θ	Arbitrary function in $\mathbb{H}$
$\vec{E}(\mathbf{x})$	Vector irradiance	η	Relative index of refraction	β_i	i^{th} basis function
$Q(\mathbf{x},\vec{\omega})$	Emittance of the media	$F_r(\eta,\vec{\omega})$	Fresnel reflectance	F	Finite element matrix
$Q_{ri}(\mathbf{x})$	Reduced intensity source	$F_t(\eta,\vec{\omega})$	Fresnel transmittance	$\mathsf{D},\mathsf{M},\mathsf{S}$	Components of F
$Q_0(\mathbf{x})$	0^{th} moment of $Q(\mathbf{x},\vec{\omega})$	$F_{dr}(\eta)$	Average Fresnel reflectance	$\vec{\mathsf{r}}$	Finite element right hand side
$p(\vec{\omega},\vec{\omega}')$	Scattering phase function	$F_{dt}(\eta)$	Average Fresnel transmittance	$\vec{\mathsf{q}},\vec{\mathsf{g}}$	Components of $\vec{\mathsf{r}}$
μ	Mean cosine of $p(\vec{\omega},\vec{\omega}')$	$A(\eta)$	Fresnel boundary term	$\vec{\mathsf{a}}$	Finite element solution vector
MC	Monte Carlo	DE	Diffusion Equation	BC	Boundary Condition
FE	Finite Element	FEDE	Finite Element Diffusion Eqn.	DSBC	Diffusive Source Bdry. Cond.
FD	Finite Difference	RI	Reduced Intensity	PDE	Partial Differential Eqn.

Fig. 2. Summary of notation and abbreviations used throughout the paper.

where

$$\phi(\mathbf{x}) = \int_{4\pi} L(\mathbf{x},\vec{\omega})\, d\vec{\omega} \quad \text{and} \quad \vec{E}(\mathbf{x}) = \int_{4\pi} L(\mathbf{x},\vec{\omega})\cdot\vec{\omega}\, d\vec{\omega}.$$

Though it requires considerable manipulation (see Ishimaru [24] for the details), essentially the substitution of (2) into (1) yields the:

4.2.2 Diffusion Equation

$$-\vec{\nabla}\cdot(\kappa_d(\mathbf{x})\vec{\nabla}\phi(\mathbf{x})) + \sigma_a(\mathbf{x})\phi(\mathbf{x}) = Q_0(\mathbf{x}) + Q_{ri}(\mathbf{x}). \quad (3)$$

This substitution introduces three new quantities. Two, $\kappa_d(\mathbf{x})$, and $Q_0(\mathbf{x})$, simply redefine the input parameters:

$$\kappa_d = \frac{1}{3[(1-\mu)\sigma_s + \sigma_a]} \quad \text{and} \quad Q_0(\mathbf{x}) = \int_{4\pi} Q(\mathbf{x},\vec{\omega})\, d\vec{\omega}.$$

However, the third term, the RI source $Q_{ri}(\mathbf{x})$, is a new secondary approximation required by the diffusion approximation.

4.3 Reduced Intensity Source

The RI source is necessary because sufficiently near the boundary incoming radiance violates the diffusion approximation.[1] Far from the boundary, the diffusion approximation's required isotropy is naturally ensured, because any radiance, deep in the material, will have randomly scattered many times smoothing away any initial angular properties. However, sufficiently near the boundary, some radiance will not yet have scattered, and this radiance can be highly anisotropic. Since this anisotropy can cause approximation errors, this yet-to-be-scattered radiance is removed from the diffusion problem, and the diffusion approximation is applied only to the remaining diffusive radiance $L_d(\mathbf{x},\vec{\omega})$. However, this radiance cannot be neglected entirely, and to account for its removal, the DE's derivation introduces a new artificial source function, the RI source $Q_{ri}(\mathbf{x})$, that supplies any diffusive radiance that would have been generated before removal [24]. Unfortunately, an exact photon-by-photon RI source computation would erase the efficiency originally gained by using the DE to model bulk scattering and, as a practical matter, the RI source model must be approximated.

4.4 Boundary Condition

Besides the RI source model, the DE requires an approximate BC. With the introduction of the RI source, the DE was restricted to the diffusive radiance. Since this radiance must have scattered at least once, the physical BC requires that $L_d(\mathbf{x},\vec{\omega})$ be zero along all incoming directions

$$\forall(\vec{n}\cdot\vec{\omega}) < 0, \quad L_d(\mathbf{x},\vec{\omega}) = 0. \quad (4)$$

However, this condition is discontinuous, and, since this discontinuity violates the diffusion approximation, a diffusion approximation-compatible approximate condition must be used instead. Previous work has already described an accurate BC approximation derived from boundary Fresnel effects [15], [24]. We discuss this approximation in Section 5.4 as part of the derivation of the diffusive source boundary condition (DSBC).

5 Diffusive Source Boundary Condition

The boundary condition (BC) and reduced intensity (RI) source model approximations strongly affect the diffusion solution on the boundary of the scattering domain. Because only the boundary is visible in an image, these approximations effectively determine the quality of the final rendering algorithm. In our heterogeneous application, where fine boundary material detail must be accurately modeled, these approximations must be as accurate as possible.

In this section, we demonstrate that the RI source model used by previous dipole algorithms, the embedded source model, is both inefficient and inaccurate. To address these issues, this section discusses our first key result, the diffusive source boundary condition (DSBC), that replaces the embedded approximation.

5.1 Embedded Source Model

The embedded source model (Fig. 3a) is derived from an analytic RI source model for a collimated beam incident on

1. A similar problem also occurs near sources embedded in the medium. However, since these sources rarely occur in physical scenes and, because many light source geometries have exact analytic models [15], [24], we review only the boundary case here.

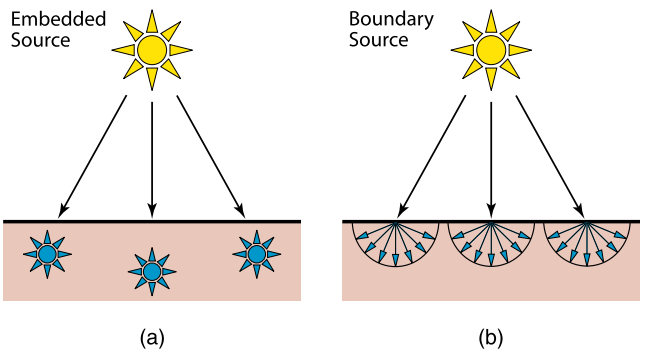


Fig. 3. Two methods of approximating the reduced intensity source $Q_{ri}(\mathbf{x})$. The embedded source model (a) approximates $Q_{ri}(\mathbf{x})$ with point sources while our model, the boundary source model (b), approximates the source as a diffusive flux arriving at the boundary.

a homogeneous, semi-infinite slab. For this problem, the RI source can be approximated by a series of omnidirectional point sources embedded inside the the medium. For arbitrary problems, the surface is divided into patches. By locally approximating each patch by the homogeneous slab problem, the collection of patches can be converted into a set of point sources that approximate the RI source throughout the scattering volume [11].

5.2 Boundary Source Model

Unlike the embedded source model, the DSBC (Fig. 3b) does not create a volumetric representation of the RI source. Instead, the DSBC is a 2D surface source model. Because of scattering and absorption, the RI source decays exponentially and, due to this rapid falloff, almost all of the power of the RI source lies close the boundary. Therefore, it is a small approximation to represent the true RI source by its 2D projection onto the boundary. This projection is modeled by a diffusive flux arriving at the boundary.

5.3 Limitations of the Embedded Source Model

In heterogeneous problems, the embedded source model has two drawbacks. Foremost, the embedded source approximation is less accurate. To demonstrate this result, we compare both models for the canonical example of an infinitesimal, collimated beam normally incident on a semi-infinite, homogeneously scattering slab with an albedo of one. In Fig. 4, we plot, outward from the point of incidence of the beam, a reference embedded solution, our FE solution using the DSBC[2] and an exact Monte Carlo (MC) result. Both near and far from the incident source, both solutions have errors. These errors are largely due to the inaccuracy of the diffusion approximation, shared by both results, in these regions. However, in the critical middle region (between 5 and 25 mean free paths) where the diffusion approximation is most accurate, both source methods closely approximate the true result. But, because it is derived from this configuration, the embedded source model should be maximally accurate for this example. Since in other more complex scenes the accuracy of the embedded source model will only degrade, we can conclude that the DSBC will be generally

2. The semi-infinite slab is replaced with a cube 100 mean free paths along each edge and the infinitesimally thin beam is replaced by a spot 0.2 mean free paths in diameter.

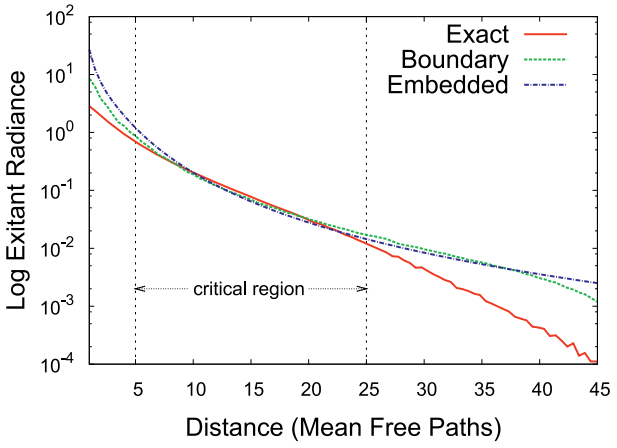


Fig. 4. Comparison of the boundary and embedded source models for a collimated beam incident on a semi-infinite slab of homogeneous materials: exact solution (solid red), diffusive source boundary condition (dotted green), and embedded source model (dashed blue).

more accurate. Though improved accuracy is already a compelling argument for the DSBC, the embedded source model has a second drawback: increased cost. When computing any subsurface solution, the source model must be evaluated at many points on the surface of an object. Each evaluation requires a lighting simulation to determine the power of the incident source. Because the embedded source model is volumetric, this simulation must be volumetric and is, therefore, more expensive. In contrast—because its evaluation is independent of any volume scattering—the DSBC can be accurately evaluated, using a lower frequency set of surface illumination samples (see Section 8.3 for our method).

5.4 Derivation of DSBC

Given these advantages, the DSBC is the clear choice for our heterogeneous diffusion problem. Deriving the DSBC, illustrated in Fig. 5, has two steps. First, we augment the Fresnel boundary condition (Figs. 5a and 5b) discussed in previous work [24] with a new term to model diffusive boundary flux, representing the RI source (Fig. 5c), and then we convert the resulting equation to a condition on the solution fluence by applying the diffusion approximation.

5.4.1 *Fresnel Boundary Condition*

Regardless of the RI source model, the diffusion BC should model Fresnel interface effects. At every boundary point $\mathbf{x}$, the basic condition forces the net inward flux of the solution

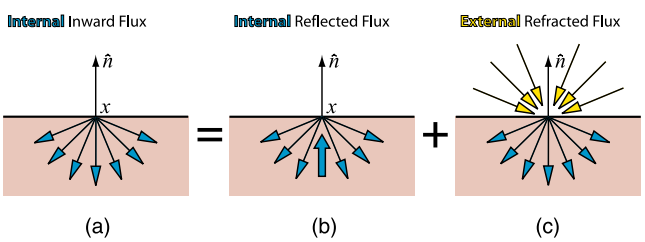


Fig. 5. Diagrams illustrating the three components of the diffusive source boundary condition (DSBC) (13). The condition forces $\Gamma_d^{\text{in}}(\mathbf{x})$, the internal inward flux at the boundary, to be equal to the sum of $\Gamma_d^{\text{ref}}(\mathbf{x})$, the internal flux reflected at the boundary, and $\Gamma_s(\mathbf{x})$ the exterior light refracted into the material. (a) $\Gamma_d^{\text{in}}(\mathbf{x})$. (b) $\Gamma_d^{\text{ref}}(\mathbf{x})$. (c) $\Gamma_s(\mathbf{x})$.

$\Gamma_d^{\text{in}}(\mathbf{x})$ to equal the total flux of internal radiance reflected back inward by the Fresnel interface $\Gamma_d^{\text{ref}}(\mathbf{x})$

$$\Gamma_d^{\text{in}}(\mathbf{x}) = \Gamma_d^{\text{ref}}(\mathbf{x}). \tag{5}$$

Using Figs. 5a and 5b as guides, these fluxes can be computed by integrating over the small blue arrows in each figure[3]

$$\Gamma_d^{\text{in}}(\mathbf{x}) = \int_{(\vec{n}\cdot\vec{\omega})<0} L_d(\mathbf{x},\vec{\omega})(-\vec{n}\cdot\vec{\omega})\, d\vec{\omega}, \tag{6}$$

$$\Gamma_d^{\text{ref}}(\mathbf{x}) = F_{dr}(\eta)\int_{(\vec{n}\cdot\vec{\omega})<0} L_d(\mathbf{x},-\vec{\omega})(-\vec{n}\cdot\vec{\omega})\, d\vec{\omega}. \tag{7}$$

In (7), like previous work [28], we approximate total reflected internal flux by scaling the total internal flux by $F_{dr}(\eta)$, the average Fresnel reflection coefficient.

5.4.2 Reduced Intensity Flux

Next, the RI source is added by augmenting the basic condition with a third term

$$\Gamma_d^{\text{in}}(\mathbf{x}) = \Gamma_d^{\text{ref}}(\mathbf{x}) + \Gamma_s(\mathbf{x}). \tag{8}$$

The new term $\Gamma_s(\mathbf{x})$ approximates the RI source from the integrated surface radiance refracted into the material at $\mathbf{x}$

$$\Gamma_s(\mathbf{x}) = e^{-\frac{\sigma_a(\mathbf{x})}{\sigma_s(\mathbf{x})}}\int_{(\vec{n}\cdot\vec{\omega})>0} F_t(\eta,\vec{\omega})L(\mathbf{x},-\vec{\omega})(\vec{n}\cdot\vec{\omega})\, d\vec{\omega}. \tag{9}$$

In (9), the exponential term approximates the absorption that has occurred as refracted radiance traveled in the medium before scattering and becoming part of the RI source.

5.4.3 Substitution of the Diffusion Approximation

Next, using an identity from previous work [24],

$$\int_{(\vec{s}\cdot\vec{\omega})>0} L_d(\mathbf{x},\vec{\omega})(\vec{s}\cdot\vec{\omega})\, d\vec{\omega} = \frac{1}{4}[\phi(\mathbf{x}) - 2\kappa_d(\mathbf{x})(\vec{s}\cdot\vec{\nabla})\phi(\mathbf{x})], \tag{10}$$

we can express (6) and (7) in terms of fluence by substituting the diffusion approximation (2) and simplifying we get

$$\Gamma_d^{\text{in}}(\mathbf{x}) = \frac{1}{4}[\phi(\mathbf{x}) + 2\kappa_d(\mathbf{x})(\vec{n}\cdot\vec{\nabla})\phi(\mathbf{x})], \tag{11}$$

$$\Gamma_d^{ref}(\mathbf{x}) = \frac{1}{4}F_{dr}(\eta)\ [\phi(\mathbf{x}) - 2\kappa_d(\mathbf{x})(\vec{n}\cdot\vec{\nabla})\phi(\mathbf{x})]. \tag{12}$$

Finally, substitute (11) and (12) into (8) and use the average Fresnel transmission ratio $A(\eta) = 1 + F_{dr}(\eta)/1 - F_{dr}(\eta)$ to simplify the resulting expression.

5.4.4 Result 1: Diffusive Source Boundary Condition

$$\phi(\mathbf{x}) + 2A(\eta)\kappa_d(\mathbf{x})(\vec{n}\cdot\vec{\nabla})\phi(\mathbf{x}) = \frac{4}{F_{dt}(\eta)}\Gamma_s(\mathbf{x}). \tag{13}$$

6 Finite Element Diffusion Equation

In the last section, we derived the diffusive source boundary condition (DSBC) to complete the approximation of our heterogeneous scattering problem. In this section, we solve this problem using the FE method. The result is our second key result: the FEDE.

6.1 Overview

The derivation of our FE solution follows that of any second-order partial differential equation (PDE), like the DE. It has two steps. The first converts the PDE into a new problem, the weak form: given a space of functions $\mathbb{H}$, find the function in $\mathbb{H}$ that optimizes an objective function derived from the PDE. The new weak solution generalizes the solution of the original PDE: it is either the exact solution or its closest approximation in $\mathbb{H}$. Next, in the second step, one assumes that $\mathbb{H}$ has a finite basis. With this basis, the weak solution can then be expressed as the solution of a linear system.

All FE solutions have three important properties [1], [14], [36]. First, if all the PDE's coefficients are bounded and have bounded derivatives,[4] then the FE matrix is positive definite and always has a solution. Second, the components of the FE matrix are nonzero, only when the support of the basis functions overlaps. If the basis functions are spatially compact (as most common bases are), then the FE system is sparse and can be computed efficiently. Third, the FE solution is the optimal approximation (as measured by the L_2 norm) in the given basis to the PDE's true solution. Together these three properties ensure that the FE solution is both accurate and efficient.

In the rest of this section, we derive the FE solution to the diffusion equation, the finite element diffusion equation. Our derivation is similar to the general elliptic weak form [14], but we tailor the general form specifically to the diffusion problem. To make the equations in this derivation more compact, we omit, in any term, both the independent spatial variable $\mathbf{x}$ and the index of refraction variable η.

6.2 Weak Form

First, we derive the weak form of the DE. In this step, let $\mathbb{H}$ be an arbitrary space of functions and let θ be any function in $\mathbb{H}$. Then multiply (3) by θ and integrate over the scattering domain Ω to obtain

$$-\int_\Omega \vec{\nabla}\cdot(\kappa_d(\mathbf{x})\vec{\nabla}\phi(\mathbf{x})))\theta\, dx + \int_\Omega \sigma_a\phi\theta\, dx = \int_\Omega Q_0\theta\, dx. \tag{14}$$

By definition, the left hand side of the weak form must be a bilinear functional over the functions ϕ and θ. To achieve this, first use the divergence theorem to integrate the first term by parts as under,

$$\begin{aligned}\int_\Omega \kappa_d\vec{\nabla}\phi\cdot\vec{\nabla}\theta\, dx - \int_{\partial\Omega}\kappa_d(\vec{n}\cdot\vec{\nabla})\phi\theta\, ds \\ + \int_\Omega \sigma_a\phi\theta\, dx = \int_\Omega Q_0\theta\, dx.\end{aligned} \tag{15}$$

Second, use the DSBC (13) to eliminate the gradient term in (15). This yields:

3. To ensure that resulting BC has the correct signs, $\vec{\omega}$ and $\vec{n}$ must be consistently defined. We have chosen to always define: $\vec{n}$ as pointing out of the material and $\vec{\omega}$ as pointing away from $\mathbf{x}$.

4. This is simpler but stronger than necessary. It is sufficient that the coefficients lie in a first-order Sobolev space. See [14] and [1] for more detail.

6.2.1 Diffusion Weak Form

Find $\phi \in \mathbb{H}$ *such that* $\forall\, \theta \in \mathbb{H}$,

$$\begin{aligned}\int_\Omega \kappa_d \vec{\nabla}\phi \cdot \vec{\nabla}\theta \, dx + \int_\Omega \sigma_a \phi\theta \, dx + \frac{1}{2A}\int_{\partial\Omega} \phi\theta \, ds \\ = \int_\Omega Q_0 \theta \, dx + \frac{2}{AF_{dt}} \int_{\partial\Omega} \Gamma_s \theta \, ds.\end{aligned} \tag{16}$$

6.3 Matrix Equation

To finish the derivation, we convert (16) into a linear system. First, assume that $\mathbb{H}$ has a finite basis:

$$\mathcal{B}(\mathbf{x}) = \{\beta_0(\mathbf{x}), \beta_1(\mathbf{x}), \ldots, \beta_{n-1}(\mathbf{x})\}.$$

Given this basis, the weak form is equivalent to ensuring that (16) holds only for each $\beta_i \in \mathcal{B}$. If $n = |\mathcal{B}|$, then this is a system of n equations for ϕ:

$$\begin{aligned}\int_\Omega \kappa_d \vec{\nabla}\phi \cdot \vec{\nabla}\beta_i \, dx + \int_\Omega \sigma_a \phi\beta_i \, dx + \frac{1}{2A}\int_{\partial\Omega} \phi\beta_i \, ds \\ = \int_\Omega Q_0 \beta_i \, dx + \frac{2}{AF} \int_{\partial\Omega} \Gamma_s \beta_i \, ds \qquad \forall \beta_i \in \mathcal{B}.\end{aligned} \tag{17}$$

Next, by its definition in the diffusion weak form (16), $\phi \in \mathbb{H}$ and constants a_i exist such that $\phi = \sum_{i=0}^{n-1} a_i \beta_i$. Expanding ϕ in (17) results in a system of linear equations for the solution's coefficient vector $\vec{\mathrm{a}}$. This linear system is as follows:

6.3.1 Result 2: Finite Element Diffusion Equation

$$\mathrm{F}\vec{\mathrm{a}} = (\mathrm{D} + \mathrm{M} + \mathrm{S})\vec{\mathrm{a}} = (\vec{\mathrm{q}} + \vec{\mathrm{g}}) = \vec{\mathrm{r}}, \tag{18}$$

where

$$\begin{aligned}\mathrm{D}_{ij} &= \int_\Omega \kappa_d \vec{\nabla}\beta_i \cdot \vec{\nabla}\beta_j \, dx, & \vec{\mathrm{q}}_i &= \int_\Omega Q_0 \beta_i \, dx, \\ \mathrm{M}_{ij} &= \int_\Omega \sigma_a \beta_i \beta_j \, dx, & \vec{\mathrm{g}}_i &= \frac{2}{AF_{dt}} \int_{\partial\Omega} \Gamma_s \beta_i \, ds, \\ \mathrm{S}_{ij} &= \frac{1}{2A}\int_{\partial\Omega} \beta_i \beta_j \, ds.\end{aligned}$$

7 Query Function

Finally, to finish the derivation of our FE subsurface rendering algorithm, we derive our third key result: the render query function. This function converts the fluence solution of the FEDE into excitant subsurface radiance. Due to the approximations of the DE and its FE solution, an accurate query function must address two computational issues.

First, as the isotropy condition of the diffusion approximation breaks down, the DE solution can begin to contain erroneous angular variation. To avoid these errors, this variation must be averaged. Since, as the diffusion approximation becomes more accurate, the true solution and the average solution converge [24], this averaging tends to remove artifacts without considerably increasing overall error. Second, as part of our FE solution, the fluence $\phi(\mathbf{x})$ was projected onto a finite mesh basis. Since, for most bases, this projection is a more accurate approximation of $\phi(\mathbf{x})$ than its gradient is an approximation of $\vec{\nabla}\phi$, accurate query functions should not compute fluence gradients.

Together the above issues define our query function. To smooth the erroneous angular variation, the initial query averages the subsurface radiance refracted outward from the scattering material:

$$L(\mathbf{x}, \vec{\omega}) = \frac{F_t(\vec{\omega})}{\pi} \int_{(\vec{n}\cdot\vec{\omega})>0} L_d(\mathbf{x}, \vec{\omega})(\vec{\omega}\cdot\vec{n}) \, d\vec{\omega}. \tag{19}$$

Next, this initial radiance query is converted to a fluence query by substituting diffusion approximation (2) and simplifying with (10):

$$L(\mathbf{x}, \vec{\omega}) = \frac{F_t(\vec{\omega})}{4\pi}[\phi(\mathbf{x}) - 2\kappa_d(\mathbf{x})(\vec{n}\cdot\vec{\nabla})\phi(\mathbf{x})]. \tag{20}$$

Finally, we use the diffusive source boundary condition (DSBC) (13) to remove the less accurate gradient term yielding:

7.1 Result 3: Query Function

$$L(\mathbf{x}, \vec{\omega}) = \frac{F_t(\vec{\omega})}{4\pi}\left[\left(1 + \frac{1}{A}\right)\phi(\mathbf{x}) - \frac{4}{AF_{dt}}\Gamma_s(\mathbf{x})\right]. \tag{21}$$

8 Implementation

In the last three sections, we derived the three key results that underlay the basic four-step FE rendering algorithm described in Section 3. This section reviews several implementation details that ensure the algorithm's performance and quality.

8.1 Tetrahedral Mesh Basis

For our FE basis, we chose a piecewise-linear basis on a tetrahedral mesh. To construct this basis, we associate each mesh vertex with a basis function such that each function equals one at its vertex, and linearly decreases to zero across the tetrahedra that include its vertex. Elsewhere, the function equals zero. We chose tetrahedral cells, rather than hexahedral cells like Wang et al. [49], because there are well-studied algorithms for generating high-quality, tetrahedral meshes within a triangular surface mesh. For our examples, we used Tetgen [41] to generate our meshes.

8.2 Assembly of Finite Element Matrix

Given a tetrahedral mesh of the domain, the FEDE ((18)) can be written as a sum of integrals over the tetrahedra volumes and faces. For example, if $\mathcal{T}$ is the set of all tetrahedra and Ω_t is the volume of tetrahedron t, an entry in D_{ij} (see (18)) can be written as

$$\mathrm{D}_{ij} = \sum_{t\in\mathcal{T}} \int_{\Omega_t} \kappa_d \vec{\nabla}\beta_i \cdot \vec{\nabla}\beta_j \, dx. \tag{22}$$

However, the terms in this sum are only nonzero if the supports of β_i and β_j overlap. For our piecewise-linear basis, this happens only if vertices i and j are part of the same tetrahedron. Given this restricted domain of integration, the assembly of (18) can be expressed as a loop over all tetrahedra. Further, we compute integrals like (22), using quadrature. In this case, computing each integral term reduces to computing the sum of the values of the integrand at a series of quadrature points. Our results in Section 9 use

```
SparseMatrix f_mat;
Vector r_vec;

matrix.zero();
rhs.zero();
foreach Tet t in mesh {
  foreach QuadPt pt in Tet {
    foreach Basis i in Tet {
      foreach Basis j in Tet {
        f_mat[i,j] +=
          pt.wt * Kd(pt) * dot(grad(i,pt),(grad(j,pt)));
        f_mat[i,j] +=
          pt.wt * sigA(pt) * value(i,pt) * value(j,pt);
      }
      r_vec[i] += pt.wt * src(pt) * value(i,pt);
  }}

  foreach Face f of Tet {
    if(f on boundary) {
      foreach QuadPt pt on f {
        foreach Basis i in Tet {
          foreach Basis j in Tet {
            f_mat[i,j] +=
              (0.5/A) * pt.wt * value(i,pt) * value(j,pt);
          }
          r_vec[i] +=
            (2/A) * pt.wt * gamma(pt) * value(i,pt);
}}}}}
```

Fig. 6. Pseudocode for the assembly algorithm. Here `Kd(pt)`, `sigA(pt)`, `src(pt)`, and `gamma(pt)` are functions that return the values of $\kappa_d(\mathbf{x})$, $\sigma_a(\mathbf{x})$, $Q_0(\mathbf{x})$, and $\Gamma_s(\mathbf{x})$, respectively. The functions `grad(i,pt)` and `value(i,pt)` return the gradient and value, respectively, of the i^{th} basis function. Finally, `dot` computes a dot product and `pt.wt` is the weight of the quadrature point.

second-order 3D Gaussian quadrature. When using quadrature, the general assembly algorithm reduces to a set of nested loops over tetrahedra, quadrature points and basis functions (see Fig. 6 for the complete pseudocode).

8.3 Material and Source Projection

During the assembly process, the scattering parameters, $\kappa_d(\mathbf{x})$ and $\sigma_a(\mathbf{x})$, and the source functions, $\Gamma_s(\mathbf{x})$ and $Q_0(\mathbf{x})$, are sampled onto a regular spacing of quadrature points. If these terms have sufficiently high frequencies, this sampling will cause aliasing that is visible in the solution. To avoid these errors, the source and material terms are first projected onto the FE basis and then sampled. For our piecewise linear basis, this projection corresponds to sampling the functions at the vertices of the tetrahedral mesh and interpolating within each cell.

8.4 Solving the Linear System

To solve the FEDE, our implementation uses the conjugate gradient algorithm with the symmetric successive over-relaxation preconditioner. If the material coefficients and source functions are bounded and have bounded derivatives, this solution exists. However, if these conditions are violated, the derivatives required to construct the FEDE may not exist or the FEDE matrix may be singular. Additionally, using quadrature could result in a singular equation even if the exactly computed system would have been invertible. Conveniently, the preprojection discussed above avoids these issues, since all functions in our tetrahedral basis satisfy the required bounds, and our quadrature is exact for these linear functions. However, even without projection, including tests with discontinuous material and source functions, our conjugate gradient solver converged.

8.5 Adaptive Mesh Refinement

To ensure accuracy across as range of scales, our implementation uses adaptive mesh refinement to capture subsurface detail. However, to support this refinement we must address two important issues: nonconforming meshes and refinement heuristics.

8.5.1 Nonconforming Meshes

Adaptively refining a mesh creates T-junction vertices when neighboring tetrahedra have differing levels of refinement. These vertices allow the basis to represent discontinuous functions, but they violate assumptions required to derive the weak form. When T-junctions are present, continuity must be imposed by constraining the coefficients of the T-junction basis functions. Creating and enforcing these constraints is a complex, but solved, problem in FE analysis (see [4]). Our implementation uses LibMesh [29] to automatically generate these constraints.

8.5.2 Refinement Heuristics

To refine our meshes, we implemented two simple refinement heuristics. Before the FEDE is assembled, our implementation first refines, once each and in order, all tetrahedra that:

1. are visible from the camera, and
2. span a large range of material coefficients.

To choose the tetrahedra in the second condition, $\sigma_a(\mathbf{x})$ and $\sigma_s(\mathbf{x})$ are computed at all the vertices of the mesh. Then, for each tetrahedron, the largest relative difference among all pairs of vertices is determined. All tetrahedra with a local difference greater than one deviation above the mean are refined.

We considered a third condition that, similar to the material refinement condition, would refine cells along shadow edges. However, even for scenes with point sources, this condition did not increase accuracy. Subsurface scattering naturally blurs all shadow edges slightly. After both of the above conditions were applied, the surface tetrahedra were already sufficiently small to capture these blurred edges accurately. However, our algorithm supports arbitrary refinement and, if necessary, new refinement heuristics could be trivially implemented.

8.6 Single Scattering

Finally, single scattering—light that scatters exactly once in the medium—is usually poorly modeled by the DE. As noted in previous work [28], accuracy can be improved, if the single scattering is simulated more accurately [21] and the diffusion solver is used only for the remaining multiple scattering. To account for the removal of the single scattering in our subsurface computation, we subtract the total power of the single scattering component from the total power of the reduced intensity (RI) source.

9 Results

This section demonstrates that our finite element algorithm is an efficient and accurate rendering method for general heterogeneous subsurface scattering problems. The analysis in this section assesses the results of the new renderer on a

set of four distinct test scenes. This section has five parts. The first two describe the test scenes and the details of the rendering computation. The last three sections use these results to highlight the advantages of the new renderer. Section 9.3 itemizes the costs of the FE algorithm and notes that the new algorithm adds at most 3 minutes to rendering cost; Section 9.4 demonstrates that the test images are nearly identical to exact images produced with a Monte Carlo (MC) path tracer; and Section 9.5 discusses the improvements our algorithm makes over the most advanced previous method [49].

9.1 Scenes

The prototype implementation was tested, using the scenes shown in Fig. 1. Each has a different geometry and material. The test scenes were chosen to demonstrate the range of effects a general, high-quality heterogeneous solver can reproduce.

Bunny uses a marble texture to simulate scattering in a complex, aggregate material. The bunny scene uses a relatively small mesh and no global illumination. This scene emphasizes that in small scenes the FE algorithm only adds a few seconds to the image cost.

Dragon is modeled, using a material similar to translucent plastic. This model shows that our solution can capture smooth changes in color and opacity.

Buddha contains a checkerboard of homogeneous marble and jade-like materials. The material is difficult because the solver must simultaneously capture the sharp edges in the material properties, but also correctly simulate the smooth translucency in thinner geometry.

Geode pushes the limits of our FE algorithm. It demonstrates that the FE approach can easily scale to capture complex, high-frequency scattering, even in a very difficult lighting environment where all light passes through the material.

9.2 Details of the Rendering Computation

All results were generated on a 8×2.66 GHz Xeon workstation with 8 GB of RAM and all images are 640×480 pixels. Tables 1a and 1b summarize the model size, mesh creation costs, and rendering time for each scene. All of these costs are fully parallelized except the mesh creation, the mesh refinement, and the FE matrix solution costs. Since parallel implementations of these operations were not available, these times are performed on a single core. Each image is $16\times$ antialiased but, for our FE solver, antialiasing is essentially free. Once our FE solution is computed, the computation of each sample reduces to a single evaluation of the render query function.

The surface and single scattering components of our images are computed, using a combination of Multidimensional Lightcuts [47] and an analytical single scattering approximation [21]. Our FE implementation is split between Java, which provides an implementation of Multidimensional Lightcuts, and C++, which provides an interface to the LibMesh [29] library. The LibMesh library provides our conjugate gradient matrix solver and mesh iteration and refinement functions.

The Geode model is lit by a small area source; Buddha and Dragon are lit by the Kitchen environment map [9]; and the Bunny is lit by a small spherical source. All images, except the Bunny, approximate global illumination by 100,000 virtual indirect sources.

TABLE 1
Summary of the Rendering Costs and Model Parameters for Our Four Test Scenes: Bunny, Dragon, Buddha and Geode

Model	Initial Tets	Time†	Refined Tets	Time†
Bunny	427,918	40.8s	427,918	0s
Dragon	375,919	59.5s	1,084,599	73s
Buddha	429,158	79.0s	1,369,965	103s
Geode	849,763	60.0s	1,317,804	103s

(a)

Model	Base Costs		FE Costs		Total
	Source	Surface	Assembly	Solve†	
Bunny	4s	32s	9s	5s	50s
Dragon	28s	59s	35s	71s	193s
Buddha	29s	40s	44s	137s	250s
Geode	152s	108s	38s	88s	386s

(b)

Operations marked with † are run on a single processor. (a) Mesh sizes, generation cost and refinement cost. Costs range from 40-123 seconds. (b) Rendering costs by model and category. Rendering times vary from 1 to 6 minutes.

9.2.1 Reference Path Tracer

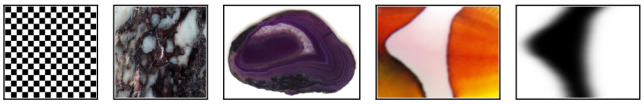
Our reference path tracer is highly optimized. It is fully parallelized and importance samples all subsurface paths. Further, we dramatically accelerate the path tracer by modeling the heterogeneous material as a large, piecewise constant grid of homogeneous cells. To create the path tracing grid, we assign each grid cell the scattering parameters of its center point. Since the interior of each grid cell is homogeneous, the path tracer can generate path segments within each cell without ray marching or repeated material queries. We chose the grid size so that this change had no effect on the final image quality. For all scenes, a grid with 256 cells in the longest dimension was sufficient. We chose the size of the remaining dimensions so the cells were cubical.[5] Without this optimization, the cost of the reference path tracer would have been prohibitive.

9.2.2 Material Parameters

The material models used in the test scenes were synthesized by orthographically projecting an image through the scattering geometry and using the pixel values to determine values for the scattering coefficients σ_a and σ_s. The parameters used in this generation, as well as the source images, are provided in Table 2. Each material coefficient is specified by three colors and an image. The first two colors, range min and max, are used to rescale the image to the dynamic range of the scattering parameter and the last color sets the parameter's overall scale. To compute a scattering parameter at a particular point, one finds the corresponding pixel in the rescaled image, inverts it and multiplies by the base scale. The inversion is necessary because the material parameters specify how much light is

5. To ensure the sharp edges in the Buddha scene, we aligned this rendering grid with the Buddha's larger material grid.

TABLE 2
Parameters Used to Procedurally Generate the Volume Scattering Textures



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

Param.	Range Min	Range Max	Base Scale	Tex.
Dragon σ_a	(.05, .05, .05)	(1.0, 1.0, 1.0)	(0.75, 1.25, 1.75)	(d)
Dragon σ_s	(.25, .25, .25)	(1.0, 1.0, 1.0)	(16.7, 16.7, 16.7)	(e)
Geode σ_a	(.01, .01, .01)	(1.0, 1.0, 1.0)	(5.0, 5.0, 5.0)	(c)
Geode σ_s	constant	constant	(5.0, 5.0, 5.0)	–
Buddha σ_a	constant	constant	(1.63, 1.18, 4.5)	–
Buddha σ_s	(.05, .05, .05)	(1.0, 1.0, 1.0)	(16.7, 16.7, 16.7)	(a)
Bunny σ_a	(.05, .05, .05)	(3.5, 3.5, 3.5)	(7.8, 7.8, 7.8)	(b)
Bunny σ_s	(.60, .60, .60)	(1.0, 1.0, 1.0)	(13.1, 15.7, 18.0)	(b)

Section 9.2.2 describes our material model.

removed during scattering whereas, in the image, the colors specify how much light should be added to each pixel.

9.3 Performance

Table 1 breaks down the cost of the new FE algorithm by component. From new scene to final image, the system requires between 2 and 10 minutes. This cost can be divided into two parts: meshing costs and rendering costs. First, Table 1a lists mesh sizes and the costs of mesh generation and refinement. Because the FE algorithm is agnostic to the mesh discretization, it can use the most efficient meshing algorithms available. For unstructured triangular surface meshes, tetrahedralization is fast (1-2 minutes for all examples) and produces a high quality discretization [41].

Second, Table 1b gives the cost to render each image after the mesh has been computed and fully refined. For the four test scenes, these costs total between 50 s and 6 minutes. These costs are further split into two categories. The base costs—computing the boundary source and rendering the surface component—are performed by a separate surface rendering method and are independent of the subsurface rendering algorithm. Only the costs of assembling the FEDE and solving it are inherent to our new FE algorithm. For all test scenes, these costs total less than 3 minutes and, except for the Buddha image, are less than half of the rendering cost. Thus, in most scenes, the cost of adding subsurface scattering roughly equals the cost of rendering high-quality images without subsurface scattering. For simpler scenes, like Bunny, our FE algorithm adds only 15s to the total render time.

9.4 Comparison with Monte Carlo

Demonstrating the quality of our algorithm, Figs. 1 and 7 compare our Dragon, Buddha, and Geode images with the same images produced by an exact, MC path tracer. Compared to previous work in subsurface rendering, there is impressive agreement between these two sets of images. However, neither set of images is perfect. Despite thorough optimization (Section 9.2.1 above), the path tracing algorithm is still expensive. To avoid excessive computation, the path tracing was performed progressively and was stopped as soon as the noise fell below a level permitting a reasonable comparison. Computing noise-free images would at least double the multiple-hour cost of the path traced references. As expected, the new algorithm has a considerable advantage in performance. It generates noise-free images in a few minutes.

To facilitate the quality comparison, absolute error images are provided. These error images are nearly black, so 4x magnified versions reveal differences. There are two principal differences. First, particularly prominent in the Buddha and Geode images, the path tracer is able to capture highlights from caustic paths that do not scatter within the material. However, since these caustic paths are not part of the subsurface scattering, this is an error of our Multi-dimensional Lightcuts surface render, not our FE algorithm. Second, because the DE treats all diffusive radiance as nearly isotropic, it tends to overestimate the frequency of scattering in thin geometry and near the surface, and, as a consequence, it underestimates contributions from low order scattering events. This slightly darkens regions when they are lit from behind, as in the optically thinner parts of the Dragon and Geode, and slightly lightens highly absorptive regions viewed directly, as in the darker checkers on the Buddha. However, overall, these differences are only visible in the magnified errors images and result from fundamental limitations in the diffusion approximation. Given the orders-of-magnitude difference in performance, these results demonstrate that our FE algorithm is suitable even for high quality rendering applications.

9.5 Comparison to Wang et al. [49]

As a final test, we compare our new renderer to the best existing approach, Wang et al. [49]. Since the focus of our

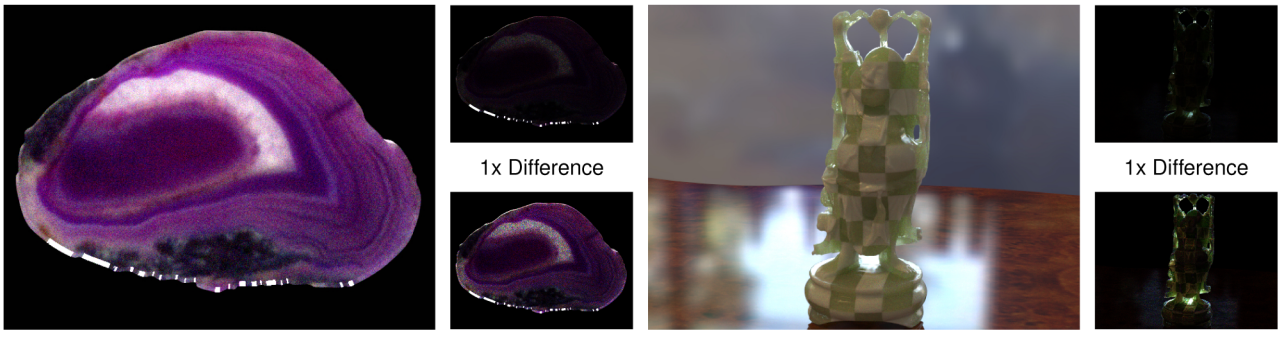


Fig. 7. Path traced reference images for Geode and Buddha from Fig. 1. Difference images are presented on the right of each image.

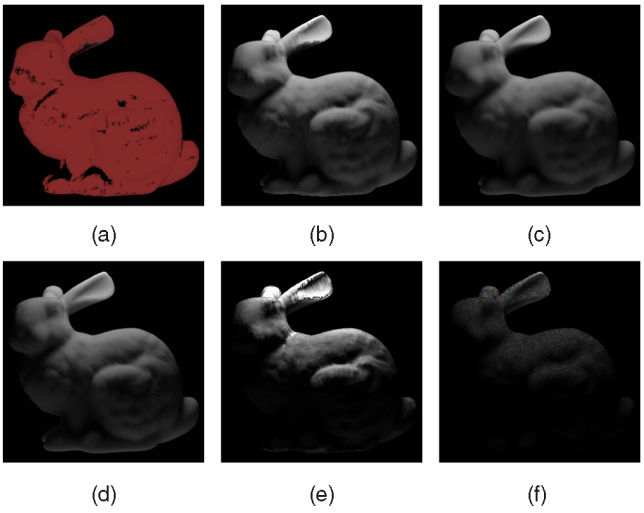


Fig. 8. Images of a constant scattering white Bunny lit by two area lights, fill below and key above: Fig. 8a as described in Wang et al. [49] with negative radiance areas highlighted in red, Fig. 8b corrected using derivation in Section 5, Fig. 8c our FE algorithm, Fig. 8b MC reference, Fig. 8e $4\times$ absolute error of Fig. 8b, and Fig. 8f $4\times$ absolute error of Fig. 8c. (a) FD original. (b) FD corrected. (c) FE. (d) MC. (e) $4\times$ FD error. (f) $4\times$ FE error.

new algorithm is quality, rather than performance, the comparison is made to a software version of their iterative finite difference (FD) algorithm. This software renderer omits the multi-resolution approximation originally required to achieve interactive performance. For our comparisons, we allow the FD algorithm to update fully each step and iterate until the solution converges. However, despite these improvements, we demonstrate that the FD algorithm has lower quality and requires an expensive PolyGrid [45] mesh that introduces rendering errors and causes divergence with some materials.

To facilitate our comparison, Wang et al. [49] kindly provided their measured material data and a PolyGrid [45] Bunny model. As an initial test, several images of the Bunny with a white, homogeneously scattering material (see Fig. 8) were created. Unfortunately, as noted in Section 2.4.1 and [3], their original diffusion formulation was incorrect and, as a result, their original algorithm sometimes computes negative radiance solutions. As shown in red in Fig. 8a, this happens almost everywhere on the homogeneous Bunny. In the original work, this error was corrected by altering the material parameters during the capture optimization and, as a result, this error did not manifest itself in any of the authors' original results. Fig. 9 demonstrates that the material parameters computed using [49] are incorrect. It compares a path traced rendering of the captured artificial stone material with a photograph of the original object. The lack of saturation in the MC result suggests that the acquired parameters have significantly less absorption, making the image brighter. Wang et al. [49] have confirmed the mismatch in Fig. 9. In order to make the remaining comparisons possible, we integrated our correct formulation into Wang et al.'s approach and used this modified algorithm (called corrected FD in Fig. 8) for comparisons.

The rest of the images in Fig. 8 directly compare, for meshes of equal size, Wang et al.'s FD algorithm, the new FE algorithm and a path traced reference (Figs. 8b, 8c, and 8d,

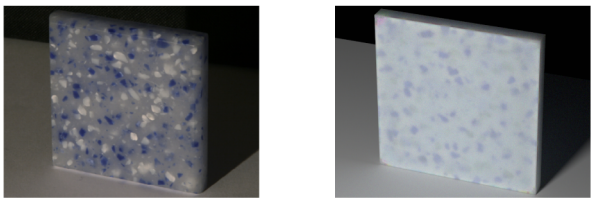


Fig. 9. (a) A photograph of the artificial stone slab captured by Wang et al. [49]. (b) A path-traced rendering of the captured material . Wang et al. [49] have confirmed that this comparison is accurate.

respectively). Figs. 8e and 8f display the error of the FD and FE methods. Since both algorithms depend on the DE, neither solution can produce an exact answer. However, because the FD algorithm relies on a special PolyGrid mesh [45], it has three additional sources of error:

1. The diffusive source boundary condition (DSBC) is enforced only approximately on the boundary.
2. The distortion of the PolyGrid is only approximated in the FD solver.
3. PolyGrid construction requires deleting boundary edge nodes, adding error in these areas.

Additionally, for certain materials, the iterative FD algorithm can also be unstable. In Fig. 10, the Bunny model is rendered with three similar materials, using both the FD and FE algorithms. The first material (left) is homogeneous and the two methods are mostly in agreement. However, in the center and right columns, a checker board is introduced by scaling the mean free path of the material in alternating sections. As this happens, the FD method diverges. In the middle and right columns, the mean free path has been reduced by factors of $3\times$ and $4\times$, respectively. For these cases, the FD algorithm was stopped after 500 iterations to prevent the fluence from overflowing. In both images, the solution is beginning to diverge in the ears, head, and foot. Our algorithm does not have this instability.

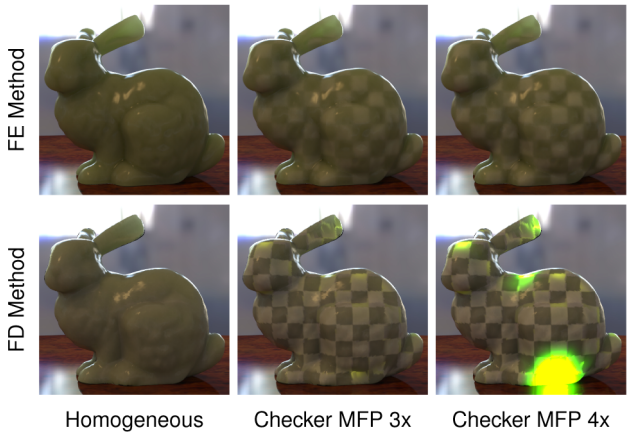


Fig. 10. An example of divergence in the iterative FD solver used by Wang et al. [49]. The Bunny's material is varied from a homogeneous green material (left) toward the material used in the Buddha (see Fig. 1) by scaling the mean free path (MFP) of the material in the lighter squares. The FD algorithm diverges beyond a $3\times$ scale (middle: see spots in ear, head, and foot) and prominently at a $4\times$ scale (right).

Clearly, unlike Wang et al. [49], our algorithm is not interactive but, to compare performance, we must note that in Wang et al. [49] only part of the complete rendering calculation is interactive. The costs of mesh construction were not presented in [49], but the authors noted they required hand optimization which could take hours. Our automated method requires a few minutes (see Table 1a). When considering rendering costs, the surface and source computations are independent of either algorithm and could, as demonstrated in [49], be done in real-time on the GPU. We forgo this optimization for our results, because it introduces considerable error and would prevent our overall solution from matching the path traced reference. In the end, the only directly comparable costs are our matrix assembly and its solution. Even for our mostly single threaded, CPU implementation, for a simple scene (see Fig. 1 and Table 1, Bunny), these steps add only 15 s to image cost.

10 Conclusion

In this paper, we present an efficient, general, and high-quality rendering algorithm for complex heterogeneous materials. To create this algorithm, we correct errors in previous work and derive an accurate diffusive scattering problem. Then, by accurately solving that problem with the FE method, we reduce subsurface rendering to a simple and adaptable four-step algorithm. To validate this algorithm, we create a general implementation and tested its accuracy on a series of four difficult scenes. Our results demonstrate that our algorithm can render in a few minutes images nearly identical to path traced references produced in hours; and that our algorithm significantly improves both the quality and generality of previous methods.

10.1 Limitations and Future Work

Of course, despite its advantages, our algorithm has its own limitations and future work remains. First, our solution is directly limited in accuracy by the diffusion approximation as a scattering model. Similar to [33], interesting future work could consider detecting where the diffusive simulation is inaccurate and computing these inaccurate regions with a more accurate algorithm. Second and related, there is the more general question of rendering diffusive and nondiffusive aggregates of material, such as a crystal, which might have both translucent *and* transparent regions randomly dispersed throughout. Third, by formulating subsurface scattering as a general FE problem, we open the door to the application of the full breadth of FE theory, including robust adaptive refinement, higher order bases, and multiresolution solvers. New work could explore how to better leverage these tools to improve the quality and performance of algorithms like ours'. Finally, there is immediate work in considering how to enhance the performance of our algorithm, both on and off the GPU and CPU.

Acknowledgments

This work was supported by the US National Science Foundation (NSF) CAREER 0644175, NSF CPA 0811680, NSF CNS 0403340, and grants from Intel Corporation and Microsoft Corporation.

References

[1] R.T. Ackroyd, *Finite Element Methods for Particle Transport: Applications to Reactor and Radiation Physics.* Research Studies, 1997.

[2] A. Arbree, B. Walter, and K. Bala, "Single-Pass Scalable Subsurface Rendering with Lightcuts," *Computer Graphics Forum,* vol. 27, no. 2, pp. 507-516, Apr. 2008.

[3] A. Arbree, B. Walter, and K. Bala, "Diffusion Formulation for Heterogeneous Subsurface Scattering," technical report, Cornell Univ. Computing and Information Science, http://hdl.handle.net/1813/14199, 2009.

[4] W. Bangerth and O. Kayser-Herold, "Data Structures and Requirements for *hp* Finite Element Software," Technical Report ISC-07-04-MATH, Inst. for Scientific Computation, Texas A&M Univ., 2007.

[5] N.A. Carr, J.D. Hall, and J.C. Hart, "GPU Algorithms for Radiosity and Subsurface Scattering," *Proc. ACM SIGGRAPH/EUROGRAPHICS Conf. Graphics Hardware,* pp. 51-59, 2003.

[6] C.-W. Chang, W.-C. Lin, T.-C. Ho, T.-S. Huang, and J.-H. Chuang, "Real-Time Translucent Rendering Using GPU-Based Texture Space Importance Sampling," *Computer Graphics Forum,* vol. 27, no. 2, pp. 517-526, Apr. 2008.

[7] Y. Chen, X. Tong, J. Wang, S. Lin, B. Guo, and H.-Y. Shum, "Shell Texture Functions," *Proc. ACM SIGGRAPH '04,* pp. 343-353, 2004.

[8] C. Dachsbacher and M. Stamminger, "Translucent Shadow Maps," *Proc. 14th Eurographics Workshop Rendering Techniques,* pp. 197-201, 2003.

[9] P. Debevec, "Image-Based Lighting," *IEEE Computer Graphics and Application,* vol. 22, no. 2, pp. 26-34, Mar./Apr. 2002.

[10] C. Donner and H.W. Jensen, "Light Diffusion in Multi-Layered Translucent Materials," *Proc. ACM SIGGRAPH '05,* pp. 1032-1039, 2005.

[11] C. Donner and H.W. Jensen, "Rendering Translucent Materials Using Photon Diffusion," *Proc. Eurographics Symp. Rendering Techniques,* pp. 243-251, 2007.

[12] C. Donner, T. Weyrich, E. d'Eon, R. Ramamoorthi, and S. Rusinkiewicz, "A Layered, Heterogeneous Reflectance Model for Acquiring and Rendering Human Skin," *ACM Trans. Graphics,* vol. 27, no. 5, pp. 1-12, 2008.

[13] J. Dorsey, A. Edelman, H.W. Jensen, J. Legakis, and H.K. Pedersen, "Modeling and Rendering of Weathered Stone," *Proc. 26th Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 225-234, 1999.

[14] L.C. Evans, *Partial Differential Equations.* Am. Math. Soc., 1998.

[15] T.J. Farrell, M.S. Patterson, and B. Wilson, "A Diffusion Theory Model of Spatially Resolved, Steady-State Diffuse Reflectance for the Noninvasive Determination of Tissue Optical Properties In Vivo," *Medical Physics,* vol. 19, no. 4, pp. 879-888, 1992.

[16] R. Fattal, "Participating Media Illumination Using Light Propagation Maps," *ACM Trans. Graphics,* vol. 28, no. 1, pp. 1-11, 2009.

[17] A. Ghosh, T. Hawkins, P. Peers, S. Frederiksen, and P. Debevec, "Practical Modeling and Acquisition of Layered Facial Reflectance," *ACM Trans. Graphics,* vol. 27, no. 5, pp. 1-10, 2008.

[18] A.P. Gibson, J.C. Hebden, and S.R. Arridge, "Recent Advances in Diffuse Optical Imaging," *Physics in Medicine and Biology,* vol. 50, no. 4, pp. R1-R43, 2005.

[19] M. Goesele, H.P.A. Lensch, J. Lang, C. Fuchs, and H.-P. Seidel, "Disco: Acquisition of Translucent Objects," *Proc. ACM SIGGRAPH '04,* pp. 835-844, 2004.

[20] T. Haber, T. Mertens, P. Bekaert, and F.V. Reeth, "A Computational Approach to Simulate Subsurface Light Diffusion in Arbitrarily Shaped Objects," *Proc. Graphics Interface Conf.,* pp. 79-86, 2005.

[21] P. Hanrahan and W. Krueger, "Reflection from Layered Surfaces Due to Subsurface Scattering," *Proc. 20th Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 165-174, 1993.

[22] X. Hao, T. Baby, and A. Varshney, "Interactive Subsurface Scattering for Translucent Meshes," *Proc. 2003 Symp. Interactive 3D Graphics,* pp. 75-82, 2003.

[23] X. Hao and A. Varshney, "Real-Time Rendering of Translucent Meshes," *ACM Trans. Graphics,* vol. 23, no. 2, pp. 120-142, 2004.

[24] A. Ishimaru, *Wave Propagation and Scattering in Random Media.* Academic Press, 1978.

[25] H.W. Jensen and J. Buhler, "A Rapid Hierarchical Rendering Technique for Translucent Materials," *Proc. 29th Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 576-581, 2002.

[26] H.W. Jensen and P.H. Christensen, "Efficient Simulation of Light Transport in Scences with Participating Media Using Photon Maps," *Proc. 25th Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 311-320, 1998.
[27] H.W. Jensen, J. Legakis, and J. Dorsey, "Rendering of Wet Materials," *Rendering Techniques,* D. Lischinski and G. W. Larson, eds., pp. 273-282. Springer-Verlag, 1999.
[28] H.W. Jensen, S.R. Marschner, M. Levoy, and P. Hanrahan, "A Practical Model for Subsurface Light Transport," *Proc. 28th Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 511-518, 2001.
[29] B. Kirk, J.W. Peterson, R.H. Stogner, and G.F. Carey, "`libMesh`: A C++ Library for Parallel Adaptive Mesh Refinement/Coarsening Simulations," *Eng. with Computers,* vol. 22, nos. 3/4, pp. 237-254, 2006.
[30] J.J. Koenderink and A.J. van Doorn, "Shading in the Case of Translucent Objects," *Proc. SPIE Human Vision and Electronic Imaging VI,* pp. 312-320, 2001.
[31] V. Kolehmainen, "Novel Approaches to Image Reconstruction in Diffusion Tomography," PhD dissertation, Kuopio Univ., 2001.
[32] H.P.A. Lensch, M. Goesele, P. Bekaert, J. Kautz, M.A. Magnor, J. Lang, and H.-P. Seidel, "Interactive Rendering of Translucent Objects," *Proc. 10th IEEE Pacific Conf. Computer Graphics and Applications,* p. 214, 2002.
[33] H. Li, F. Pellacini, and K.E. Torrance, "A Hybrid Monte Carlo Method for Accurate and Efficient Subsurface Scattering," *Proc. 16th Eurographics Workshop Rendering Techniques,* pp. 283-290, June 2005.
[34] T. Mertens, J. Kautz, P. Bekaert, F.V. Reeth, and H.-P. Seidel, "Efficient Rendering of Local Subsurface Scattering," *Proc. 11th IEEE Pacific Conf. Computer Graphics and Applications,* p. 51, 2003.
[35] T. Mertens, J. Kautz, P. Bekaert, H.-P. Seidelz, and F.V. Reeth, "Interactive Rendering of Translucent Deformable Objects," *Proc. 14th Eurographics Workshop Rendering Techniques,* pp. 130-140, 2003.
[36] K.W. Morton, *Numerical Solution of Convection-Diffusion Problems.* Chapman and Hall, 1996.
[37] M. Pauly, T. Kollig, and A. Keller, "Metropolis Light Transport for Participating Media," *Proc. Eurographics Workshop Rendering Techniques,* pp. 11-22, 2000.
[38] P. Peers, K. vom Berge, W. Matusik, R. Ramamoorthi, J. Lawrence, S. Rusinkiewicz, and P. Dutré, "A Compact Factored Representation of Heterogeneous Subsurface Scattering," *Proc. ACM SIGGRAPH,* pp. 746-753, 2006.
[39] M. Pharr and P. Hanrahan, "Monte Carlo Evaluation of Non-Linear Scattering Equations for Subsurface Reflection," *Proc. 27th Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 75-84, 2000.
[40] M. Schweiger, S.R. Arridge, M. Hiraoka, and D.T. Delpy, "The Finite Element Method for the Propagation of Light in Scattering Media: Boundary and Source Conditions," *Medical Physics,* vol. 22, no. 11, pp. 1779-1792, 1995.
[41] H. Si and K. Gaertner, "Meshing Piecewise Linear Complexes by Constrained Delaunay Tetrahedralizations," *Proc. 14th Int'l Meshing Roundtable,* pp. 147-163, Sept. 2005.
[42] P.-P. Sloan, B. Luna, and J. Snyder, "Local, Deformable Precomputed Radiance Transfer," *Proc. ACM SIGGRAPH,* pp. 1216-1224, 2005.
[43] Y. Song, X. Tong, F. Pellacini, and P. Peers, "Subedit: A Representation for Editing Measured Heterogeneous Subsurface Scattering," *ACM Trans. Graphics,* vol. 28, no. 3, pp. 1-10, 2009.
[44] J. Stam, "Multiple Scattering as a Diffusion Process," *Proc. Eurographics Workshop Rendering Techniques,* pp. 41-50, 1995.
[45] M. Tarini, K. Hormann, P. Cignoni, and C. Montani, "PolycubeMaps," *Proc. ACM SIGGRAPH,* pp. 853-860, 2004.
[46] X. Tong, J. Wang, S. Lin, B. Guo, and H.-Y. Shum, "Modeling and Rendering of Quasi-Homogeneous Materials," *Proc. ACM SIGGRAPH,* pp. 1054-1061, 2005.
[47] B. Walter, A. Arbree, K. Bala, and D.P. Greenberg, "Multidimensional Lightcuts," *Proc. ACM SIGGRAPH,* pp. 1081-1088, 2006.
[48] B. Walter, S. Fernandez, A. Arbree, K. Bala, M. Donikian, and D.P. Greenberg, "Lightcuts: A Scalable Approach to Illumination," *Proc. ACM SIGGRAPH,* pp. 1098-1107, 2005.
[49] J. Wang, S. Zhao, X. Tong, S. Lin, Z. Lin, Y. Dong, B. Guo, and H.-Y. Shum, "Modeling and Rendering of Heterogeneous Translucent Materials Using the Diffusion Equation," *ACM Trans. Graphics,* vol. 27, no. 1, pp. 1-18, 2008.
[50] R. Wang, J. Tran, and D. Luebke, "All-Frequency Interactive Relighting of Translucent Objects with Single and Multiple Scattering," *Proc. ACM SIGGRAPH,* pp. 1202-1207, 2005.
[51] Y. Wang, J. Wang, N. Holzschuch, K. Subr, J.-H. Yong, and B. Guo, "Real-Time Rendering of Heterogeneous Translucent Objects with Arbitrary Shapes," *Computer Graphics Forum,* vol. 29, no. 2, pp. 497-506, 2010.

Adam Arbree received the BS degree in computer science and physics from the University of Florida in 2003, and the PhD degree in computer science from Cornell University in 2009. His dissertation research developed scalable, high-quality rendering algorithms for subsurface scattering. He is currently researching novel imaging solutions for Autodesk®. His latest research focuses on developing fast, scalable, and robust rendering algorithms accurate for a broad range of materials, lighting, and geometry. This work encompasses material models and acquisition, rendering parallelization, scalable rendering methods, shadowing approximations, and the efficient reproduction of complex effects such as volume scattering and motion blur.

Bruce Walter received the BA degree in computer science and physics from Williams College in 1991, and the PhD degree in computer science from Cornell University in 1998. He is a research associate in the Cornell Program of Computer Graphics, currently working on physically based rendering in complex environments, with emphasis on robust, scalable algorithms, general material models, and efficient parallelization. He was previously a postdoctorate with iMagis Group in Grenoble, France, and the lead developer for the trueSpace product at Caligari Corp.

Kavita Bala received the BTech degree from the Indian Institute of Technology (IIT, Bombay), and the SM and PhD degrees from the Massachusetts Institute of Technology (MIT). She is an associate professor in the Computer Science Department at Cornell University. She leads research projects in scalable illumination, perceptually based rendering and modeling, subsurface scattering and volumetric rendering, shadowing algorithms, real-time global illumination, image-based rendering and texturing, texture synthesis, and parallel graphics algorithms. She has coauthored the graduate textbook *Advanced Global Illumination* (AK Peters Publisher, second edition), and has served on more than 30 program committees. She has co-chaired the Eurographics Symposium on Rendering (EGSR), 2005, and Pacific Graphics, 2010. She is a recipient of the US National Science Foundation (NSF) CAREER Award, Cornell's College of Engineering James and Mary Tien Excellence in Teaching Award (2006 and 2009), and the Affinito-Stewart Award. She is a member of the IEEE.