



Precomputed Multiple Scattering for Rapid Light Simulation in Participating Media

Beibei Wang, Liangsheng Ge, Nicolas Holzschuch

► To cite this version:

Beibei Wang, Liangsheng Ge, Nicolas Holzschuch. Precomputed Multiple Scattering for Rapid Light Simulation in Participating Media. IEEE Transactions on Visualization and Computer Graphics, 2020, 26 (7), pp.2456-2470. 10.1109/TVCG.2018.2890466 . hal-01970399

HAL Id: hal-01970399

<https://inria.hal.science/hal-01970399>

Submitted on 5 Jan 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Precomputed Multiple Scattering for Rapid Light Simulation in Participating Media

Beibei Wang, Liangsheng Ge and Nicolas Holzschuch

Abstract—Rendering translucent materials is costly: light transport algorithms need to simulate a large number of scattering events inside the material before reaching convergence. The cost is especially high for materials with a large albedo or a small mean-free-path, where higher-order scattering effects dominate. We present a new method for fast computation of global illumination with participating media. Our method uses precomputed multiple scattering effects, stored in two compact tables. These precomputed multiple scattering tables are easy to integrate with any illumination simulation algorithm. We give examples for virtual ray lights (VRL), photon mapping with beams and paths (UPBP), Metropolis Light Transport with Manifold Exploration (MEMLT). The original algorithms are in charge of low-order scattering, combined with multiple scattering computed using our table. Our results show significant improvements in convergence speed and memory costs, with negligible impact on accuracy.

Index Terms—Participating Media, Precomputation, Multiple Scattering.



1 INTRODUCTION

Computing illumination simulation in scenes with participating media is still a costly process, even with algorithms specifically designed to handle them. To render high-albedo materials, such as milk or skin, we need to simulate a large number of scattering events before convergence. Worse, the overall appearance of these materials is often very smooth, almost uniform, meaning we used a lot of computational power for an almost constant appearance.

In recent years, several algorithms have been introduced for rendering participating media, such as Virtual Ray Lights (VRL) [1], several extensions to photon mapping culminating with UPBP (Unified Points, Beams and Paths) [2] and Manifold Exploration Metropolis Light Transport [3]. All these methods have greatly improved simulation of participating media. They still encounter problems for simulation of materials with high albedo where multiple scattering dominates. Rendering these materials degrades the quality of the simulation for the entire scene (see Figure 1): a lot of CPU power is used for the translucent material, leaving little for the rest of the scene. This explains the large amount of noise in Figure 1(b).

Dipole methods [4], [5] allow fast simulation of translucent materials by approximating surface response and precomputing incoming radiance. This can be combined with single scattering computed from existing simulation algorithms. Complete integration with existing algorithms is difficult because dipole methods usually rely on precomputing incoming radiance on the surface of the object.

We present a method for faster simulation of light transport in translucent material. Our method is easy to integrate with existing illumination simulations algorithms. We provide examples for path

tracing (VRL), photon-mapping (UPBP) and Metropolis Light Transport (MEMLT). Our algorithm uses precomputed tables for multiple scattering, and relies on the original algorithm for single and double scattering. More specifically, we shoot photons from a point light source and a segment source in an infinite medium and we accumulate radiance for samples distributed in two spatial dimension and two angular dimension, or two 4D tables (one for point light source and one for segment source). We then approximate the lobes in these two 4D tables using axial symmetry for point sources and using asymmetric lobes for segment sources, resulting in compact representations of multiple scattering for homogeneous medium. These representations can be used for fast multiple scattering computation in many rendering algorithms by connecting the point / segment from light side and point / segment from camera side directly. Our method results in faster convergence for all algorithms, greatly reducing noise in the pictures with minimal impact on accuracy. Reducing the computational cost for participating media makes more computational power available for the rest of the scene, improving image quality overall.

In the next Section, we review some of the previous work on rendering participating media. In Section 3, we present our algorithm for precomputing multiple scattering and use it in existing algorithms. We explain implementation details in Section 4. We present our results, compare with previous work and analyze performances in Section 5, and conclude in Section 6.

2 PREVIOUS WORKS

Photon Mapping: Chandrasekhar [6] introduced the *radiative transfer equation*, describing radiation transport in participating media. Jensen and Christensen [7] apply this equation to light transport, and presented an algorithm based on Photon Mapping. Jarosz et al. [8], [9], [10] extend this algorithm using beams instead of photons for faster computations, with less noise. Krivánek et al. [2] automatically selects between beams, points and paths in participating media using multiple importance sampling. Bitterli and Jarosz [11] further extend the idea by tracing photon planes and volume.

- B. Wang is with PCA Lab, Key Lab of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, and Jiangsu Key Lab of Image and Video Understanding for Social Security, School of Computer Science and Engineering, Nanjing University of Science and Technology.
- L. Ge is with School of Software, Shandong University.
- N. Holzschuch is with Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France.

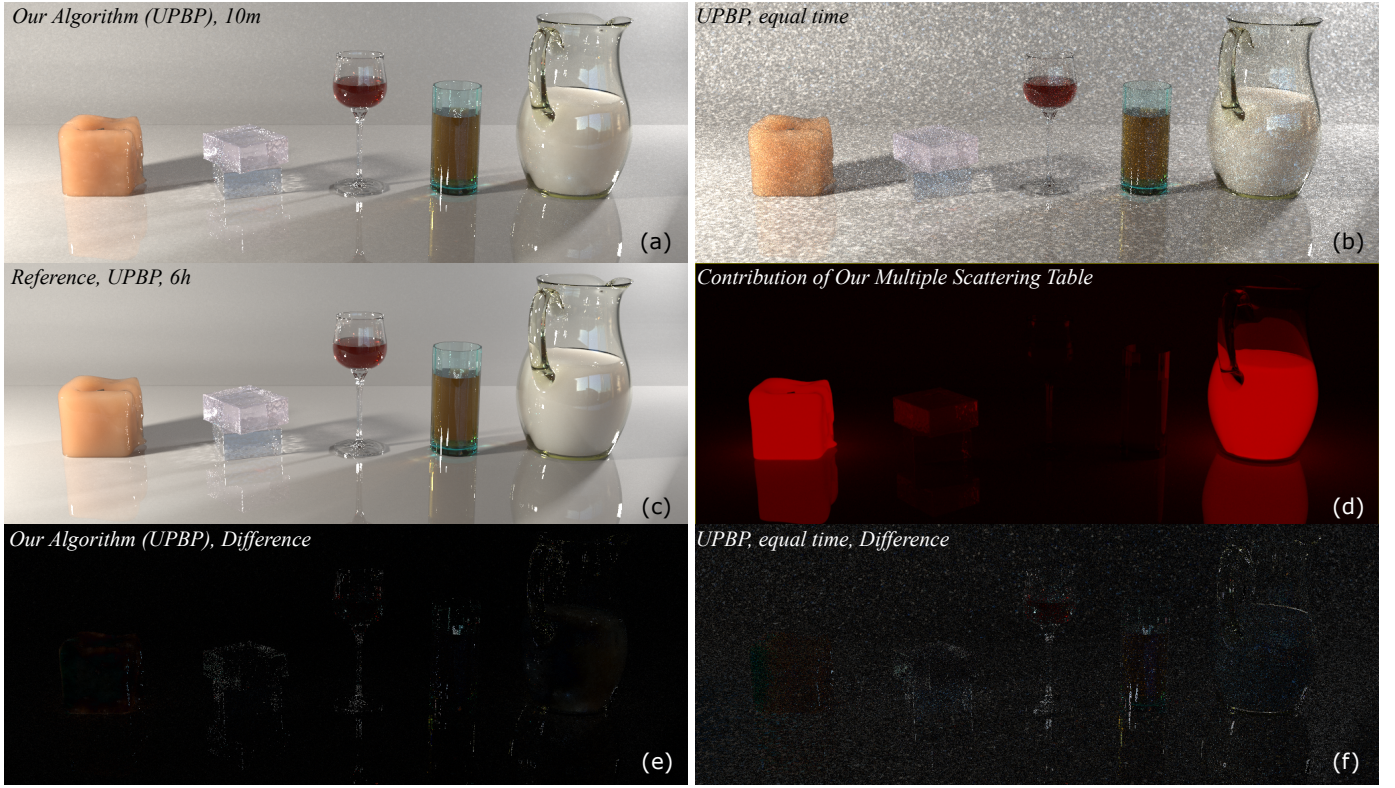


Fig. 1. Our algorithm (UPBP) (a) produces realistic pictures of translucent materials. There is substantially less noise than simulation with UPBP in equal time (b); our picture is indistinguishable from reference (c). Our precomputed tables are used only for multiple scattering effects (d), so mostly for high-albedo, small mean-free-path materials. Reducing computation time for these has improved quality for the entire picture. The difference images between our algorithm (UPBP) / UPBP with equal time and the reference are shown in (e) and (f). The resolution of the pictures is 1024×512 .

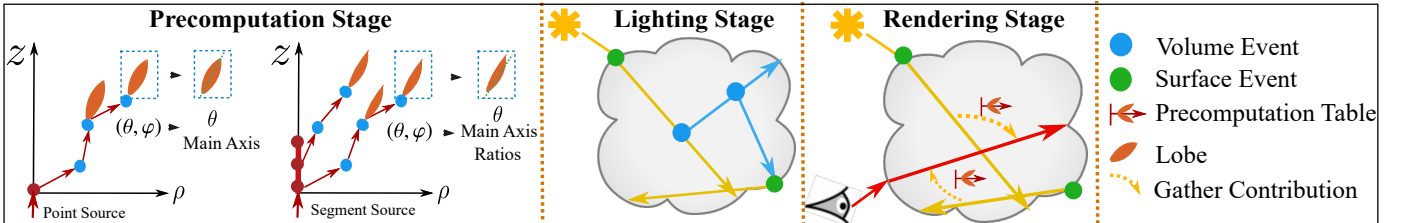


Fig. 2. Our algorithm: we begin by precomputing two multiple scattering tables. We shoot particles from the origin of the point source (red dot) or point samples (red dots) from the segment source. The photons propagate in the imaging medium, being absorbed or scattered. For scattering event, we update the contribution of the photon to each sampling direction to the table. After the shooting, we get two 4 dimension tables. Then we approximate the lobes in these two tables with different features of the lobe: using axially symmetric lobes for point source and with asymmetric lobes for segment source. In the end, we get two much more compact tables. This step is required only once per material. We use these tables for multiple scattering in existing simulation algorithms (here, VRL). In the lighting stage of VRL, we shoot, scatter rays, and store the rays (orange line) with first scatter events. In the rendering stage of VRL, the precomputation tables are used to compute the multiple scattering contribution from the light rays (orange line) to the camera rays (red line).

Photon-mapping-based methods can provide high quality simulations of light transport in participating media, but they usually require a long time to converge. In contrast, our method can provide similar quality with much faster convergence for highly scattering media, as we connect the light point / segment and camera point / segment through the tables with already converged radiance.

Diffusion Theory: Jensen et al. [4] approximate surface response for high-albedo materials using the Dipole method. Jensen and Buhler [5] present a fast method for computing dipole response, relying on precomputed incoming radiance at the surface. D'Eon and Irving [12] improved the accuracy using Quantized Diffusion method. Habel et al. [13] combined photon beams and the diffusion model by distributing extended virtual

sources along the photon beams. Frisvad et al. [14] introduced the *Directional Dipole*, taking into account the orientation of incoming light relative the surface.

Diffusion-based methods are fast, and designed to work with materials with a high albedo, like our own algorithm. Diffusion-based methods tend to be faster than our method to compute multiple scattering effects. In comparison, our method can handle highly anisotropic media, while diffusion theory based methods use similarity theory for anisotropic media, which leads to inaccuracy. Even for isotropic media, our method provides better quality, by tracing the light inside the medium and combining with VRL, UPBP or MEMLT.

Virtual Point Lights: Keller [15] proposed virtual point light method for fast GI computation. Hařan et al. [16] improved it by

introducing spherical lights to avoid singularity. Novák et al. [17] used virtual point lights and virtual ray lights for light transport inside translucent materials. Novák et al. [1] replaced virtual ray lights with virtual beam lights to remove singularities. They used importance sampling for the transfer between camera rays and virtual light sources.

Virtual Point Lights and its variants can simulate multiple scattering with high quality, at the cost of large amount of virtual points or rays, resulting in slow convergence speed, especially for highly scattering media. Compared to them, combining our method with VRL, UPBP or MEMLT is much faster, while providing similar results.

Path tracing / Metropolis Light Transport: Georgiev et al. [18] used the product of geometry and scattering terms for importance sampling in rendering participating media. Compared to our method, the joint sampling method limits to low-order scattering, while our method focuses on high-order scattering. Veach and Guibas introduced Metropolis Light Transport (MLT) for illumination simulation, mutating existing light transport paths to find more efficient transport. Jakob and Marschner [3] proposed a new mutation strategy for Metropolis Light Transport based on manifold exploration (MEMLT), improving the sampling for paths involving specular and highly glossy surfaces.

Tracing light in highly scattering media usually leads to very long paths. These long paths require a large amount of mutations to provide noise free result. When combining with our precomputed table, the paths are shortened, and much fewer mutations are needed.

Precomputation based Methods: Donner et al. [19] precomputed material response on the surface for translucent materials, storing the result as a function of surface position and outgoing direction. They stored double and multiple scattering effects together. Our focus on multiple scattering allows for a more compact representation. We also focus on scattering inside the material, leaving surface diffraction to the original algorithm.

Moon et al. [20] also precompute material response, but store it on a set of concentric spheres or shells. We use instead a planar grid, resulting in fewer interpolation artefacts. Meng et al. [21] and Müller et al. [22] extend the approach for granular materials, while keeping a concentric shell approach.

Wang et al. [23] and Wang and Holzschuch [24] proposed a precomputed table for multiple scattering of a light source in the infinite media. This table was used in a specifically designed extension of Point-Based Global Illumination. We extend their work, but show the approach is compatible with almost all illumination simulation algorithms. We also introduce multiple scattering from a segment source and provide a more compact representation.

3 PRECOMPUTED MULTIPLE SCATTERING SIMULATION

3.1 Radiative Transfer Equation

We consider a scene containing objects with translucent material. Each of these is assumed to be made of an homogeneous material, with index of refraction η , scattering coefficient σ_s , absorption coefficient σ_a and phase function $p(\omega, \omega_t)$. We note ℓ the mean-free path inside the material (mfp), with $1/\ell = \sigma_t = \sigma_s + \sigma_a$.

Radiance leaving point x in the direction ω is the sum of exitant radiance from the nearest surface along this direction and

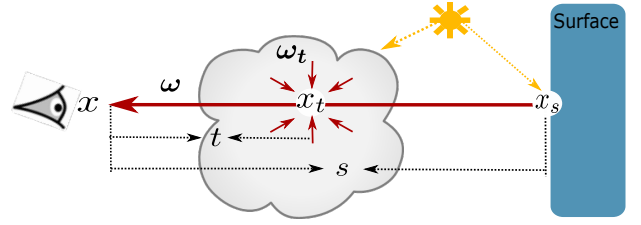


Fig. 3. The Radiative Transfer Equation.

in-scattered radiance from the medium along the whole length of the ray [6] (see Figure 3):

$$L(x, \omega) = T_r(x \leftrightarrow x_s)L(x_s, \omega) + \int_0^s T_r(x \leftrightarrow x_t)\sigma_s(x_t)L_i(x_t, \omega)dt, \quad (1)$$

where T_r is the transmittance, defined as:

$$T_r(x \leftrightarrow x_t) = e^{-\sigma_t\|x-x_t\|}, \quad (2)$$

s is the distance through the medium to the nearest surface at $x_s = x - s\omega$, and $x_t = x - t\omega$ with $t \in (0, s)$. $L(x_s, \omega)$ is the exit radiance from the nearest surface, which is governed by the *rendering equation* [25]. $L_i(x_t, \omega)$ is the in-scattering radiance at x_t from all direction ω_t over the sphere of directions $\Omega_{4\pi}$ using the phase function p , defined as:

$$L_i(x_t, \omega) = \int_{\Omega_{4\pi}} p(\omega, \omega_t)L(x_t, \omega_t)d\omega_t. \quad (3)$$

3.2 Notations and Context

We separate between **single**-, **double**- and **multiple**- scattering effects, depending on the number of volume scattering events inside the translucent material. *Single scattering* corresponds to a light paths with only one scattering event inside the material, *double scattering* to paths with two scattering events and *multiple scattering* to paths with more than two scattering events. We only count the number of scattering events, independently of the number of internal reflections on the specular surface.

Our algorithm is designed to work with any rendering framework. We describe here integration with Virtual Ray Lights (VRL), UPBP and MEMLT. We focus on computing multiple scattering; the original algorithms compute low-order scattering.

3.3 Precomputing Multiple Scattering Effects

3.3.1 Precomputing Multiple Scattering Radiance

In the precomputation step, we assume an infinite, homogeneous, participating medium, with the same properties as the original medium. We run two types of precomputations: with a point directional light source and with a segment light source. A *segment source* is defined with $(\mathbf{o}, \mathbf{v}, l)$, where \mathbf{o} is the origin and set at the spatial coordinates origin $(0, 0)$, \mathbf{v} is the direction of the segment and is aligned with the z axis, l is its length, set at $2 \times \text{mfp}$. This is a compromise, considering both the efficiency and the accuracy. A larger size leads to inaccuracy for short light rays, and smaller size requires more accumulation for a given light ray. We treat the segment as a combination of point sources during the precomputation. All spatial dimensions are normalized by the mean-free-path of the material, to reduce the number of parameters. In both cases, we shoot photons from the light source, let them travel in the medium, being scattered or

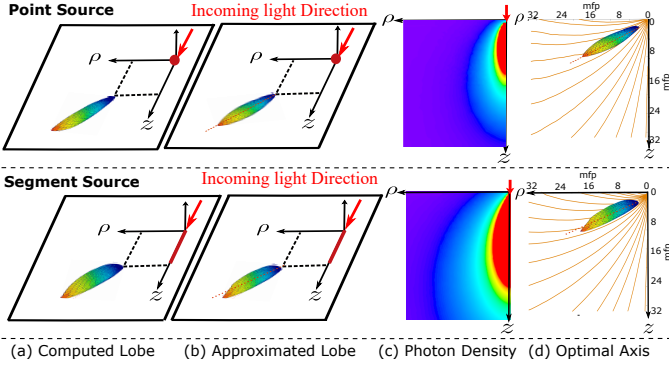


Fig. 4. Multiple scattering from a point source and a segment source: first, we compute lobe shapes indexed by cylindrical coordinates (ρ, z) for position, using spherical coordinates (θ, φ) for direction (a). We approximate these lobes using axially symmetric lobes for point source and with asymmetric lobes for segment source, whose axis (red dashed line) is computed to minimize representation error (b). In (c) and (d), we visualize the photon density with varying color and the optimal axis with stream lines.

absorbed, and accumulate their contributions in a table, exploiting the symmetry of revolution of the problem around the direction of propagation. This table is a discretized representation of the multiple scattering radiance in the medium, sampling in both the spatial and directional domain. The core idea behind the table computation is similar to volumetric photon mapping. Each grid cell in the table works as a kernel, gathering all the particles scattered inside it. In each grid cell, we sample the direction and evaluate the radiance at this grid for each sampled direction, with unit watt per steradian per square metre. The radiance functions L_o for a point source and a segment source at position (ρ, z) with direction (θ, φ) are defined:

$$r_p(\rho, z, \theta, \varphi) = \frac{1}{N \times V_{\rho, z}} \sum_{i=1}^{M_p} \Phi_i \sigma_s p(\theta, \varphi, \omega_i), \quad (4)$$

$$r_s(\rho, z, \theta, \varphi) = \frac{l}{N \times V_{\rho, z} \times K} \sum_{i=1}^{M_s} \Phi_i \sigma_s p(\theta, \varphi, \omega_i). \quad (5)$$

Where N is the total count of photons (we use 500 K in practice, see Table 1). $V_{\rho, z}$ is the volume of grid cell at position ρ, z , M_p and M_s are the number of particles who reach the cell at position (ρ, z) for a point source and a segment source respectively, Φ_i is the flux of the photon i and $p(\theta, \varphi, \omega_i)$ is the phase function of the medium with incoming direction ω_i and outgoing direction (θ, φ) . K is the count of samples for segment source, and we sample the segment source per grid length uniformly in our implementation.

Both problems (point and segment source) have a symmetry of revolution around the direction of propagation. We index points by their cylindrical coordinates (ρ, z) around this axis, and store contributions as lobes in spherical coordinates (see Figure 6). This first representation is 4-dimensional, with a memory cost of 209 MB.

3.3.2 Compressing Multiple Scattering Representation

We then compress this representation, using axially symmetric lobes for point light sources and asymmetric lobes for segment sources (see Figure 4).

The radial plane, defined by ρ and z , is also a symmetry plane for each lobe. We refer to it as the *sagittal plane* (see Figure 5). The intersection between the lobe and the sagittal plane is a

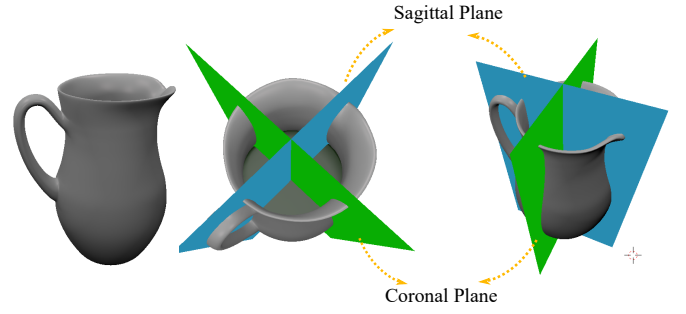


Fig. 5. Definition of the sagittal plane (blue) and coronal plane (green).

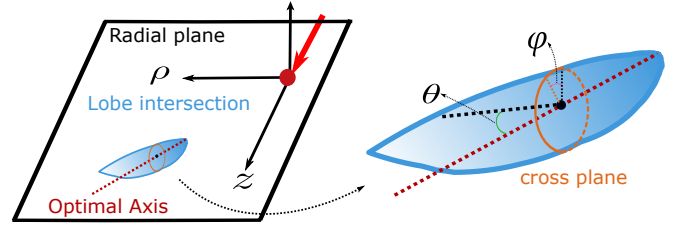


Fig. 6. The radial plane, optimal axis and sagittal cross section. The lobe is parametrized as (θ, φ) , where θ is measured with respect to the optimal axis and φ is the azimuthal angle.

sagittal cross-section. Inside the sagittal plane, we find the axis that separates the sagittal section into similar parts by minimizing the difference between the cross-section and its reflection around the axis. This gives us the *optimal axis*. The plane perpendicular to the sagittal plane that includes the optimal axis is called the *coronal plane*. See Figures 5, 6 and 7.

For point sources, we approximate the lobes using radially symmetric lobes around the optimal axis. For each lobe, we replace the precomputed values for (θ, φ) with a sampled function of θ .

For segment sources, we extend the approach, but take into account the asymmetry of the lobes. We begin by approximating the sagittal cross-section using sampled (R, G, B) values for one side. For the other side of the sagittal cross-section and for the coronal section, we store the ratio between computed values and the sampled function for the first side of the sagittal section.

This representation combines accuracy and compactness. Each lobe shape requires $3n$ samples for point sources, $5n$ samples for segment sources, where n is the number of sample points in θ . The

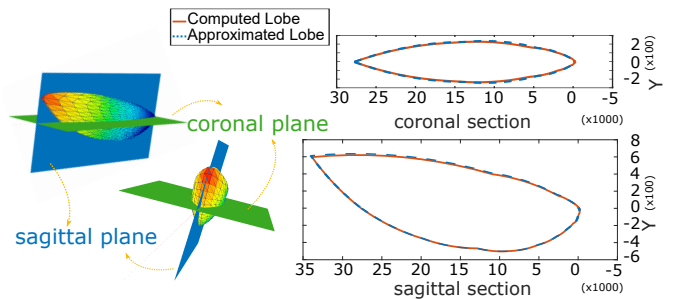


Fig. 7. Sagittal and coronal cross-section. The sagittal plane is a plane of symmetry for the lobe, the coronal plane is perpendicular to it along the optimal axis. For both sections, we visualize both the computed lobe (red) and the approximation (dashed blue).

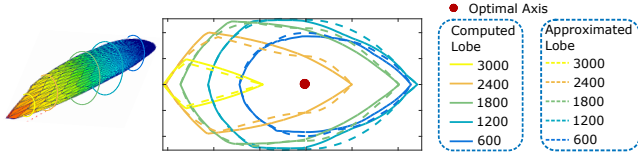


Fig. 8. Comparison between computed lobe and resulting approximation, looking at cross-sections perpendicular to the optimal axis. Medium with anisotropy $g = 0.9$, sample position (5, 5). The curves are isolines with the numbers representing the radiance on the right. The solid lines represent the computed lobe and the dashed lines represent the approximated lobes.

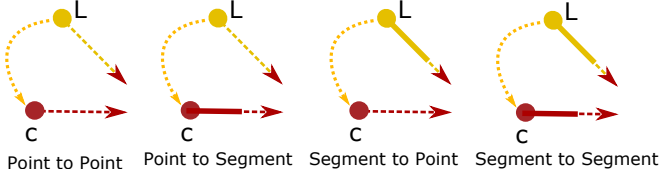


Fig. 9. In rendering, we encounter four different types of radiance estimator.

first half of the sagittal cross-section is stored using (R, G, B) ; the ratios are averaged across (R, G, B) and stored using floats.

Lobe direction is stored using a single float. We use the angle between the optimal axis and the line joining the origin and the current point. Total memory cost after this lobe compression is 9.1 MB.

We further compress our tables using quadrees, resulting in a memory cost of 0.5 to 1.5 MB depending on material properties.

The strongest approximation we made was for asymmetric lobes for segment sources. We found experimentally that it has a limited impact on overall accuracy (see Figures 7 and 8, as well as Section 5.5).

3.4 Querying Values From Our Precomputed Tables

Depending on the rendering algorithm, we need to compute multiple scattering from either a point source or a segment source inside the medium, to either a point destination or a segment destination, resulting in four possible combinations (see Figure 9): point-to-point, point-to-segment, segment-to-point and segment-to-segment. VRL and UPBP use segment-to-segment, MEMLT uses point-to-point. Other rendering algorithms could require the other configurations; for example Beam-Radiance-Estimate would require point-to-segment.

Point to Point Contribution. Computing contributions from a point source to a point destination is a direct reading of the precomputed tables. We first transform the point target and ray direction into the table local coordinates, then extract the outgoing radiance from multiple scattering for this point and direction:

$$z = (\mathbf{P} - \mathbf{v}) \cdot \mathbf{d}, \quad (6)$$

$$\rho = \|(\mathbf{P} - \mathbf{v}) - z\mathbf{d}\|, \quad (7)$$

$$\text{mult.}(\mathbf{P}, \omega_t) = L_o\left(\frac{\rho}{\ell}, \frac{z}{\ell}, T_{(\mathbf{v}, \mathbf{d})}(\omega_t)\right). \quad (8)$$

where (\mathbf{v}, \mathbf{d}) is the position and orientation of the precomputed table, (ρ, z) are the cylindrical coordinates around the axis of propagation, and $T_{(\mathbf{v}, \mathbf{d})}(\omega_t)$ is the direction corresponding to ω_t in the frame defined by (\mathbf{v}, \mathbf{d}) .

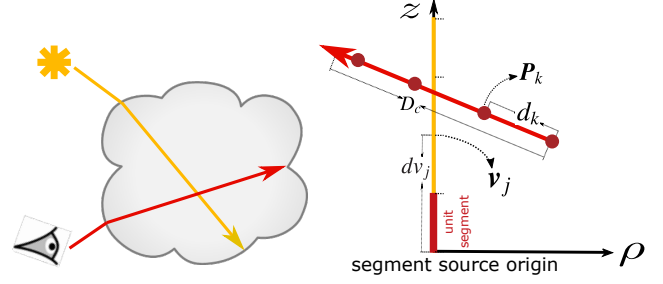


Fig. 10. Contributions from a segment source (orange line) to a segment target (red line) with length D_c . The segment target is transformed into the coordinates of the segment source. We divide the segment source into several unit segments (red square), which have the same length as the precomputed segment source. Each unit segment has a start point \mathbf{v}_j , with distance dv_j from the segment origin. The segment target is sampled regularly into camera samples (red dots), namely \mathbf{P}_k with a distance d_k from the start point of the camera sample. The contributions from the first unit segment are evaluated by directly reading the table, called unit contribution. The contributions from the following unit segments are evaluated by multiplying the unit contribution with the attenuation along the distance.

Point to Segment Contribution. Contributions from a point source to a segment target begin in the same way (Equations 6 and 7). First, we transform the segment into the coordinates of the point source. We then place point samples on the transformed segment, gather outgoing radiance for each sample using the point sampling algorithm. We multiply this radiance with the transmittance along the segment and sum them:

$$\text{mult.}(\mathbf{S}, \omega_t) = \frac{D_c}{K_c} \sum_{\mathbf{P}_k} L_o\left(\frac{\rho_k}{\ell}, \frac{z_k}{\ell}, T_{(\mathbf{v}, \mathbf{d})}(\omega_t)\right) e^{-\sigma_t d_k}, \quad (9)$$

where D_c is the length of the camera ray, K_c is the count of point samples along the camera ray, \mathbf{P}_k is a point sample along the camera ray which is transformed in the local frame defined by (\mathbf{v}, \mathbf{d}) , and d_k is the distance of the sample along the segment target.

Segment to Point Contribution. Contributions from a segment source to a point target also begins in the same way, transforming the point or segment into the coordinates of the segment source. The precomputed segment has a fixed length ($2 \times \text{mfp}$), but the length of the queried light segment may have different length. We subdivide the querying light segment into several unit segments, each having the same length as the segment source. We assume each of these unit segments have the same contribution to the camera sample, except for attenuation along the light segment. We gather radiance for the transformed point as follows:

$$\text{mult.}(\mathbf{P}, \omega_t) = \sum_{\mathbf{v}_j} L_o\left(\frac{\rho}{\ell}, \frac{z}{\ell}, T_{(\mathbf{v}_j, \mathbf{d})}(\omega_t)\right) e^{-\sigma_t dv_j}, \quad (10)$$

where \mathbf{v}_j is the start point of each unit segment along the segment source and dv_j is the distance of this unit segment from the segment source origin.

Segment to Segment Contribution. Contributions from a segment source to a segment target is similar to the contributions from a segment source to a point source, except that we sample the queried camera ray into point samples at first, gather contribution for each of these point samples and then sum them (see Figure 10):

$$\text{mult.}(\mathbf{S}, \omega_t) = \frac{D_c}{K_c} \sum_{\mathbf{P}_k} \sum_{\mathbf{v}_j} L_o\left(\frac{\rho_k}{\ell}, \frac{z_k}{\ell}, T_{(\mathbf{v}_j, \mathbf{d})}(\omega_t)\right) e^{-\sigma_t dv_j} e^{-\sigma_t d_k}. \quad (11)$$

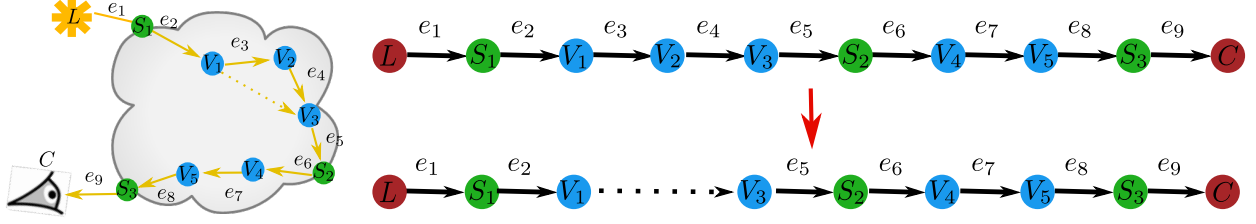


Fig. 11. Application to MEMLT: For each generated seed path, we replace the sub-path with multiple events with one *special* edge. For example, here the subpath from V_1 to V_3 is replaced by the dashed edge. This special edge carries multiple scattering from one side of the edge to the other. When mutating the path, the first endpoint of the special path is not allowed to be a mutation source. The weight of the special edge is computed from the multiple scattering table.

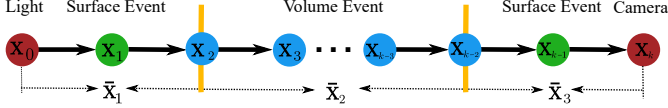


Fig. 12. Definition of subpaths and subspaces. A path x_0, x_1, \dots, x_k is subdivided into three sub-paths: \bar{x}_1 , \bar{x}_2 and \bar{x}_3 . \bar{x}_1 starts from the light source and ends at the first volume event. \bar{x}_2 starts from (not include) the first volume event and ends at (not include) the last volume event. \bar{x}_3 starts from the last volume event and ends at the camera. All the sub-paths which satisfy these three conditions construct three sub-path space: Ω_1, Ω_2 and Ω_3 .

Visibility in the media and Boundaries. We do not consider visibility between the source (segment or point) and the query (segment or point) or boundaries. This is not an issue where the medium has a convex shape, but it can be a source of inaccuracy for scenes with occlusion or air inside the medium. We illustrate this in Section 5.9 and Figures 32 and 33.

3.5 Application to Existing Rendering Algorithms

Our precomputation tables can be used in combination with many existing global illumination algorithms. In this section we describe integration with VRL, UPBP and MEMLT. For each of them, we let the original algorithm handle single and double scattering, and combine the result with multiple scattering from our precomputed tables.

3.5.1 Combination with Virtual Ray Lights (VRL)

Virtual Ray Lights (VRL) [17] is a two-pass ray-tracing algorithm: First render path from the light source inside the medium, with multiple bounces and scattering events and store incoming radiance on these rays. In a second phase, render rays from the camera and accumulate contributions from the rays computed in the first phase.

We follow the structure of the original VRL algorithm, except we do not compute multiple scattering inside the medium. During the first phase, we only store single scattering events in the volume (see Figure 2), but still allow for multiple bounces.

During the rendering phase, we consider rays in the participating medium for each path from the camera. We compute double scattering for these rays using the importance sampling methods described in the original paper [17], and multiple scattering using our precomputed tables. We need segment-to-segment contributions. Note that VRL does not compute single scattering effects.

3.5.2 Combination with Unified Point, Beams and Paths (UPBP)

Unified Point, Beams and Paths (UPBP) is an extension of the Photon Mapping algorithm. In a first step, we send photons and beams inside the scene, including the participating medium, storing incoming radiance at each of them. In a final gathering step, we trace rays from the camera and accumulate contributions from neighbouring photons and beams.

We keep the first phase, shooting photons inside the scene, bouncing on surfaces and scattering inside the medium. We restrict scattering to at most two scattering events, and store single-scattered and double-scattered beams separately.

In the final gathering phase, double-scattered beams are only used to compute double scattering, using the original UPBP algorithm. Single-scattered beams are used both to compute single scattering (using the original algorithm) and to compute multiple scattering using our precomputed tables. For the first event on the camera ray inside a medium, we loop over all the beams stored inside this specific medium and gather their contributions to the camera ray. We need segment-to-segment contributions.

3.5.3 Combination with Manifold-Exploration Metropolis Light Transport (MEMLT)

Metropolis Light Transport (MLT) computes paths from the camera to the light source, then mutates these paths to look for more interesting connections. Manifold Exploration is an extension of MLT, where mutations involving path vertices on specular surfaces are constrained by the specularity.

We position our contribution in a modified path integral framework. We first review the original path integral framework. Image pixel intensity I is an integral over the path space Ω :

$$I = \int_{\Omega} f(\bar{x}) d\mu(\bar{x}). \quad (12)$$

\bar{x} represents a path: x_0, x_1, \dots, x_k , which connects the light source and the camera, where x_0 is on the light source and x_k is on the camera lens, and x_1, \dots, x_{k-1} represents vertices on the surface or scattering event in the media. $d\mu\bar{x}$ is the differential measurement of the path, corresponding to area integration for surface vertices or scattering events. The measurement contribution function $f(\bar{x})$ is the product of the geometry throughput $G(\bar{x})$, scattering throughput $\chi(\bar{x})$, path transmittance $T_r(\bar{x})$ and visibility $V(\bar{x})$.

Given path \bar{x} , without loss of generality, we assume it includes two surface events only and at least three volume scattering event. x_1 and x_{k-1} are surface events and x_2, \dots, x_{k-2} ($k > 5$) are volume scattering events. We divide this path into three subpaths: \bar{x}_1 , \bar{x}_2 and \bar{x}_3 , where $\bar{x}_1 = x_0, x_1, x_2$, $\bar{x}_3 = x_{k-2}, x_{k-1}, x_k$, and $\bar{x}_2 = x_2, \dots, x_{k-2}$.

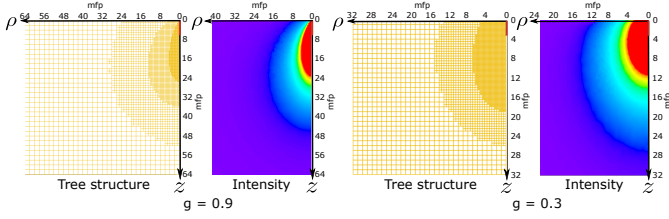


Fig. 13. Precomputed table quadtree structure for segment sources, for materials with anisotropy $g = 0.9$ and $g = 0.3$.

In subpath $\bar{\mathbf{x}}_2$, the endpoints are decided by the other two subpaths. We further divide the path space into three subpath spaces: Ω_1 , Ω_2 and Ω_3 , where Ω_1 represents all the subpaths from the light source to the first volume scattering event, Ω_3 represent all the subpaths from the last volume scattering event to the camera and Ω_2 represents the other scattering events in the volume (see Figure 12). Thus, we have

$$\Omega_1 = \{\bar{\mathbf{x}}_1\}, \Omega_2 = \{\bar{\mathbf{x}}_2\}, \Omega_3 = \{\bar{\mathbf{x}}_3\}. \quad (13)$$

We rewrite Equation 12 with the new subpaths definitions:

$$I = \int_{\Omega_1} f(\bar{\mathbf{x}}_1) \int_{\Omega_3} f(\bar{\mathbf{x}}_3) \int_{\Omega_2} f(\bar{\mathbf{x}}_2) d\mu(\bar{\mathbf{x}}_2) d\mu(\bar{\mathbf{x}}_3) d\mu(\bar{\mathbf{x}}_1). \quad (14)$$

We replace the subpath $\bar{\mathbf{x}}_2$ with a *special edge* (see Figure 11). This edge transports multiple scattering from one path vertex to the other, represented as $g(\mathbf{x}_i, \omega_i, \mathbf{x}_o, \omega_o)$. We merge g into Equation 14:

$$I = \int_{\Omega_1} f(\bar{\mathbf{x}}_1) \int_{\Omega_3} f(\bar{\mathbf{x}}_3) g(\mathbf{x}_2, \omega_1, \mathbf{x}_{k_2}, \omega_{k-2}) d\mu(\bar{\mathbf{x}}_3) d\mu(\bar{\mathbf{x}}_1), \quad (15)$$

where ω_i represent unit vector pointing from \mathbf{x}_i to \mathbf{x}_{i+1} .

The precomputed table is exactly an approximation of function g . Our precomputed table includes G , χ and Tr . We ignore the visibility along the subpath. We replace g with a point to point contribution in our table:

$$I \approx \int_{\Omega_1} f(\bar{\mathbf{x}}_1) \int_{\Omega_3} f(\bar{\mathbf{x}}_3) mul.(\mathbf{x}_2, \omega_1, \mathbf{x}_{k-2}, \omega_{k-2}) d\mu(\bar{\mathbf{x}}_3) d\mu(\bar{\mathbf{x}}_1). \quad (16)$$

To solve the new integral Equation 16, we keep the MEMLT algorithm for all paths outside the medium, as well as for paths with less than two scattering events inside the medium. Both the proposal distribution and the acceptance probability have the same form as MEMLT. We use the multiple scattering evaluated from the table as the weight for $\bar{\mathbf{x}}_2$ and the pdf is set as 1. Both endpoints of the special edge are considered as non-specular, and the end point from the light source can not be considered as the mutation source. The invertibility is not changed in the imagined converged subpaths, except when MEMLT itself has non-invertible cases.

4 IMPLEMENTATION DETAILS

4.1 Precomputed Table Structure

We store our precomputed tables for ρ and z going from 0 to 32 mfp and up to 64 mfp for highly anisotropic materials with large g . Spatial coordinates are normalized by the material mean-free-path (mfp). We compress the spatial representation using a quadtree. The quadtree structure is computed in a preliminary step: first, we shoot a small number of photons to compute the spatial

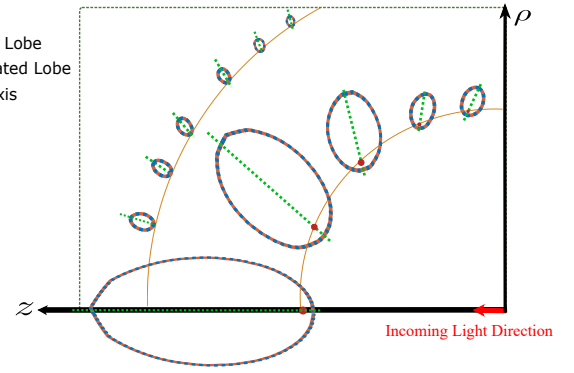


Fig. 14. Comparison between computed and approximated lobes, in the sagittal plane, with their position. Segment light source. Material: $g = 0.8$.

TABLE 1

Parameters for the materials used in this paper. We use 0.5 M particles for the simulations. Memory costs are for both tables, after compression. Computation times are for segment sources (S) and point sources (P).

| Name | α | | | ℓ | | | g | mem. | | time (s) | |
|--------|----------|-------|-------|--------|------|------|-----|------|----|----------|--|
| | R | G | B | R | G | B | | MB | S | P | |
| B.S. | 0.955 | 0.677 | 0.457 | 4.55 | 3.23 | 2.17 | 0.9 | 1.88 | 18 | 2 | |
| Wax | 0.980 | 0.962 | 0.750 | 0.65 | 0.63 | 0.59 | 0.8 | 1.24 | 28 | 2 | |
| Milk | 0.999 | 0.999 | 0.999 | 0.84 | 0.75 | 0.68 | 0.7 | 1.32 | 28 | 2 | |
| Skin | 0.958 | 0.838 | 0.678 | 1.29 | 0.95 | 0.67 | 0.0 | 0.45 | 15 | 2 | |
| Marble | 0.999 | 0.998 | 0.997 | 0.46 | 0.38 | 0.33 | 0.0 | 0.46 | 27 | 3 | |

subdivision. We begin with a 32×32 regular grid, accumulate photons in each cell, and subdivide cells when the photon count reaches a certain threshold (see Figure 13). Once we have the tree structure, we shoot a large number of photons to compute the lobes inside each cell.

4.2 Lobe Approximation

For the lobe approximations, we sample θ every 10° , resulting in 18 samples for each curve. Including the optimal axis position, the memory cost is 55 floats for each lobe for a point light source, and 91 floats for each lobe for a segment source.

The approximation is very close to what we computed originally (see Figures 7 and 8). Figure 14 shows a comparison between the computed and approximated sagittal sections of several lobes in the radial plane, with their position.

5 RESULTS

5.1 Implementation

We have implemented our algorithm inside the Mitsuba Renderer [26] for VRL and MEMLT, and inside smallUPBP [27] for UPBP. All timings in this section are measured on a 2.20GHz Intel i7 (40 cores) with 32 GB of main memory. Unless otherwise specified, all timings correspond to pictures with 512×512 pixels. We report the full computation time, for both our algorithm and the environment. Reference solutions are computed using UPBP and MEMLT.

All materials in our scenes are homogenous materials, with Henyey-Greenstein phase functions and refractive boundaries. Material properties are from Krivánek et al. [2], Narasimhan et al. [28] and Holzschuch [29] (see Table 1).

TABLE 2
Computation time, memory costs and error for our test scenes. The count of multiple beams for all the scenes are 10K.

| | | Ours (VRL) | | VRL | | | Ours (UPBP) | | UPBP | | | Ours (MEMLT) | | MEMLT | |
|------------|------------|------------|-------------------|------------|-------------------|------------|--------------|-------------------|--------------|-------------------|------------|--------------|-------------------|----------|-------------------|
| scene | time. s | VRLs. K | \overline{PSNR} | VRLs. K | \overline{PSNR} | time. m | #iterations. | \overline{PSNR} | #iterations. | \overline{PSNR} | time. m | samples. | \overline{PSNR} | samples. | \overline{PSNR} |
| BumpS. | 53 | 3 | 34.5 | 1 | 23.21 | 1 | 269 | 35.8 | 80 | 19.48 | 1.4 | 1024 | 32.12 | 32 | 24.01 |
| Wax | 180 | 40 | 37.4 | 1 | 12.28 | 3 | 722 | 37.3 | 85 | 21.01 | 1.9 | 1024 | 35.47 | 40 | 23.99 |
| Stilllife. | — | — | — | — | — | 10 | 1160 | 31.83 | 128 | 21.08 | — | — | — | — | — |
| Head. | — | — | — | — | — | 10 | 2044 | 40.59 | 440 | 36.06 | — | — | — | — | — |
| Athena | — | — | — | — | — | 20 | 4093 | 37.0 | 600 | 32.3 | — | — | — | — | — |

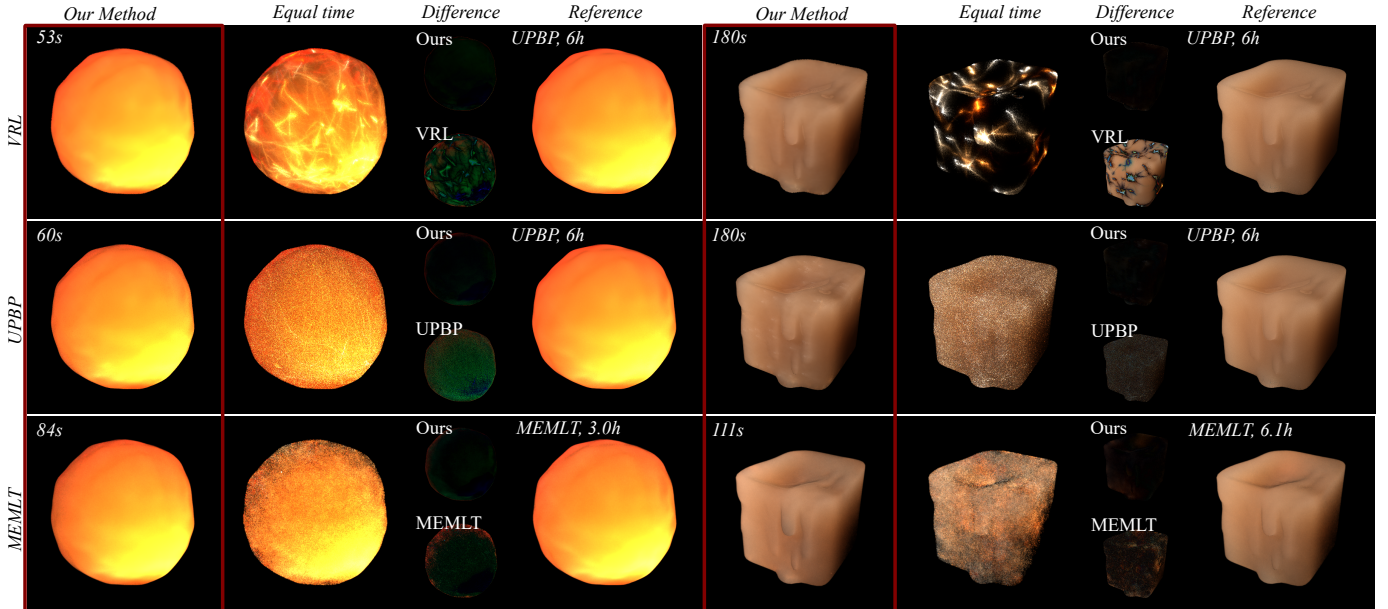


Fig. 15. Comparison (multiple scattering only) between our method and the original methods over two test scenes (bumpy sphere and wax). Our algorithm generates pictures almost identical to the fully converged solution, while being at least one order of magnitude faster. VRL and UPBP use a point light source, MEMLT use an area light source. The BRDFs of both the scenes are dielectric, with IOR 1.5 in Bumpy Sphere Scene and 1.446 in Candle Scene.

5.2 Comparison with High-Quality Methods

Our precomputed multiple scattering greatly speeds up computations of multiple scattering, with all rendering algorithms. Figure 15 shows a comparison between the original methods and our methods, for the three algorithms. We display only multiple scattering for better comparison. Our algorithm produces pictures that are virtually identical to the references, while being at least one order of magnitude faster. We used a point light source for VRL and UPBP, and had to use an area light source for MEMLT.

A strong advantage of our method is that we only change the part of the rendering algorithm dealing with multiple scattering. The rest of the simulation algorithm is unmodified, so we can easily render participating media in a complex setup. Figure 1 shows two such scenes, with multiple refractive interfaces around the participating media. Our algorithm provides noise-free pictures, along with all expected illumination effects, such as diffuse and specular reflections and caustics.

Light transport simulation is more difficult in participating media with refractive interface, due to the absence of a direct path to the light source. Our algorithm also works on index-matched participating media, without refraction at the interface. We have used our method for simulation in mist with varying density (see Figure 16). We observe that the gains are more important on dense mist (see Figure 16 (a)) than for thinner mist (see Figure 16 (b)). This is consistent with our observations on other media:

precomputed multiple scattering is especially interesting when multiple scattering dominates.

5.3 Comparison with Dipole Methods

We also compare our method to fast computation of multiple scattering using the Dipole [4] and Normalized Dipole [30] methods, both implemented in the Appleseed Renderer. Figures 17 and 18 show a side-by-side comparison between our method (UPBP), UPBP, dipole and normalized dipole. We show all scattering levels (single, double and multiple). For Dipole-based methods, we compute single scattering using UPBP and add it to double and multiple scattering computed using the Dipole approximation.

Dipole methods are very fast. Computing single scattering reduces this timing advantage (600 s for Figure 17), but the method is still the fastest to reach a noise-free picture. It tends to overestimate absorption by thin features, such as the edge of the nose in Figure 17 or the helmet in Figure 18. Our algorithm is slightly slower, but still substantially faster than UPBP, and produces pictures that are indistinguishable from the reference.

5.4 Comparison with Diffusion Approximation

The diffusion approximation is based on the observation that the light distribution in highly scattering media tends to become isotropic. In this case, the radiance in the medium from

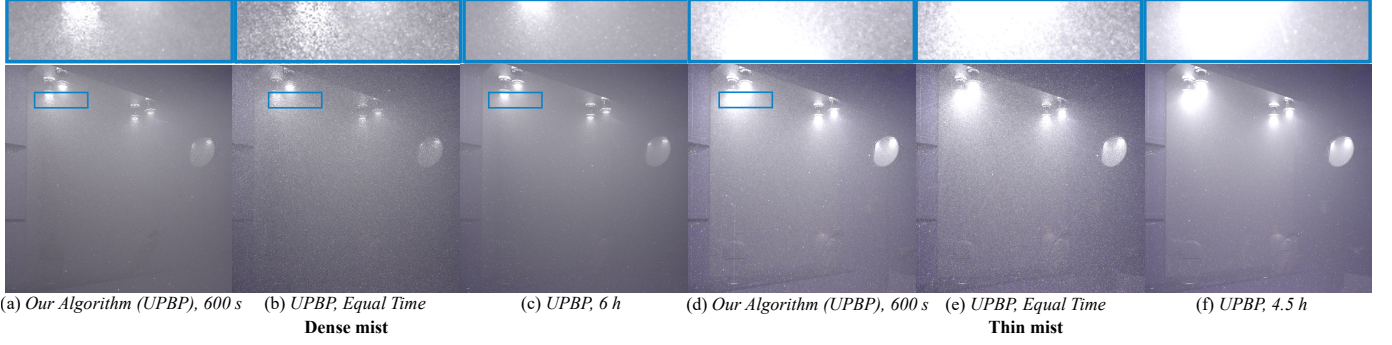


Fig. 16. Comparison (full solution) between our method (with UPBP) and UPBP on the Bathroom Scene, with added mist. Our method provides a good improvement on dense mist (a,b and c), compared to the thinner mist (d, e and f). Material parameters: ($g = 0.4$, $\sigma_s = \{0.016, 0.016, 0.016\}$, $\sigma_a = \{0.001, 0.001, 0.001\}$).

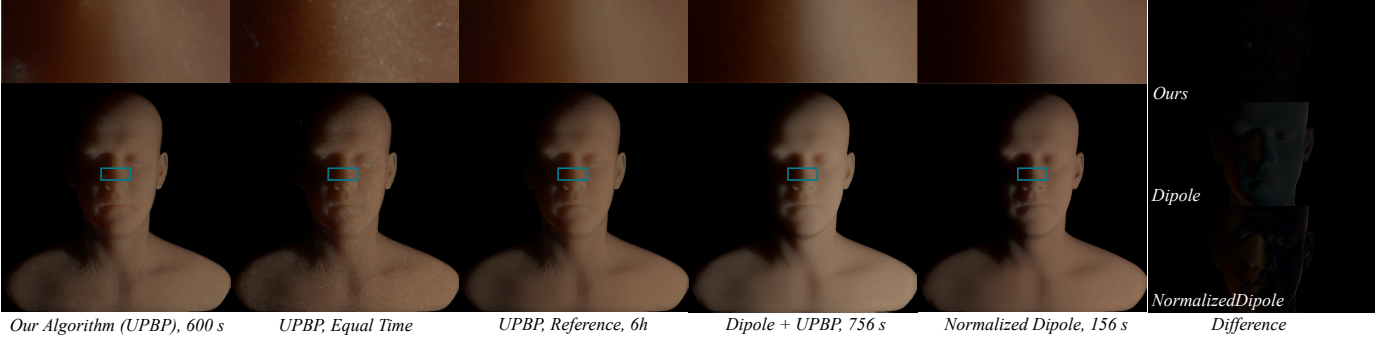


Fig. 17. Comparison with dipole-based methods for the full solution (including single and double scattering). Dipole-based methods are faster, but miss some of the translucency effects. Dipole + UPBP represents using dipole for double + multiple scattering and using UPBP for single scattering. Material: skin.

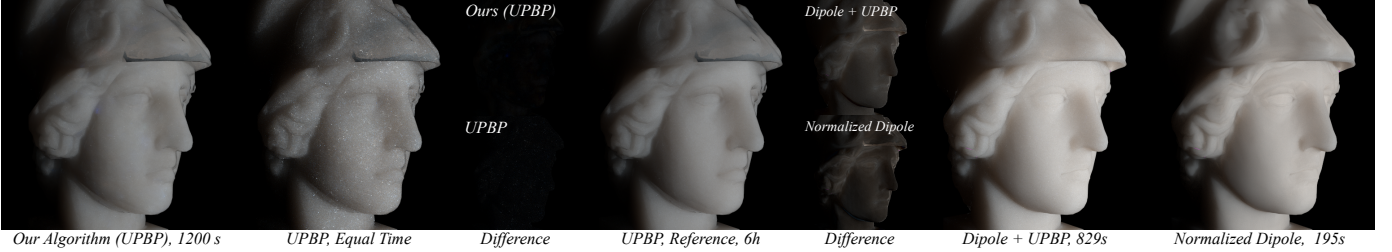


Fig. 18. Comparison with dipole-based methods for the full solution (including single and double scattering). Material: marble.

an isotropic point light source may be approximated by a two term expansion involving the radiant fluence $\phi(x)$ and the vector irradiance $\mathbf{E}(x)$ [4]:

$$L(x, \omega) = \frac{1}{4\pi}\phi(x) + \frac{3}{4\pi}\omega \cdot \mathbf{E}(x), \quad (17)$$

$$\phi(x) = \frac{\Phi}{4\pi D} \frac{e^{-\sigma_t r(x)}}{r(x)}, \quad (18)$$

$$\mathbf{E}(x) = \frac{1 + r(x)\sigma_{tr}}{4\phi r(x)^2} e^{-\sigma_t r(x)} \hat{\mathbf{r}}, \quad (19)$$

$$D = \frac{1}{3\sigma_t}, \sigma_t' = \sigma_a + \sigma_s(1 - g), \quad (20)$$

where r is the distance from the light source, $\sigma_{tr} = \sqrt{\sigma_a D}$ is the effective transport coefficient and $\hat{\mathbf{r}}$ is the unit vector pointing from the source to x .

The main differences between diffusion approximation and our precomputed table with a point light source are: diffusion approximation is limited to highly scattering media, and approximated all

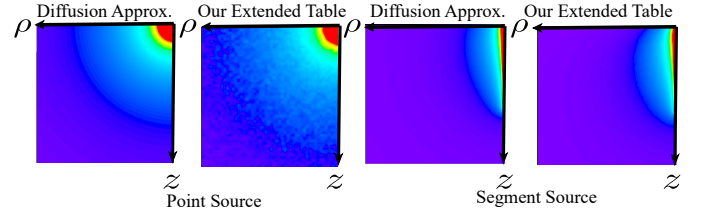


Fig. 19. Comparison between extended precomputed table (including double scattering) and diffusion approximation for a point source in infinite medium and segment source in semi-infinite medium.

media with an isotropic version; diffusion approximation includes both double scattering and multiple scattering, while our method decouples them.

Figure 19 shows a comparison between the values computed by diffusion approximation and the values stored in our precomputed table, to which we added double scattering. For an isotropic

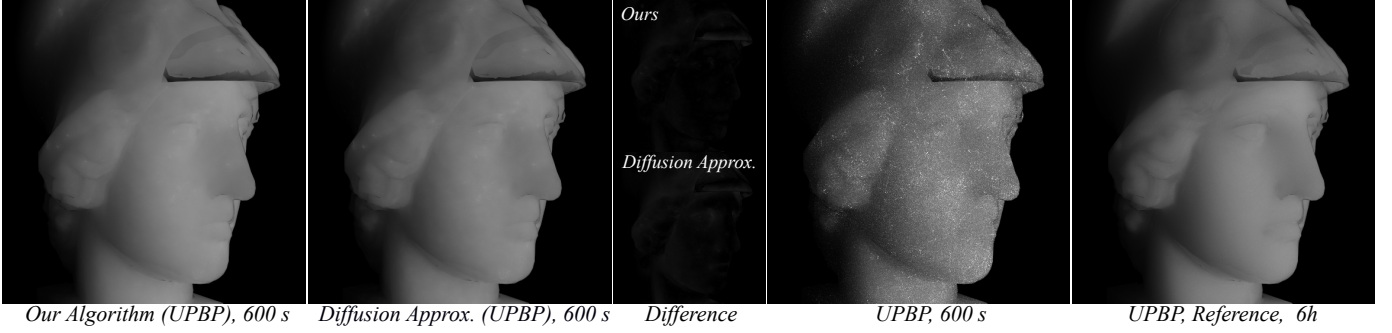


Fig. 20. Comparison between extended precomputed table (including double scattering) and diffusion approximation in UPBP and UPBP as a reference on the Athena Scene. Material: ($g = 0$, $\sigma_s = \{2, 2, 2\}$, $\sigma_a = \{0.01, 0.01, 0.01\}$).



Fig. 21. Comparison between extended precomputed table (including double scattering) and diffusion approximation in VRL and UPBP as a reference on the Bumpysphere Scene. Material: Bumpysphere, $g = 0.9$.

medium ($g = 0$), with low absorption ($\sigma_s = \{2, 2, 2\}$, $\sigma_a = \{0.01, 0.01, 0.01\}$), we get the same results. Figure 20 shows a comparison between the diffusion approximation and our pre-computed table (with double scattering added) for illumination simulation on a simple scene, again with an isotropic medium. We get similar results. Figure 21 shows the same comparison, but with a highly anisotropic material ($g = 0.9$). For this material, the diffusion approximation provides incorrect results, which our table provides results very similar to the reference.

5.5 Precomputed Multiple Scattering

The precomputed tables are an important part of our algorithm. Table 1 gives the computation time and memory cost (after compression) for the materials in our test scenes. Memory costs are almost the same for both tables. Materials with a high albedo require longer computations, as particles are less likely to be absorbed at each event.

Moon et al. [20] store the probability density on a set of concentric spheres or shells, computed in a preprocess step. During the rendering phase, for each scatter event in the medium, they find the sphere with maximum radius with no intersection between the sphere and the geometry. This sphere is sampled to get the next scatter position and direction using the stored probability distribution function. This allows larger step tracing, instead of smaller steps, accelerating computations in participating media (see Figure 23).

Figure 22 shows a comparison between our method and precomputed multiple scattering stored using the concentric spheres [20], with equal memory cost. Our storage method has the same dimension in space (2), and one less dimension in angle. Quadtree-based compression allows for adaptive representation, resulting in better approximation, especially near the propagation axis. Our method also avoids the Mach banding effects of concen-

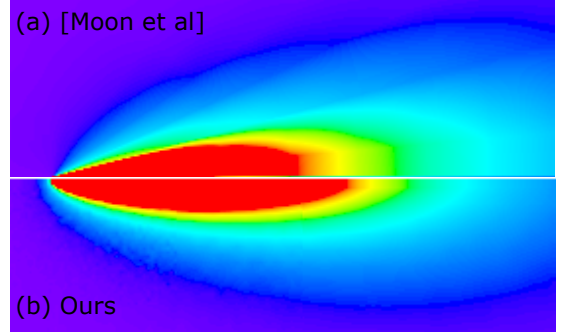


Fig. 22. Comparison between precomputed multiple scattering stored using Concentric Spheres [20] (top) and our method (bottom). Concentric spheres have lower spatial resolution, resulting in poor representation on the axis and Mach banding effects.

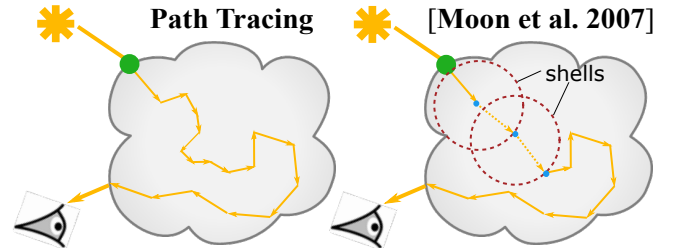


Fig. 23. In the rendering process of Moon et al. [20], a sphere with a maximum radius (the red round circle) is found, called a shell. The next scatter position is on this shell, which is found using the stored pdf, so is the outgoing direction. With this shell tracing, a bigger step is traced instead of a large amount of real scattering event as shown left.

tric spheres, caused by an insufficient number of samples.

We implement Moon et al. [20] method to compare with our method (VRL) (see Figure 24). For equal-time comparison, our method results in a noise-free picture. Both methods save time in material traversal, using different approaches: Moon et al. [20] make a bigger step using the sphere radius and the stored probability distribution function, while our method connects the light ray and the camera ray using stored precomputed multiple scattering.

Moon et al. [20] method was designed for discrete media, such as sand. When used in a continuous medium, we find its efficiency is related to the ratio between the mean-free-path of the medium and the object size. Figure 25 shows a comparison between our method and Moon et al. [20], for varying mean-

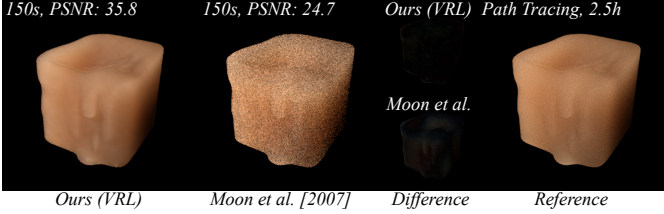


Fig. 24. Comparison between [20] and our method (VRL) with equal time. In this scene, we use a large area light to make path tracing converge faster. The scale of the mfp is 1.

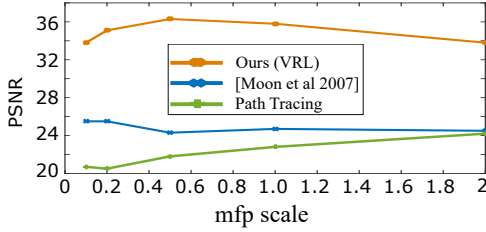


Fig. 25. Comparison between our method (VRL), Moon [20] and Path Tracing for varying mean free path with equal time (150 seconds). A bigger value of PSNR means better quality.

free-path. Their method works better for smaller mean-free-path (meaning more opaque materials), as more scatter events can be saved. Our method provides better quality over the entire mfp range.

We make a strong approximation by replacing the full lobes in spherical coordinates by asymmetric lobes. Figure 26 shows the result of this approximation on multiple scattering: the impact is not visible in pictures. There is a small decrease on the Peak Signal to Noise Ratio (PSNR). We think this limited impact is related to our focus on multiple scattering. Double scattering lobes have a more pronounced elliptic shape [19], even with a point light source.

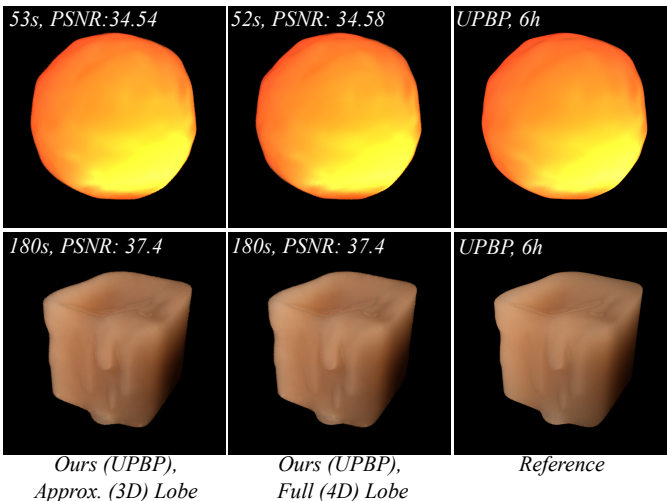


Fig. 26. Approximating lobes as 1D functions has little effect on overall accuracy.

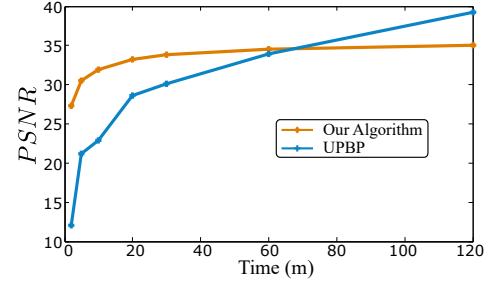


Fig. 27. Error comparison between our method (implemented inside UPBP) and unmodified UPBP as a function of rendering time.

5.6 Performance comparison

Table 2 reports the PSNR between reference pictures and pictures generated with the rendering algorithms, both unmodified and using our precomputed multiple scatter. This quantitative data confirms the qualitative visual impression: the pictures generated by our algorithm are very close to the reference images, and better than the pictures generated by the unmodified algorithms for small computation times (resulting in larger PSNR). The significant reduction in time spent for each interaction allows us to render more iterations within the required time frame (UPBP), to allocate more virtual ray lights (VRL) or more samples (MEMLT).

5.7 Performance analysis: UPBP

Figure 27 shows how PSNR evolves as a function of rendering time for UPBP, both unmodified and using our precomputed multiple scattering, on the Stilllife Scene in Figure 1. Our algorithm contains a strong approximation of multiple scattering effects; as computation time increases, the errors we introduced become dominant, imposing a cap on the PSNR. For long computation time (over 1 h), the unmodified algorithm becomes more accurate.

Our algorithm only applies to multiple scattering. Its advantage depend on the importance of multiple scattering effects in the scene. Figure 28 shows the effect of changing the material mean-free-path while keeping all other parameters constant, along with a density map showing the usage of our precomputed material response. Our algorithm is used often for smaller mean-free-paths, where multiple scattering effects dominate, and scarcely for larger mean-free-paths, where low-order scattering effects dominate.

5.8 Performance Analysis: MEMLT

Figure 30 shows how RMSE evolves as a function of the number of mutations for MEMLT, both unmodified and using our pre-computed multiple scatter, using a log-log scale. For this test, we used a perfect sphere (see Figure 29) to focus on multiple scattering and remove boundary issues. We use the result rendered with MEMLT with 8196 mutation count as the ground truth. The error decreases as the mutation count increases for both methods. We observe the same behaviour as with UPBP: for a small number of mutations, our method has a smaller RMSE. If the number of mutations is large enough (after 4096), the unmodified algorithm provides better results, as our algorithm is biased. On the other hand, the cost for each mutation is much smaller for our algorithm than for unmodified MEMLT, as we only deal with surface mutations. Figure 31 shows how RMSE evolves as a function of time (in log-log scale). For the same computation time, our method consistently outperforms MEMLT, and provides high quality results in a very short time.

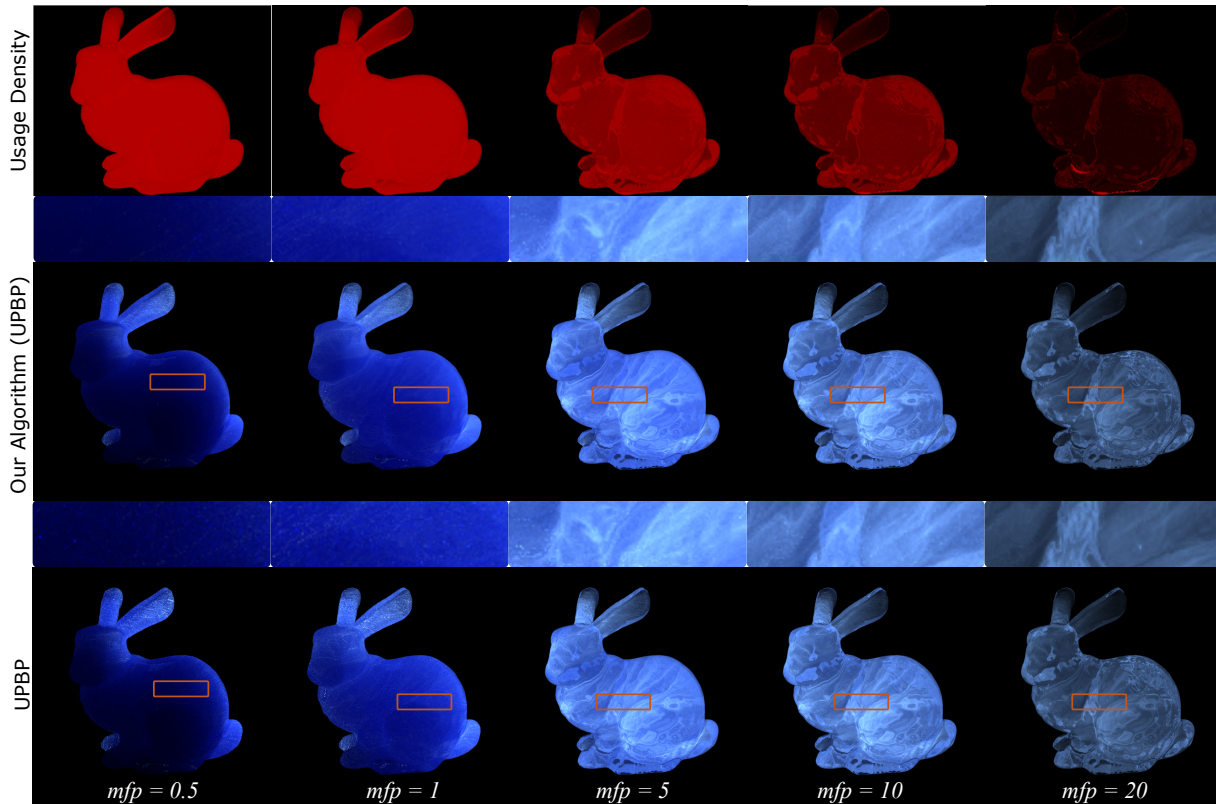


Fig. 28. Comparison between our method (in UPBP) and UPBP, with varying mean-free-paths, rendered with 600 seconds. The albedo of all the materials are set as $\alpha = (0.24, 0.48, 0.96)$. g is 0.3. Top row: usage density for our precomputed table.

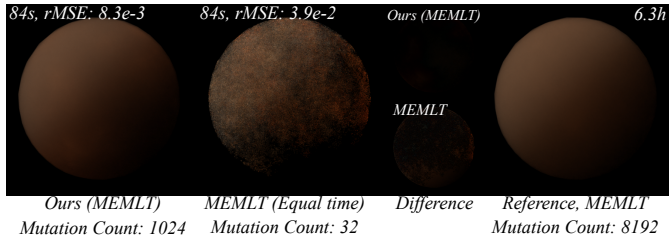


Fig. 29. Comparison between our method (MEMLT) and MEMLT on a Sphere Scene. For this scene boundaries have little effect, allowing us to focus on multiple scattering convergence. Material: Wax.

5.9 Limitations

Our method was designed for homogeneous, continuous, translucent materials with refractive boundaries. We assume that illumination can propagate freely inside the material. This is not an issue in most of our test scenes, as the translucent material blocks have mostly convex shapes. To illustrate the impact of this limitation, we designed two different scenes: in the first, we place a diffuse black ball inside the Candle scene (see Figure 32). In the second, we create a hollow candle, keeping only a shell of thickness 3 mfp. (see Figure 33). For both scenes, the accuracy of our method decreases as it does not account for all the effects. Our algorithm tends to overestimate scattering effects inside the medium.

6 CONCLUSION AND FUTURE WORK

We present a new method to store precomputed multiple scattering in participating media. Our method is easy to integrate with

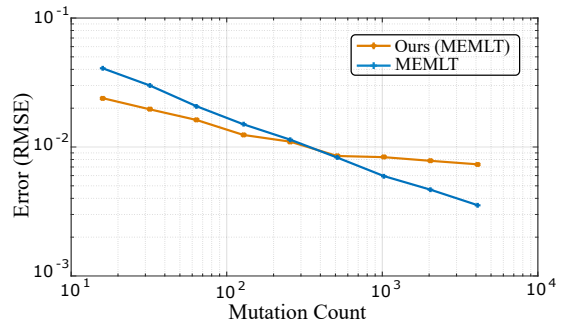


Fig. 30. Error comparison (RMSE) as a function of mutation count, using a log-log scale, for both our method (implemented inside MEMLT) and unmodified MEMLT for a simple Sphere Scene. A smaller value of RMSE means better quality.

existing rendering algorithms. We use the original algorithms for single and double scattering, and our precomputed data for multiple scattering. Our method reduces noise, with limited impact on accuracy. Reducing the cost for simulating participating media allows for faster convergence overall. This method is especially interesting for materials with large albedo and small mean-free-path, where higher-order scattering effects dominate. It has limited impact — but also limited cost — for more transparent materials with larger mean-free-path.

In this paper we used precomputed multiple scattering for all scattering events after the second, and used the original algorithms for single and double scattering. An easy extension is to use the unmodified algorithms for more lower-order scattering (up to the

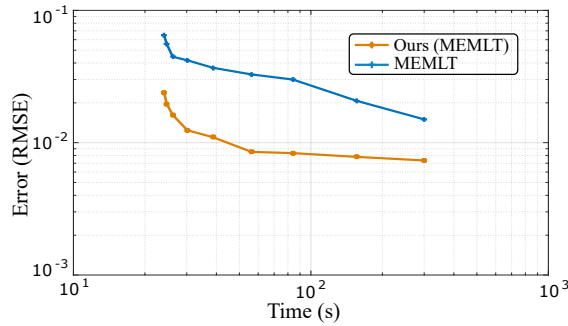


Fig. 31. Error comparison (RMSE) as a function of rendering time, using a log-log scale, for both our method (implemented inside MEMLT) and unmodified MEMLT for a simple Sphere Scene. To keep both methods within the same time range the mutation count for our method is from 16 to 4096, and for unmodified MEMLT it is from 4 to 128.

third or fourth scattering event) and use our precomputed tables for subsequent scattering events.

We also want to consider importance sampling to guide longer path length.

Acknowledgments. We thank the reviewers for the valuable comments. This work has been partially supported by the National Key R&D Program of China under grant No. 2017YFB0203000, the National Natural Science Foundation of China under grant No. 61802187, 61872223, the Natural Science Foundation of Jiangsu under grant No. BK20170857, the fundamental research funds for the central universities No. 30918011320, Open Project Program of the State Key Lab of CAD&CG Zhejiang University under grant No. A1804 and ANR project ANR-15-CE38-0005 "Materials".

REFERENCES

- [1] J. Novák, D. Nowrouzezahrai, C. Dachsbacher, and W. Jarosz, "Progressive virtual beam lights," *Comput. Graph. Forum (Proc. EGSR)*, vol. 31, no. 4, 2012.
- [2] J. Křivánek, I. Georgiev, T. Hachisuka, P. Vévoda, M. Šik, D. Nowrouzezahrai, and W. Jarosz, "Unifying points, beams, and paths in volumetric light transport simulation," *ACM Trans. Graph. (proc. SIGGRAPH)*, vol. 33, pp. 1–13, Aug. 2014.
- [3] W. Jakob and S. Marschner, "Manifold exploration: A markov chain monte carlo technique for rendering scenes with difficult specular transport," *ACM Trans. Graph.*, vol. 31, pp. 58:1–58:13, July 2012.
- [4] H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan, "A practical model for subsurface light transport," in *SIGGRAPH*, pp. 511–518, ACM, 2001.
- [5] H. W. Jensen and J. Buhler, "A rapid hierarchical rendering technique for translucent materials," *ACM Trans. Graph.*, vol. 21, pp. 576–581, July 2002.
- [6] S. Chandrasekhar, *Radiative transfer*. New York: Dover publications, 1960.
- [7] H. W. Jensen and P. H. Christensen, "Efficient simulation of light transport in scenes with participating media using photon maps," in *SIGGRAPH*, pp. 311–320, 1998.
- [8] W. Jarosz, M. Zwicker, and H. W. Jensen, "The beam radiance estimate for volumetric photon mapping," *Comput. Graph. Forum (proc. Eurographics)*, vol. 27, p. 557–566, Apr. 2008.
- [9] W. Jarosz, D. Nowrouzezahrai, I. Sadeghi, and H. W. Jensen, "A comprehensive theory of volumetric radiance estimation using photon points and beams," *ACM Trans. Graph.*, vol. 30, pp. 5:1–5:19, Jan. 2011.
- [10] W. Jarosz, D. Nowrouzezahrai, R. Thomas, P.-P. Sloan, and M. Zwicker, "Progressive photon beams," *ACM Trans. Graph. (proc. SIGGRAPH Asia)*, vol. 30, no. 6, 2011.
- [11] B. Bitterli and W. Jarosz, "Beyond points and beams: Higher-dimensional photon samples for volumetric light transport," *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 36, July 2017.
- [12] E. D'Eon and G. Irving, "A quantized-diffusion model for rendering translucent materials," *ACM Trans. Graph. (proc. SIGGRAPH)*, vol. 30, pp. 56:1–56:14, July 2011.

- [13] R. Habel, P. H. Christensen, and W. Jarosz, "Photon beam diffusion: A hybrid monte carlo method for subsurface scattering," *Comput. Graph. Forum (proc. of EGSR)*, vol. 32, June 2013.
- [14] J. R. Frisvad, T. Hachisuka, and T. K. Kjeldsen, "Directional dipole model for subsurface scattering," *ACM Trans. Graph.*, vol. 34, pp. 5:1–5:12, Nov. 2014.
- [15] A. Keller, "Instant radiosity," in *SIGGRAPH*, pp. 49–56, 1997.
- [16] M. Hašan, J. Křivánek, B. Walter, and K. Bala, "Virtual spherical lights for many-light rendering of glossy scenes," *ACM Trans. Graph. (proc. Siggraph Asia)*, vol. 28, pp. 143:1–143:6, Dec. 2009.
- [17] J. Novák, D. Nowrouzezahrai, C. Dachsbacher, and W. Jarosz, "Virtual ray lights for rendering scenes with participating media," *ACM Trans. Graph. (proc. SIGGRAPH)*, vol. 31, July 2012.
- [18] I. Georgiev, J. Křivánek, T. Hachisuka, D. Nowrouzezahrai, and W. Jarosz, "Joint importance sampling of low-order volumetric scattering," *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2013)*, vol. 32, nov 2013.
- [19] C. Donner, J. Lawrence, R. Ramamoorthi, T. Hachisuka, H. W. Jensen, and S. Nayar, "An empirical bssrdf model," *ACM Trans. Graph.*, vol. 28, pp. 30:1–30:10, July 2009.
- [20] J. T. Moon, B. Walter, and S. R. Marschner, "Rendering Discrete Random Media Using Precomputed Scattering Solutions," in *Rendering Techniques (J. Kautz and S. Pattanaik, eds.)*, The Eurographics Association, 2007.
- [21] J. Meng, M. Papas, R. Habel, C. Dachsbacher, S. Marschner, M. Gross, and W. Jarosz, "Multi-scale modeling and rendering of granular materials," *ACM Trans. Graph.*, vol. 34, pp. 49:1–49:13, July 2015.
- [22] T. Müller, M. Papas, M. Gross, W. Jarosz, and J. Novák, "Efficient rendering of heterogeneous polydisperse granular media," *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, vol. 35, p. 168:1–168:14, December 2016.
- [23] B. Wang, J.-D. Gascuel, and N. Holzschuch, "Point-Based Light Transport for Participating Media with Refractive Boundaries," in *Eurographics Symposium on Rendering 2016 (EI&I)*, (Dublin, Ireland), June 2016.
- [24] B. Wang and N. Holzschuch, "Point-based rendering for participating media with refractive boundaries," *IEEE Transactions on Visualization and Computer Graphics*, 2018.
- [25] J. T. Kajiya, "The rendering equation," *SIGGRAPH Comput. Graph.*, vol. 20, no. 4, pp. 143–150, 1986.
- [26] W. Jakob, "Mitsuba renderer." <http://www.mitsuba-renderer.org/>, 2010.
- [27] J. Křivánek, "SmallUPBP." <http://www.smallupbp.com/>, 2014.
- [28] S. G. Narasimhan, M. Gupta, C. Donner, R. Ramamoorthi, S. K. Nayar, and H. W. Jensen, "Acquiring scattering properties of participating media by dilution," *ACM Trans. Graph. (proc. Siggraph)*, vol. 25, pp. 1003–1012, July 2006.
- [29] N. Holzschuch, "Accurate computation of single scattering in participating media with refractive boundaries," *Comput. Graph. Forum*, vol. 34, no. 6, pp. 48–59, 2015.
- [30] P. H. Christensen, "An approximate reflectance profile for efficient subsurface scattering," in *ACM SIGGRAPH 2015 Talks, SIGGRAPH '15*, pp. 25:1–25:1, 2015.

PLACE
PHOTO
HERE

Beibei Wang is an Associate Professor at Nanjing University of Science and Technology. She received her PhD from Shandong University in 2014 and visited Telecom ParisTech from 2012 to 2014. She worked as a Postdoc in Inria from 2015 to 2017. She joined NJUST in March 2017. Her research interests include rendering and game development.

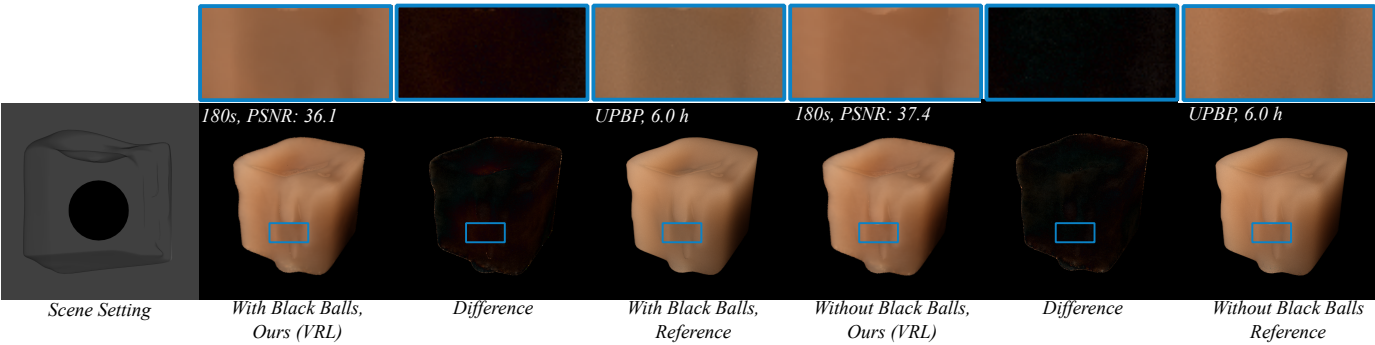


Fig. 32. Comparison between our method (VRL) and the reference (UPBP) for Candle Scene with and without an opaque diffuse black ball placed inside the candle (multiple scattering only). Our method (VRL) provides brighter results than the reference, as it ignores the visibility issues between camera rays and light rays. Material: wax.

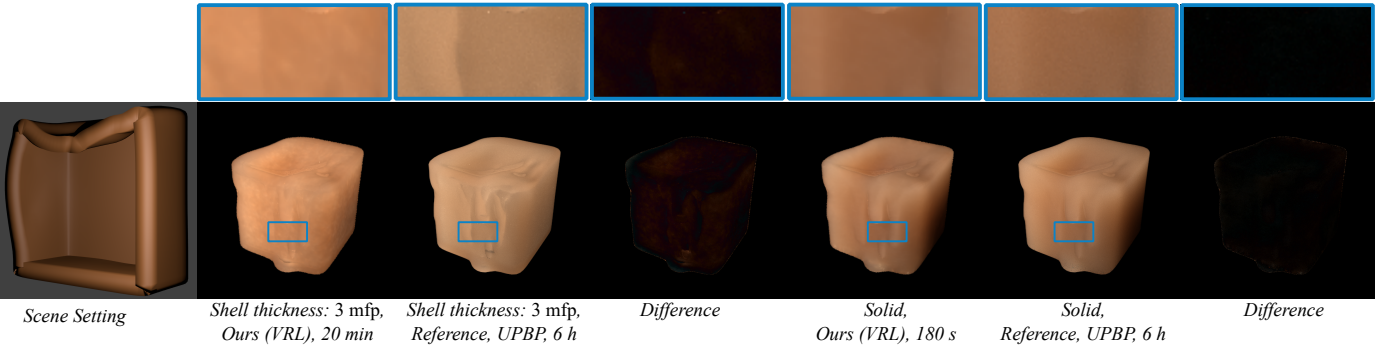


Fig. 33. Comparison between our method (VRL) and the reference (UPBP) for a hollow candle scene (multiple scattering only). We only kept a shell of the candle, with a thickness of 3 mfp. Here, our method tends to overestimate the multiple scattering contribution. Material: wax.



Liangsheng Ge is a first year master student at Shandong University. He received his Bachelor Degree from Nanjing Tech University in June 2017. His research interests include rendering techniques.



Nicolas Holzschuch is a Senior Researcher at INRIA Grenoble Rhône-Alpes, and the scientific leader of the MAVERICK research team. He received his PhD from Grenoble University in 1996 and his Habilitation in 2007. He joined INRIA in 1997. His research interests include photorealistic rendering and real-time rendering, with an emphasis on material models and participating media.