Journal of Information Processing Vol. 17 180–190 (June 2009)

Regular Paper

Rendering Translucent Materials with Plane-parallel Solution

Mikio Shinya,[†1] Michio Shiraishi,[†1]
Yoshinori Dobashi,[†2] Kei Iwasaki[†3]
and Tomoyuki Nishita[†4]

Lighting simulation is very important in realistic image synthesis, and the simulation of subsurface scattering has recently attracted much attention. Although the dipole/multipole model has succeeded in creating realistic images, it is still difficult to deal with volumetric features in subsurface scattering, which is important when rendering optically thin objects. This paper proposes a novel rendering method that utilizes the plane-parallel solution and the ray-marching method. The ray-marching method has been used to calculate single scattering solutions, and the plane-parallel solution has been adopted to calculate BRDFs. By combining these techniques, the proposed method efficiently captures volumetric features in multiple subsurface scattering events. In our experiments, the proposed method demonstrated a performance superior to that of previous methods in terms of accuracy.

1. Introduction

Translucent materials can be seen everywhere in daily life, and rendering these materials is very important for the synthesis of realistic images. Since light scattering is dominant in such materials, an effective and efficient scattering simulation is critical in the rendering process. Light scattering can be described mathematically by a linear integro-differential equation, known as the volume rendering equation. Although the equation can be numerically solved by Monte Carlo methods [1),2)], they are computationally expensive, and several approximated solutions have been proposed.

†1 Toho University
†2 Hokkaido University
†3 Wakayama University
†4 The University of Tokyo

The single scattering approximation is a simple and efficient model for low-albedo media. Ray-marching is a powerful solution in this situation, because the scattered light can be simply evaluated by using a line integral along the viewing ray. However, most translucent objects such as foods and drinks consist of high-albedo materials. Extending this technique to multiple scattering events requires the use of integrals in five dimensions, two with the direction and three with the space, which is computationally impractical.

Jensen, et al. [3),4)] introduced the dipole/multipole diffusion model, which provides a diffusive reflection in an analytical form. Although the dipole/mutipole model has produced realistic images, there are still problems. In the dipole model, the intensity of scattering light is estimated by a sum of influences from a pair of virtual light sources. There are singularities at the dipoles, and the calculation is inaccurate near them. This may cause several practical problems, especially for optically thin objects, where dipoles are located near the surfaces. Directional transmission through objects is also difficult to handle with the dipole/multipole model, which can only simulate almost directionless reflections/transmissions. Texturing is another issue that is hard to treat. The dipole/multipole model tends to blur the irradiance too severely, and the color mixture is neglected by the heuristics that are commonly used.

When layered homogeneous materials with parallel planar boundaries are uniformly illuminated, the volume rendering equation can be simplified to a linear ordinal integro-differential equation. By discretizing the direction, the equation becomes a set of first-order ordinal differential equations, which can be analytically solved. This solution is known as the plane-parallel solution [5)]. The plane-parallel solution is a complete and efficient method to calculate BRDF of scattering materials and has been applied to shading parameter design of skin [6)] and leaves [7)]. However, it relies on strong assumptions regarding the uniform illumination, planar surfaces and homogeneous materials, and it is hard to capture spatial variations by using this technique alone.

This paper presents an efficient and flexible rendering method that couples the plane-parallel solution with the ray-marching scheme. The plane-parallel solution represents the complete distribution of multiple light-scattering in a compact form. Ray-marching integrating the plane-parallel solution instead of simply at-

tenuating the direct light allows us to simulate subsurface scattering in more practical situations including variations in illumination, density, albedo, and so on. With the benefits gained from the ray-marching, this algorithm is more tolerant to the spatial variation, and the use of the plane-parallel solution captures multiple scattering features. This allows us to deal with textures by regarding them as spatial albedo variations. By using integral tables, the computational cost is almost comparable to ray-marching with single scattering. In addition, this technique does not require particular spatial structures such as hierarchical meshes, and can be easily applied to flexible objects with changing shapes and topologies. We made experimental comparisons with the multipole model, and confirmed that the proposed method is significantly more accurate than the multipole method. The method was also successfully implemented on GPU, which performed interactive rendering of scattering objects.

2. Scattering Theories

This section overviews fundamental theories of scattering. We first describe the rendering equation, and then the plane-parallel theory, and finally review the dipole/multipole models.

2.1 Volume Rendering Equation

Light energy propagates through a translucent material via repeated scattering and absorption events. The light intensity $I(x, s)$ at x in the direction s satisfies the volume rendering equation:

$$(\nabla \cdot s)I(x,s) = \sigma_t \left(-I(x,s) + \alpha \int_\Omega p(s,s')I(x,s')ds' \right). \quad (1)$$

Major symbols representing physical properties are listed in **Table 1**. This partial differential equation has an equivalent integral equation form, and the intensity at x can be described by

$$I(x,s) = \int_0^l \left[\exp(-d(l'))\sigma_s \int_\Omega p(s,s')I(l',s')ds' \right] dl' + I_0(x,s), \quad (2)$$

$$d(l') = \int_0^{l'} \sigma_t(l'')dl'', \quad (3)$$

$$I_0 = i_0(s)\exp(-d(l)), \quad (4)$$

Table 1 Symbols.

σ_s	scattering coefficient	d	optical depth along view
σ_a	absorption coefficient	D	optical depth along light path
σ_t	$\sigma_s + \sigma_a$	θ	angle between s and z-axis
α	albedo(σ_s/σ_t)	$\bar{\alpha}$	average albedo
p	phase function	Ω	unit sphere
g	average cosine of p	I	intensity
σ_t'	$\sigma_t - g\sigma_s$	s	direction

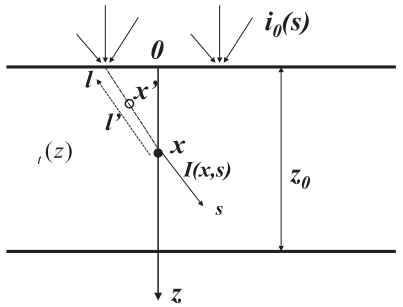


Fig. 1 A layered material with depth z_0.

$$x' = x - l's,$$

where $i_0(s)$ is the incident light distribution at surface point x_0 and I_0 is the attenuated intensity (**Fig. 1**).

This integral equation can be solved iteratively, and the n-th solution is:

$$I^{(n)} = \int_0^l \left[\exp(-d(l'))\sigma_s \int_\Omega p(s,s')I^{(n-1)}(l',s')ds' \right] dl' + I_0(l,s). \quad (5)$$

The single scattering solution $I^{(1)}$ can be calculated using ray-marching by evaluating this line integral.

The volume rendering equation becomes simple when the material is homogeneous and bounded by parallel planes (Fig. 1). In this case, the intensity I only depends on z and s, and gradient ∇ in Eq. (1) is replaced by the ordinal

differential d/dz, as:

$$\cos\theta dI(z,s)/dz = -\sigma_t\left(I(z,s) + \alpha\int p(s,s')I(z,s')ds'\right), \quad (6)$$

where θ represents the angle between s and the z-axis. When the illumination is also uniform, there exist analytic solutions, known as the plane-parallel solutions, as discussed in the next section.

2.2 Plane-parallel Solution

2.2.1 Discretization

The plane-parallel problem can be solved through discretization. Let us discretize the scattering direction s. Using an orthogonal function system on the unit sphere, $\phi_j(s)$, the intensity $I(z,s)$ can be approximated as

$$I(z,s) \simeq \sum_{j=1}^{M} I_j(z)\phi_j(s). \quad (7)$$

As the function system, the spherical harmonic functions were used in Ref. 7), but in this paper, we selected compactly supported piecewise constant functions, which allow more intuitive analyzes of results and a better representation of high frequency components. We subdivide the upper and the lower unit hemi-sphere into $M/2$ domains, Ω_i, respectively, and define the function ϕ_i by

$$\phi_i(s) = \begin{cases} 1/|\Omega_i| & (s \in \Omega_i), \\ 0 & (\text{otherwise}), \end{cases}$$

where $|\Omega_i|$ denotes the solid angle of the domain Ω_i. In this case, I_i simply represents the average intensity over the domain Ω_i.

After substituting Eq. (7) into Eq. (6), multiplying $\phi_i(s)$ on both sides and integrating over the unit sphere discetizes the equation as

$$k_i(d/dz)I_i(z) = \sigma_t\left(-I_i(z) + \alpha\sum_j p_{ij}I_j(z)\right), \quad (8)$$

$$k_i = \int \phi_i(s)\cos\theta ds,$$

$$p_{ij} = \iint p(s,s')\phi_j(s')\phi_i(s)dsds',$$

where α represents the albedo.

2.2.2 Solution

The discretized equation, Eq. (8), can be solved by a matrix eigen value decomposition, which transforms the equation system into a simple diagonal form in the following way. Set

$$I = \begin{pmatrix} I_1 \\ \vdots \\ I_M \end{pmatrix}, \quad P = \begin{pmatrix} p_{11} & \cdots & p_{1M} \\ \vdots & \ddots & \vdots \\ p_{M1} & \cdots & p_{MM} \end{pmatrix},$$

$$K = \mathrm{diag}(k_1,\ldots,k_M),$$

$$Q = K^{-1}(-E + \alpha P),$$

where $\mathrm{diag}(k_i)$ denotes a diagonal matrix with diagonal elements k_i, and E represents the identity matrix. Eq. (8) becomes

$$KdI/dz = \sigma_t(-E + \alpha P)I$$

$$dI/dz = \sigma_t QI. \quad (9)$$

Applying eigen decomposition to matrix Q, we have

$$V^{-1}QV = \Lambda,$$

$$V = \begin{pmatrix} v_1 & \cdots & v_M \end{pmatrix},$$

$$\Lambda = \mathrm{diag}(\lambda_1,\ldots,\lambda_M),$$

where v_j and λ_j are the j-th eigenvector and the eigenvalue of matrix Q. Using this, Eq. (9) can be diagonalized as

$$(d/dz)\tilde{I} = \sigma_t\Lambda\tilde{I}, \quad \tilde{I} = V^{-1}I.$$

The j-th row of this equation is

$$d\tilde{i}_j/dz = \sigma_t(z)\lambda_j\tilde{i}_j,$$

which can be easily solved as

$$\tilde{i}_j(z) = c_j\exp(\lambda_j D(z)), \quad (10)$$

$$D(z) = \int_0^z \sigma_t(z')dz'.$$

By determining the constant coefficient c_j from boundary conditions, the intensity distribution at z can be calculated as

$$I(z) = V\tilde{I}(z) = \sum_j c_j \exp(\lambda_j D(z)) v_j. \quad (11)$$

The boundary conditions can be applied in the following way. For simplicity, let us assume that the elements of the intensity vector I is arranged as

$$I = \begin{pmatrix} i_+ \\ i_- \end{pmatrix},$$

in such a way that $M/2$ dimensional vectors i_+ and i_- represent the incoming components in the positive z orientation and the outgoing components, respectively. When the boundary at $z = 0$ is uniformly illuminated by an environment map with distribution i_0 and the other boundary at $z = z_0$ is not lit, the boundary conditions become

$$i_+(0) = i_0, \quad i_-(z_0) = 0. \quad (12)$$

By substituting Eq. (11) into Eq. (12), the conditions can be expressed as a linear equation system with respect to the coefficients c_i, which can be numerically solved [8].

2.3 Dipole/Multipole Model

The dipole model provides a Bi-directional Scattering Surface Reflectance Distribution Function (BSSRDF), which represents the impulse response of the scattering system. Using BSSRDF S_t, the outgoing light at x, $I(x, s)$, can be calculated from the incoming light $I(x', s')$ and S by integrating over the boundary plane A and the incoming hemisphere Ω_+,

$$I(x, s) = \int_A \int_{\Omega_+} S_t(x_i, s_i; x_o, s_o) I(x', s') dx ds.$$

When we can assume that the intensity $I(x, s)$ is approximated by a simple two term expansion,

$$I(x, s) = (1/4\pi)(\phi(x) + 3(s \cdot \Phi(x))),$$

then $\phi(x)$ satisfies the diffuse equation and the BSSRDF can be calculated analytically as:

$$\begin{aligned} S_t(x_i, s_i; x_o, s_o) &= S_d(x_i, s_i) \\ &= (1/\pi) F_t(x_i, s_i) R(|x_i - x_o|) \times F_t(x_o, s_o), \end{aligned}$$

where S_d represents the BSSRDF for the dipole model, and F_t denotes the Fresnel transmittance due to refraction, which takes $F_t = 1$ when the optical index of the material is 1. Note that the weight function $R()$ neither depends on s_o and s_i, which makes S_d almost uniform over the directions.

When the material is semi-infinitely thick, S_d can be regarded as the sum of the influences from a pair of point sources, or dipoles. One of the dipole is located at $1/\sigma_t'$ below the plane, where σ_t' denotes the reduced extinction coefficient. Note that S_d is singular at the position of the dipole. This never happens when dealing with only planar surfaces, as required by the theory, but becomes problematic when rendering thin objects, as shown in Section 5.

When the material width is finite, several dipoles are placed such that they satisfy the boundary conditions. This extended dipole model is called the multipole model. The multipole model is also capable of handling multiple layers.

3. Ray-marching with Plane-parallel Solution

The plane-parallel solution obtained in the previous section relies on strong assumptions regarding uniform illumination, planar surfaces, and homogeneous materials. Although it is an ideal method for calculating BRDF values of scattering materials, it is hard to capture spatial variations by using this technique alone. In this section, the plane-parallel solution is coupled with ray-marching to provide a practical rendering algorithm for translucent materials.

3.1 Basic Idea

Ray-marching performs the linear integral operation in Eq. (5), which can be regarded as a refinement process that provides a better solution I_n from an approximated solution I_{n-1}. For example, the attenuated direct light, I_0, is refined to the single scattering solution, I_1, by this operation. Our idea is to use the plane-parallel solution, I_{pp}, as I_{n-1} and to apply the ray-marching operation:

$$\begin{aligned} I &= \int_0^l \exp(-d(l'))\sigma_s \left(\int_\Omega p(s, s') I_{pp}(x', s') ds' \right) dl' + I_0(x, s), \quad (13) \\ x' &= x - l's. \end{aligned}$$

We can expect the plane-parallel solution to provide a reasonable approximation when the required conditions are approximately maintained within the scale of several mean free paths. Therefore, the refined solution is expected to provide much better results than the single scattering solution.

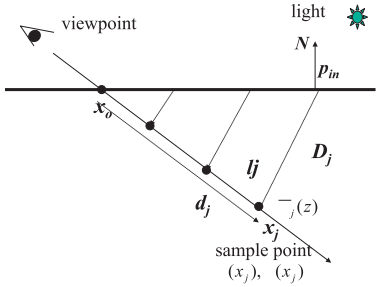


Fig. 2 Ray-marching in a subsurface, where N and x_j denote the normal vector and the position of the j-th sampling point.

The key to successful simulation is the efficient evaluation of the integral and optical depth $D(x)$. We store the optical depth $D(x)$ in a voxel structure for each directional light. This optical density map can be produced by simple alpha-blending, and is referred to during the evaluation of integral. The integration can be performed efficiently by using integral tables because the plane-parallel solution is the sum of a set of simple exponential functions.

3.2 Integral Tables

The calculation involves integration with direction s and length l. We prepare two types of tables; one evaluates only the angular integral and the other evaluates both integrals.

Let us fix the incident light distribution. First, the plane-parallel solution I_{pp} is approximated by the sum of a small number, n_e, of eigen vectors as:

$$I_{pp}(x;N) \simeq \sum_j^{n_e} c_j \exp(\lambda_j D(x)) v_j, \tag{14}$$

where N denotes the vector normal to the surface (**Fig. 2**). We observed that the change in I_{pp} with respect to N is small unless the incident angle is near-tangential, so we neglect the dependence on N in the current implementation.

Then, the angular integral, I_s, can be evaluated by

$$\begin{aligned} I_s &= \sigma_s \int p(s,s') I_{pp}(x',s') ds' \\ &= \sigma_s \sum_j^{n_e} c_j \exp(\lambda_j D(x')) \left[\int p(s,s') v_j(s') ds' \right] \\ &= \sigma_s \sum_j^{n_e} c_j [\exp(\lambda_j D) P_{Vj}(s)] \\ &= \sigma_s R(D;s), \end{aligned} \tag{15}$$

where

$$P_{Vj}(s) = \int p(s,s') v_j(s') ds'.$$

Note that it only depends on the cosine between N and s if the phase function p is isotropic. Referring to this two-dimensional table $R(D;s)$, the integral can be summed up by

$$I = \sum_j \exp(-d_j)\sigma_s R(D_j;s) + I_0.$$

It is also possible to take account of the Fresnel reflection and refraction on the boundary surface by multiplying the Fresnel factors, as

$$I = F_t(x_0) \sum_j F_t(p_{in}) \exp(-d_j)\sigma_s R(D_j;s) + I_0,$$

where $F_t(p_{in})$ represents the Fresnel factor at the light incident point, p_{in}, and d_j and l_j are the optical depth and the length along the viewing ray.

When local linearity of the optical depth D,

$D(x') = al' + D_1$,

$x' = x - l's$

can be assumed, it is possible to analytically integrate the line integral by

$$\begin{aligned} &\int_0^l \exp(-d(l'))\sigma_s \left(\int_\Omega p(s,s') I_{pp}(x',s') ds' \right) dl' \\ &= \sum_j c_j P_{Vj} \int_0^l \exp(-\sigma_t l) \exp(\lambda_j a\, l') dl' \\ &= \sum_j c_j P_{Vj} (1/(-\sigma_t + \alpha\lambda_i)) \cdot (\exp(-\sigma_t l + \lambda_j D_2) - \exp(\lambda_j D_1)) \\ &= \exp\{(-\sigma_t l/a)\} S(D_2) - S(D_1), \\ &D_2 = D_1 + la \end{aligned} \tag{16}$$

where

$$S(D;a,s)=\sum_{j}^{n_e} c_j P_{Vj}(1/(-\sigma_t+a\lambda_i))\exp(\lambda_j D). \quad (17)$$

Next, let us consider the situation that the integral interval is subdivided into several intervals in which the local linearity can be assumed. In this case, the overall integral can be efficiently evaluated by using this three-dimensional table S, as

$$I = I_0 + F_t(x_0)\sum_j [F_t(p_{in})\exp(-\sigma_t l_j) \times \{\exp\{(\sigma_t/a)(d_j-d_{j+1})\}S(D_{j+1})-S(D_j)\}].$$

Although the evaluation with the table S involves exponential calculations, this provides more smooth images with a lower number of samples than the function table R. We adopted the table S in the CPU implementation and the R in the GPU implementation.

3.3 Albedo Map

Textures are very important in a realistic image synthesis. Since the color is closely related to the albedo, we assume that textures are given as albedo maps, as in previous work [4]. In our implementation, albedo maps are prepared as 3D textures or projection textures. Let us consider the situation where the albedo α changes in space, as shown in Fig. 2. As discussed in Ref. 8), the intensity at x_j can be approximated by the plane-parallel solution with the average albedo $\bar{\alpha}$ as:

$$I_{alb}(D_j,s) \simeq I_{pp}(D_j,s;\bar{\alpha}(x_j)),$$
$$\bar{\alpha}(x_j)=\int_{p_i n}^{x_j}\alpha(x)dl/|p_{in}-x_j|$$

Similar to the case of the optical density map, the average albedo can be pre-calculated and saved in voxels. Using I_{alb} instead of I_{pp} takes the variation in albedo into account in the integral. A straightforward solution for the efficient integration is to prepare integration tables $S(\alpha)$ or $R(\alpha)$ for several albedo values and to interpolate them.

3.4 Algorithm

The algorithm consists of a rendering process and a pre-process stage. The rendering process performs the integration of Eq. (13). It first determines sample points x_i along the viewing ray, then obtains scattering properties such as the optical depth $D(x_i)$, $d(x_j)$, and the average albedo $\bar{\alpha}(x_i)$ at each sample point. The process finally evaluates the integral by using the integral table, R or S. This is carried out for every viewing ray, i.e., pixel or vertex.

Pre-process:

1 calculate matrices related to the plane-parallel solution.
2 3D-scanconversion to make the density map .
3 alpha-blending to make the optical density map D and the average albedo ¯.
4 build integral table, S or R.
5 draw objects and read back to furthest point buffer, Fp.

Rendering process:

1 determine sample points using Fp
2 for each sample, calculate the integral using The maps and the table S or R.
3 sum up the sampled integral.

Fig. 3 Algorithm.

The pre-process prepares the matrices, maps and tables that are referred to in the rendering process, namely:

(1) matrices related to the plane-parallel solution, such as V and V^{-1},
(2) the optical depth $D(x_i)$,
(3) the average albedo $\bar{\alpha}(x_i)$,
(4) the integral table R or S,
(5) the furthest point buffer F_p.

The furthest point buffer, $F_p(x,y)$ stores the object points that are furthest away when viewed by the eye, and is referred to when the integral is evaluated in the rendering process. Note that the Fresnel factor and directional shadows can be taken into account by the optical density map by adding the logarithm of the values.

F_p, R and S are updated every frame, and $D(x_i)$ and $\bar{\alpha}(x_i)$ are re-calculated whenever the incident lights or the shapes of objects change.

An outline of the algorithm is listed in **Fig. 3**. It is straight-forward to imple-

ment the rendering process both on CPU and GPU.

4. Experiments

4.1 Comparisons

We first conducted experiments to estimate the accuracy of the proposed method under various controlled conditions.

Figure 4 shows the situation simulated in the experiments. The object shape is a rectangular tube with dimensions 10 mfp × 3 mfp in x and y and an infinite length in the z direction, where mfp denotes the mean free path, $1/\sigma_t$. The incident light illuminates the upper surface S_1. The optical index of the material was set to 1.0. The light intensity was calculated by the proposed method, the multipole method (MP) with/without the single scattering component, and by a Monte Carlo method (MC), as a reference. For the phase function p, Henyey-Greenstein's function

$$p(s, s') = (1/4\pi)(1 - g^2)/(1 - 2g(s \cdot s') + g^2)^{3/2},$$

was adopted. 128 points are sampled on S_1 and S_2, and 36 points on S_3 and S_4, at which the intensity distribution was calculated.

4.1.1 Uniform Illumination

In the first experiment, a uniform directional light is used for the illumination. The intensity distribution at $x = 0$ on S_1 and S_2 is shown in **Fig. 5**, where the green line indicates the incident light direction. The root-mean-square (RMS) difference compared with the results from the Monte Carlo method is also presented in **Table 2**. As expected, the proposed method matches well with the MC method.

Although the reflectance of the multipole model is in good agreement with that of the MC method with the uniform phase function ($g = 0$) and perpen-

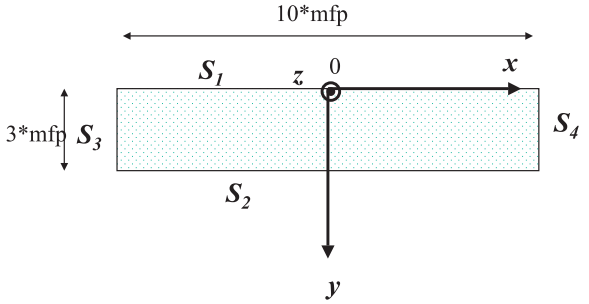


Fig. 4 Cross section of the rectangular tube used in the experiments. The length along the z-direction is infinite.

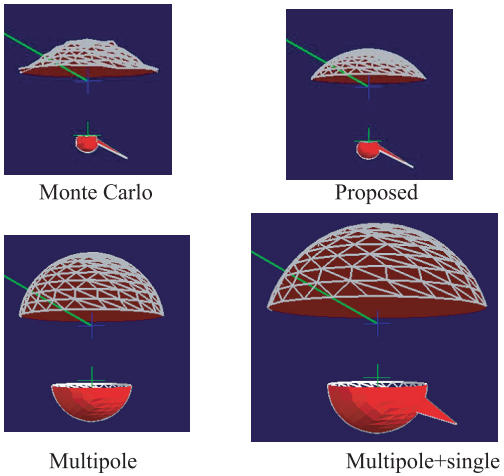


Fig. 5 Reflectance and transmittance distribution calculated by the Monte Carlo method, the proposed method, the multipole method without/with the single scattering term. The green line indicates the incident light direction. $g = 0$, $\alpha = 0.95$.

Table 2 RMS differences. R/T indicates reflectance/transmittance. S and MP+S means the results from the single scattering model and the multipole model with single scattering.

light angle	g	R/T	Proposed	MP	MP +S	S
0°	0	R	13%	13%	23%	65%
		T	4	72	24	56
60°	0	R	10	43	101	57
		T	8	210	302	89
0°	−0.5	R	20	23	51	48
		T	5	84	189	47
0°	0.5	R	24	19	6	81
		T	7	58	26	66

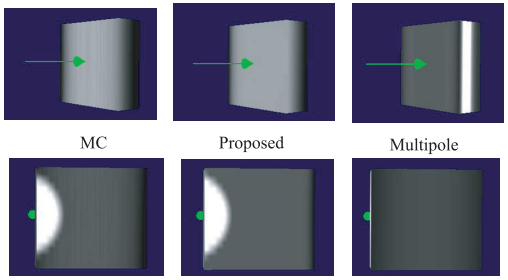


Fig. 6 Shaded images based on calculated intensity, where the infinite tube is clipped out in the z-direction. The incident light is perpendicular to S_1, as indicated by the green arrows. In the top row, surfaces S_1 and S_3 are shown, and surface S_2 is shown in the lower row. $g = 0$, $\alpha = 0.95$.

dicular light, the difference is significant in other cases, especially for transmittance. Adding the single scattering term presents a better shape, but it reflects too much energy, violating the energy preservation law. The conventional ray-marching method only captures the single scattering contribution, which considerably under-estimated the intensity and produced large errors, as shown in the table.

The rectangular object was shaded based on the intensity distribution calculated at the sample points, **Fig. 6** shows the shaded images, where the infinite tube has been clipped out for easier observation. The side surfaces, S_3 and S_4, are perpendicular to S_1 and it is very difficult for layered material models to deal with them. We found serious artifacts produced by the multipole model; the multipole model locates one of the point sources below the surface S_1, and this causes a 'bright belt' on the side surfaces. This artifact might be reduced by introducing some heuristics based on the surface geometry, but it is really troublesome when the width of the material is close to $1/\sigma_t'$. The proposed method also overestimates the intensity, but provides much better results. As for the lower surface S_2, the multipole model failed to reproduce the directional light transmittance on S_2, which makes the RMS very large.

4.1.2 Density, Albedo and Shadow

Next, we changed σ_t stepwise at $x = 0$ from 1 to 1/2, emulating the density variation. The shaded images are shown in **Fig. 7**. The proposed method agrees

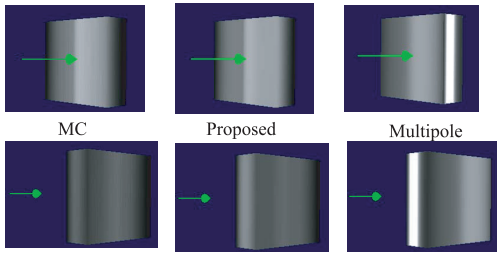


Fig. 7 Shaded images when a stepwise change in density was applied at $x = 0$. $g = 0$, $\alpha = 0.95$.

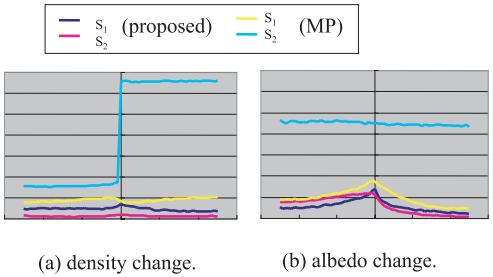


Fig. 8 RMS difference plotted against x. At $x = 0$, (a) the density, (b) the albedo, and (c) illumination, changes stepwise. $g = 0$, $\alpha = 0.95$.

well with the MC method. On the other hand, the low density area S_2 appears to be too bright in the result from the multipole model. The bright area on S_2 was caused by the same artifact as we observed in the side surfaces in Fig. 6. The step pattern on S_1 was also much too blurred by the multipole model. Similar over-blurring was also observed in the albedo and shadow experiments.

In **Fig. 8** (a), the RMS difference was plotted against x. As shown in the figure, the proposed method provides a good approximation where $x > 2$ mfp.

In a similar way, we examined the influence of the spatial variation in the albedo and the illumination. Figure 8 (b) shows the RMS difference measured in a simulation in which the albedo changes stepwise at $x = 0$ from 0.95 to 0.95×0.5. In Fig. 8 (c), the surface S_1 was only illuminated where $x < 0$ and the other half was completely in the shadow. As shown in the figures, the proposed method

provides a good approximation where $x > 2$ mfp.

4.2 Image Synthesis

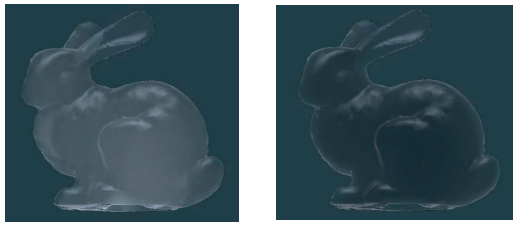
We rendered several translucent objects using the proposed method with both the CPU and GPU implementation. In the CPU implementation, the intensity is evaluated at each vertex and images are generated by Gouraud shading. We also implemented the rendering process on GPU by GLSL, where the intensity is calculated pixel by pixel. **Figure 9** (a) shows a bunny made from a homogeneous scattering material ($g = 0$, $\alpha = 0.95$). As a reference, an image was rendered with only a single scattering term, which appears to be too dark because it neglects multiple scattering. The objects consist of 15 K polygons in total. We rendered the image on an AMD Athlon 64 (2 GHz) with an NVIDIA GeForce 8800GTX, and the rendering time was 0.17 seconds per frame on the CPU. The preprocessing time was 1.3 sec for the eigenvector calculation, 1.3 seconds for the 3D scan-conversion to build the density map, and 0.16 seconds to create the optical depth map. The 334 directions were sampled for s_i on the hemisphere (Eq. (7)).

Figure 9 (b) shows a piece of chestnut Yokan (a sweet jelly made from red beans) on a diffusive glass plate. The chestnut and Yokan have different albedo and density values and are composed into a single density and average albedo map. The used albedo rgb values are (0.16, 0.11, 0.07) for the Yokan and (0.51, 0.42, 0.16) for the chestnut. The rendering time was 0.21 sec per frame on CPU and 0.017 sec on GPU.

Figure 9 (c) shows pieces of sushi, where albedo mapping was applied to raw fish and rice. Volumetric scattering in tuna was successfully simulated, and images of sushi with a tasty appearance were generated. Three types of tuna, *akami* (low fat tuna), *chu-toro* (high fat tuna), *oh-toro* (extremely high fat tuna), are displayed in the figure, and the differences between them were effectively enhanced by the albedo textures. Albedo maps were applied to the rice and fish using projection textures. The albedo value α in the albedo map was calculeted from the texture value r by regarding this as the reflectance. According to the Kubelka-Munk reflectance formula [8], α is calculeted by

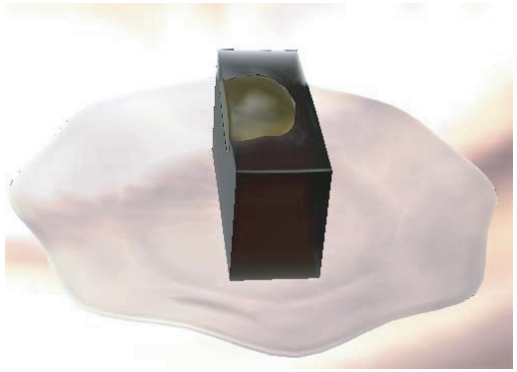
$$\alpha = 2R/(R^2 + 1).$$

The rendering time was 1.8 sec per frame on CPU and 0.12 sec on GPU.

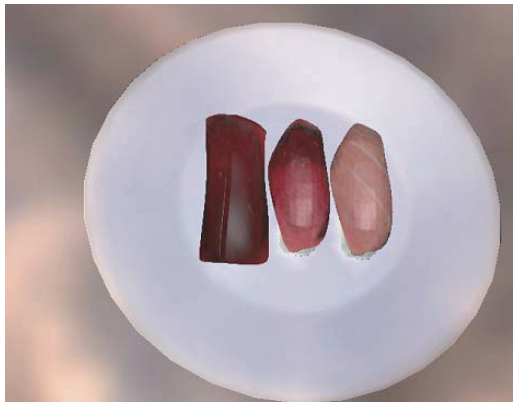


proposed method single scattering

(a) Translucent bunny.



(b) A piece of chestnut Yokan.



(c) Sushi.

Fig. 9 Examples of food rendering.

4.3 Discussion

As shown in the controlled experiments, the proposed method exhibited far better performance than the multipole model. Although the multipole model has been proven to be capable of creating realistic images, serious problems were observed when optically-thin objects were concerned.

The proposed method, on the other hand, provided reasonable approximations at locations that are separated from boundaries by a few mean free paths. The proposed method is essentially based on a voxel structure and is very flexible to shape deformation and topological changes. Unlike mesh-based filtering, a simple volume reconstruction adapts to changes, without the necessity of re-classifying meshes and/or re-meshing. The scattering properties and the light direction can also be edited interactively without any particular algorithmic modification.

Albedo mapping was incorporated into the proposed method in a natural way that simply integrates the plane-parallel solution according to the average albedo values. As demonstrated in the images, the proposed method successfully reproduced realistic textures in multiple scattering environments. In the multipole model, on the other hand, the mapping is achieved in a heuristic way that causes scattering effects to be normalized into gray scale.

5. Conclusion

We proposed a novel rendering method that couples the plane-parallel solution with ray-marching. The method captures both multiple scattering features and volumetric appearance. In the experiments, the proposed method demonstrated far better performance than previous methods in terms of accuracy and flexibility. The method was successfully applied to the rendering of food. Interactive speed was achieved by a CPU implementation for simple objects, and the GPU implementation further improved the display rate more than 10 times faster than the CPU implementation.

The proposed method involves several sampling processes. We observed aliasing artifacts, and the future work includes its efficient anti-aliasing. We are also interested in applying the method to participating media such as clouds, by which we could establish a unified practical algorithm that is capable of dealing with all scattering media. It is also of interest to apply the plane-parallel solution to texture-based approaches such as the shell texture [9] because the mode functions may represent the light field in a compact manner.

References

1) Dorsey, J., Edelman, A., Jensen, H.W., Legakis, J. and Pedersen, H.: Modeling and rendering of weathered stone, *Proc. SIGGRAPH'99*, pp.225–234 (1999).
2) Pharr, M. and Hanrahan, P.: Monte Carlo evaluation of non-linear scattering equations for subsurface reflection, *Proc. SIGGRAPH 2000*, pp.75–84 (2000).
3) Jensen, H.W., Marschner, S.R., Levoy, M. and Hanrahan, P.: A practical model for subsurface light transport, *SIGGRAPH 2001*, pp.511–518 (2001).
4) Donner, C. and Jensen, H.W.: Light diffusion in multi-layered translucent materials, *ACM Trans. Gr.*, Vol.24, No.3, pp.1032–1039 (2005).
5) Ishimaru, A.: *Wave propergation and scattering in random media*, Vol.1, Academic Press, New York (1978).
6) Stam, J.: An illumination model for a skin layer bounded by rough surfaces, *Proc. 12th Eurographics Workshop on Rendering*, pp.39–52 (2001).
7) Wang, L., Wang, W., Dorsey, J., Yang, X., Guo, B. and Shum, H.-Y.: Real-time rendering of plant leaves, *ACM Trans. Gr.*, Vol.24, No.3, pp.712–719 (2005).
8) Shinya, M., Shiraishi, M., Dobashi, Y., Iwasaki, K. and Nishita, T.: Fast Display of Sub-surface Scattering using Eigen Solutions, VC2008, 08-01 (2008) (In Japanese).
9) Chen, Y., Tong, X., Wang, J., Lin, S., Guo, B. and Shum, H.-Y.: Shell texture functions, *ACM Trans. Gr.*, Vol.23, No.3, pp.343–352 (2004).

(Received December 1, 2008)
(Accepted March 6, 2009)
(Released June 10, 2009)

Mikio Shinya is currently a Professor at Department of Information Science, Toho University. He received a B.Sc. in 1979, an M.S. in 1981, and a Ph.D. in 1990 from Waseda University. He joined NTT Laboratories in 1981, and moved to Toho University in 2001. He was a visiting scientist at the University of Toronto in 1988–1989. His research interests include computer graphics and visual science.

Michio Shiraishi was born in 1974. He received his Ph.D. degree from The University of Tokyo in 2003. He has been a lecturer of Toho University since 2005. His current research interests are computer graphics and web application systems.

Yoshinori Dobashi received the B.E., M.E., and Ph.D. in Engineering in 1992, 1994, and 1997, respectively, from Hiroshima University. He worked at Hiroshima City University from 1997 to 2000 as a research associate. He is presently an associated professor at Hokkaido University in the graduate school of engineering. His research interests are computer graphics including lighting models.

Kei Iwasaki received the B.S., M.S., and Ph.D. degrees from The University of Tokyo in 1999, 2001, 2004, respectively. He is presently a research associate at the Faculty of Systems Engineering, Wakayama University. His research interests are mainly for computer graphics.

Tomoyuki Nishita received the B.E., M.E., and Ph.D. degrees from Electrical Engineering from the Hiroshima University, Japan, in 1971, 1973, and 1985, respectively. He worked for Mazda Motor Corporation from 1973 to 1979. He has been a lecturer at the Fukuyama University since 1979, then became an associate professor in 1984, and later became a professor in 1990. He moved to the Department of Information Science of the University of Tokyo as a professor in 1998 and now is a professor at the Department of Complexity Science and Engineering of the University of Tokyo since 1999. His research interests are mainly for computer graphics.