A Computational Approach to Simulate Subsurface Light Diffusion in Arbitrarily Shaped Objects

Tom Haber Tom Mertens Philippe Bekaert Frank Van Reeth

Expertise Centre for Digital Media
Limburgs Universitair Centrum
Universitaire Campus, 3950 Diepenbeek, Belgium
firstname.lastname@luc.ac.be

Abstract

To faithfully display objects consisting of translucent materials such as milk, fruit, wax and marble, one needs to take into account subsurface scattering of light. Accurate renderings require expensive simulation of light transport. Alternatively, the widely-used fast dipole approximation [15] cannot deal with internal visibility issues, and has limited applicability (only homogeneous materials).

We present a novel algorithm to plausibly reproduce subsurface scattering based on the diffusion approximation. This yields a relatively simple partial differential equation, which we propose to solve numerically using the multigrid method. The main difficulty in this approach consists of accurately representing interactions near the object's surface, for which we employ the embedded boundary discretization [5, 16]. Also, our method allows us to refine the simulation hierarchically where needed in order to optimize performance and memory usage. The resulting approach is capable of rapidly and accurately computing subsurface scattering in polygonal meshes for both homogeneous and heterogeneous materials. The amount of time spent computing subsurface scattering in a complex object is generally a few minutes.

Key words: Rendering, subsurface scattering, heterogeneous materials, embedded boundary, multigrid

1 Introduction

In our daily life, we are surrounded by many translucent objects, such as milk, marble, wax, skin, paper, and so on. The translucency is caused by light entering the material, scattering inside it, and eventually exiting at a seemingly arbitrary location. The diffusive character of this phenomenon yields a distinct appearance, typically smoothing and concealing small surface detail. Furthermore, light may scatter through an object, which lights up thin geometric detail if illuminated from behind. Also, chromatic changes may appear due to wavelength dependent scattering. Traditional reflection models cannot represent such effects since they assume light does not enter the material; it is only scattered or reflected directly at the surface.

Subsurface scattering is a difficult global illumination problem, which aims at solving the (volume) rendering equation for participating media. Direct numerical solutions have proven to be accurate, but slow.

A breakthrough was achieved with the popular dipole model [15] based on the diffusion approximation. Although it generates plausible results at a fraction of the computational cost of numerical methods, it is inherently limited to homogeneous media (material properties do not vary spatially) and lacks accuracy for arbitrarily shaped objects, due to the semi-infinite plane approximation. We wish to deal with these problems.

We restrict ourselves to optically thick (or highly scattering) materials, where multiple scattering (cfr. interreflection in global illumination) is dominant. Under this condition it makes sense to apply the diffusion approximation [28, 11]. This significantly simplifies the volume rendering equation to the so-called diffusion equation (see appendix A). For moderately scattering materials, light transport can be approximated efficiently by the sum of a single scattering and multiple scattering term [15], thus our diffusion solution can be easily extended to also treat this case.

The diffusion equation is essentially a linear partial differential equation (PDE) with mixed boundary conditions. We show that it can be solved using a multigrid-based approach in arbitrarily shaped domains. Stam [28] was the first to suggest the multigrid method for solving the diffusion PDE. However, his paper only discusses an illustrative case in 2 dimensions for a simple homogeneous square slab. This paper can be seen as an extension of this work, in order to render the multigrid method fit for practical use.

To summarize, the contributions described in this paper are:

- We propose to use the embedded boundary dis-

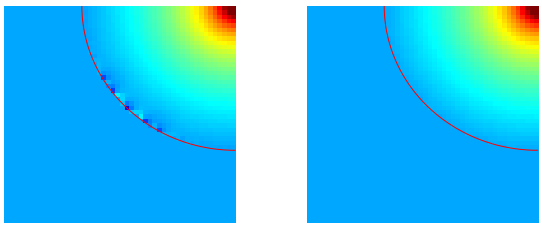


Figure 1: Solutions of the diffusion equation in 2D on a 100×100 grid, for the geometry of a circle (outlined in red). The source term is a 2×2 block at the midpoint of the circle. Only the lower left side is shown for illustration purposes. ***Left:*** *Discretization using finite differences yields instabilities near the boundary. Similar artifacts occur in the upper right part.* ***Right:*** *Discretization using the embedded boundary results in a consistent solution.*

cretization (EBD) [5, 16] for accurately representing the object's complex surface. It is essential to retain numerical stability for the multigrid algorithm and to obtain an accurate solution near the locations of interest for subsurface scattering.

- To increase performance and to keep memory requirements feasible, we describe an extension for hierarchical refinement [1, 6, 12] of the solution where needed. This is highly recommended, as volumetric representations tend to be large.

2 Related Work

Simulating subsurface light transport is a special case of global illumination in participating media, also referred to as the volume rendering problem. Here, one tries to solve the general volume rendering equation, for which several solutions have been proposed, including finite element methods [27], (bidirectional) path tracing [9, 17], metropolis light transport [21], photon mapping [13, 7], and scattering equations [22]. In recent work, Premože et al. [24] presented a mathematical framework for path integration, which led to an efficient algorithm [25] capable of accurately rendering multiple scattering effects. Although volume rendering techniques can be applied to the problem of subsurface scattering, they are generally too expensive to generate images within a reasonable amount of time.

The diffusion equation is a sound approximation to the general volume rendering equation in the case of highly scattering media, which significantly reduces the complexity of the problem. Stam [28] was the first to introduce the diffusion approximation in the context of rendering. He applied a finite element method and the multigrid method to solve the diffusion equation. This was demonstrated with simple, illustrative examples. As we will show in this paper, readily applying multigrid is not sufficient, as numerical and efficiency problems may arise when applying it to arbitrary objects.

Jensen et al. [15] introduced the dipole BSSRDF model, which does not require a numerical solution of the volume rendering or diffusion equation. Instead, an approximate analytical solution to the diffusion equation for a semi-infinite homogeneous slab is used directly on arbitrary 3D objects. This crude approximation yields highly plausible renderings and served as the basis for an efficient hierarchical technique [14], and several interactive methods [18, 4, 20, 19, 2, 10]. Although the practical use of this model is clear, it is far from accurate (see figure 5) and is inherently limited to homogeneous materials. Also, it cannot deal with internal visibility caused by voids and concavities.

Gösele et al. [8] and Chen et al. [3] presented data-driven methods capable of rendering heterogeneous translucent objects. Our method is also capable of rendering heterogeneities, and will not require real world data [8] or many hours of precomputation [3].

3 Outline

In this section, we outline a hierarchical approach to solve the diffusion approximation efficiently and accurately. As will be detailed in the following sections, we discretize the diffusion PDE using the EBD, in order to translate it to a linear system of equations. We will briefly review the EBD method as described by Johansen et al. [16] and Day et al. [5], and explain how it is applied to our problem.

3.1 Discretization of the Diffusion Equation

Before going into the mathematical background, we refer the reader to the appendix A for a brief review of the diffusion equation. It can be skipped safely — we will formulate the diffusion equation (1) and its boundary conditions (5) shortly, although in a slightly modified form.

The EBD builds on an approximation of the divergence found in the diffusion equation. Therefore, let us define an auxiliary vector function $\vec{F} = \frac{1}{3\sigma_{tr}}\vec{\nabla}U_d$. The diffusion equation now becomes:

$$\vec{\nabla} \cdot \vec{F} - 4\pi\sigma_a U_d = S \tag{1}$$

The solution of this PDE will be represented on a 3D uniform cartesian grid, in which its function variables are discretized at the center of each cubical cell. However, our domain boundary, the object's surface, does not necessarily align with the boundaries of the cells, so care must be taken to consistently represent the information

in these cells. We will refer to cells that intersect with the surface as "boundary cells". The part of a boundary cell that is split by the surface is dubbed "cut cell". Non-boundary cells are either "full" or "empty".

Let's call $\hat{U}_d$ the discretization of the unknown function U_d (fluence). The differential operator in equation 1 can be represented in the grid by a weighted summation over its values $\hat{U}_d$, for instance, using finite differences [26]. One calls the arrangement of the weights "stencil", and is akin to the kernel of a convolution filter. The number of terms is very low; typically, only the values in the 4-connected neighborhood are taken into account, and can thus be written as a sparse linear system of equations:

$$A\hat{U}_d = \hat{S} \tag{2}$$

Here, $\hat{S}$ is the discretized version of the source term in equation 1, and A contains the stencil. Note that in equation 2, all the $\hat{U}_d$ and $\hat{S}$ values from the 3D grid have been enumerated as vector.

We will now describe the EBD in detail. Consider $\vec{\nabla}\cdot\vec{F}$ in a single full cell C of width h and approximate this term by averaging it over the cell. Using the divergence theorem, we can relate the resulting volume intergral to a surface integral:

$$\vec{\nabla}\cdot\vec{F} \approx \frac{1}{h^3}\int_C \vec{\nabla}\cdot\vec{F}dV = \frac{1}{h^3}\oint_{\partial C}\vec{F}\cdot\vec{n}dA$$

Applying the midpoint integration rule for each cell face yields:

$$\vec{\nabla}\cdot\vec{F} \approx \frac{1}{h}\sum_{facef}\vec{n}_f\cdot\vec{F}(x_f) \tag{3}$$

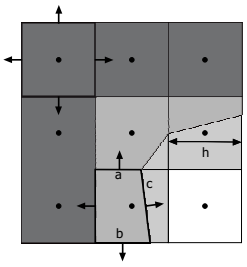
where x_f represents the center of face f. Equation 3 can be translated to a stencil using central differences. Note that the same result can be achieved by direct finite differencing of $\vec{\nabla}\cdot\vec{F}$.

The strength of the EBD is the way boundary cells are treated. A first order approximation of the surface is introduced in order to generalize the integration scheme in equation 3:

$$\vec{\nabla}\cdot\vec{F} \approx \frac{\sum_{facef}\alpha_f\vec{n}_f\cdot\vec{F}(x_f) + \alpha_B\vec{n}_B\cdot\vec{F}(x_B)}{\kappa h} \tag{4}$$

Here x_B is the center of the boundary surface and n_i^B the normal at this location. Let us clarify the ratio variables (see figure 2):

- κ: volume ratio of cut cell to full cell;
- α_f: ratio of cut face area to full face area;
- α_B: ratio of the (first order approximation) boundary area to face area.



	full	boundary
$\alpha_{f=l}$	1	1
$\alpha_{f=r}$	1	0
$\alpha_{f=t}$	1	$\frac{a}{h}$
$\alpha_{f=b}$	1	$\frac{b}{h}$
α_B	0	$\frac{c}{h}$

Figure 2: ***Left:*** *Geometry of cells in the 2D case for clarity (3D case is analogous). Dark, light and white cells represent full, boundary and empty cells, respectively. The domain's geometry is indicated by the darkening overlay. The arrows indicate normals at the face center x_f used to evaluate the midpoint integration scheme (see equation 3).* ***Right:*** *Face weights used in the discretization of the diffusion PDE into cubic cells of width h (see equation 4). Face indices are written as (l)eft, (r)ight, (t)op and (b)ottom.*

Again using central differences, one can construct a single stencil that can be used on both full and boundary cells. Note that for the case of ordinary finite differences, a different and less accurate stencil needs to be employed in boundary cells. Figure 1 illustrates the problems that might occur with the use of finite differencing [26]. These problems occur because finite differencing requires that any boundaries present in the domain are aligned with the cell faces. "Irregular" boundaries need to be approximated, leading to inconsistencies. The consequences of such inconsistencies are possible failure of the multigrid algorithm (no convergence) and an inaccurate solution of the radiance where it is of utmost importance for subsurface scattering. The EBD solves these problems.

The EBD method allows us to naturally incorporate the necessary boundary conditions. Day et al. [5] mention that homogeneous Neumann conditions can be enforced by setting the flux vector at the boundary to zero (i.e. by cancelling the second term in equation 4). In our case we have mixed Neumann and Dirichlet conditions (see A):

$$\vec{n}\cdot\vec{F} = \frac{1}{2A}U_d - \frac{1}{3\sigma_{tr}}\vec{n}\cdot\vec{Q}_1 \tag{5}$$

In the same spirit, our condition can be easily enforced by substituting $\vec{n_B}\cdot\vec{F}(x_B)$ in the second term of equation 4, by equation 5. As a result, this stencil can now be applied to both full and boundary cells.

Technically, equation 4 is only valid at the center of mass of the cut cells. We use the data-centering scheme by Johansen and Colella [16] to solve this problem; it essentially moves the cell data to the geometric center of the full cell.

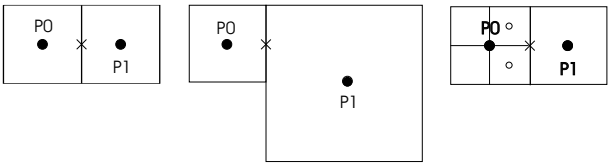


Figure 3: Three cell arrangements that can occur when computing the gradient on a cell face: (left) cells at the same level, (middle) fine-coarse boundary and (right) coarse-fine boundary.

3.2 Multigrid Method

Using the EBD, we apply the multigrid method [29] to rapidly solve the linear system in equation 2. Multigrid solves the system on hierarchical grids, which communicate via a set of operators. First, an approximate solution is computed by removing high-frequencies in the error, a process known as "smoothing". The residual $r = A\hat{U}_d - \hat{S}$ is then projected onto the next coarser grid and a correction is obtained by solving the equation with the restricted residual as right-handed side. These steps are repeated until the coarsest level is reached. Afterward the direction is reversed and the corrections at each level are interpolated to the next finer grid. The whole process is called a "V-cycle" and is iterated until convergence is reached (determined by $\|r\| < \epsilon$).

In our implementation, smoothing is carried out by Gauss-Seidel relaxation [26]. This is a fairly common technique, yielding adequate performance on most problems. The residual is projected by volume-weighted averaging. This seems to be the best choice considering the irregular nature of the boundary cells. For interpolation of the correction, we adopted a straightforward operator which simply replicates the value to the 4 child cells.

3.3 Adaptive Refinement

As a result of the linear approximation of the embedded boundary, high curvature regions may not be faithfully represented in the grid. Also, sudden changes in illumination and material properties require the appropriate amount of detail. Naively increasing resolution solves this problem, albeit at a substantial performance and memory cost. We alleviate these problems by only refining the grid when necessary.

As refinement criteria, we observe the divergence of the source term S and the effective transport coefficient σ_{tr}, to account for changes in illumination and material properties, respectively. The refinement stops when these values drop below a specified threshold.

When neighbouring cells are not the same size, the gradient at the face can no longer be computed using central differencing. A more involved scheme must be applied across these coarse-fine and fine-coarse boundaries (see figure 3). A three-point interpolation scheme [23] is used to compute these gradients with second-order accuracy.

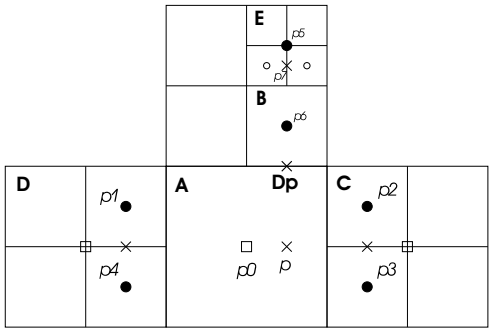


Figure 4: Second-order interpolation scheme used for face gradient (Dp) calculation at a fine-coarse cell boundary.

Consider the point Dp in figure 4 at a fine-coarse boundary. The gradient at this point is computed by fitting a parabola through points p, $p6$ and either $p5$ or $p7$. If E is a leaf cell, the gradient can be expressed as

$$hDp = -\frac{U_d(p6)}{3} - \frac{U_d(p5)}{5} + \frac{8U_d(p)}{15}$$

If E is not a leaf, the value of U_d at p7 needs to be interpolated from the closest children (indicated by a $\circ$ in the figure). The value at p in turn is evaluated in the same way using the values from cells C and D. Due to refinement constraints, these two cells are at the same level as A.

The coarse-fine gradient is constructed as minus the average of gradients constructed from the children of the fine cell. Those gradients are all at a fine-coarse boundary and can be computed using the scheme defined above. This approach results in consistent gradients.

The use of this three-point scheme implies that the levels of all neighboring cells (direct or diagonal) cannot differ by more than one level and all cells neighbouring a cut-cell must all be at the same level. The first constraint has little impact on the refinement operation but the second is more restrictive as it forces all cells cut by the boundary to be at the same level.

4 Results

We implemented our algorithm in C++ on a Pentium 4 1.7Ghz 512 MB RAM configuration.

In Table 1 we illustrate the performance of our application with different models. The memory requirements include both the mesh and the octree. The preprocessing time is the time needed to load the mesh and build the octree. For large meshes, this step can be quite time

consuming. The "source" column has the amount of time required to compute the source term (right hand side of equation 1). This involves casting shadow rays to the light source from the center of each cell and computing the attenuation along these rays.

Table 2 illustrates the performance with different materials. We see that as the object becomes more opaque, the rendering time decreases. This is contrary to the behavior of the fast hierarchical dipole solution [14], where the number of required samples increases dramatically w.r.t. opaqueness. This increase is due to the behavior of the Green's function used in the dipole model. It peaks strongly for more opaque materials, and therefore requires many samples to guarantee a smooth reconstruction. Also, in the case of our diffusion simulation, light has to travel over a smaller distance, which reduces the number required of iterations. All timings were gathered on the dragon model under the same lighting conditions. The last four rows represent heterogeneous materials, constructed by interpolating between two different materials with Perlin noise (Figure 6.1 shows a screenshots of one such rendering).

All images in figure 6 were rendered using OpenGL after interpolating the colors at the vertices from the grid solution. In figures 6.1-2-3, the venus, dragon and buddha models are rendered with a mixture of marble with a ratio of minimum and maximum density of 1:10. The scales of the objects are 10cm, 2cm and 4cm respectively. Note that, with "scale" we refer to the length of the longest side of the object's bounding box. In figure 6.4 a marble dragon at a scale of 2cm is backlit. Figures 6.5-6-7 shows a sculpture with skim milk and whole milk. The last image is backlit: the scattering of light is very obvious for the thin geometric features. Also, notice the chromatic shifts and shadowing. In figures 6.8-9-10 the marble dragon is rendered using varying anisotropy g: from left to right, 0.0, 0.5 and 0.9. For small values of g (more isotropic), interactions are localized near the surface, while for higher values light penetrates deeper inside, yielding a more translucent effect.

model	#tris	mem	pre	src	mgrid	tot
dragon	200K	38.3	16.1	5.0	29.8	50.9
buddha	800K	61.0	72.8	8.2	16.0	97
venus	31K	32.4	3.1	1.8	83.1	88

Table 1: Overview of performance with different models. Timings are in seconds and memory usage is in MB. "#tris", "mem", "pre", "src", "mgrid" and "tot" refer to number of triangles, memory usage, preprocess, source term computation, multigrid simulation and total render time, respectively. Material was marble scaled at 6mm.

material	scale	simulation
marble	5mm	444.69
marble	10mm	295.9
marble	20mm	214.97
skim milk	6mm	605.95
milk mix	1cm	105.15
milk mix	2cm	62.57
marble mix	10cm	85.55
marble mix	2cm	205.55

Table 2: Overview of performance with different materials and scales. Timings are in seconds. We used the dragon model for all results except for the last three where we used the venus model.

Figure 5 shows a comparison of the fast dipole approximation (FDA) [15] and our approach. Several visual differences are noticeable. Most importantly, our solution seems to be more dependent on the geometry, causing strong varieties in brightness. The same variations are seen in the reference solution. The dipole solution however, remains smooth regardless of the geometry. One of the causes is the complex internal visibility in this model, which the FDA cannot take into account due to the semi-infinite slab approximation. Note that the FDA rendering is more accurate in terms of chromatic shifting. The brownish tint in our solution can be explained by the attenuation of incoming radiance by $e^{-\sigma_t d}$ (with d the traveled distance inside the object). This function is evaluated at the midpoint of each cell, hence $d > 0$, causing a "chromatic bias". This problem can be alleviated easily by further refinement (see next section). Note that both the FDA and our method lack high frequency shading details caused by low order scattering, esp

5 Discussion and Conclusion

We have presented a flexible method for the simulation of subsurface scattering based on the diffusion approximation. In particular, we showed that a combination of an octree discretisation, a multigrid solver and an embedded boundary stencil proves to be a feasible and efficient technique for the numerical solution of the diffusion equation in complex objects within a reasonable amount of time (typically a few minutes). The approach differs from the previous attempt with multigrid [28] by employing a better representation of the boundaries, and a simple refinement strategy. Compared to the popular dipole approximation [15], our method is able to resolve visibility correctly, and can deal with heterogeneous materials.

Cell subdivision may reach practical limits in terms of memory consumption, causing aliasing artifacts. When scattering is too local w.r.t. cell size (e.g. at large scales

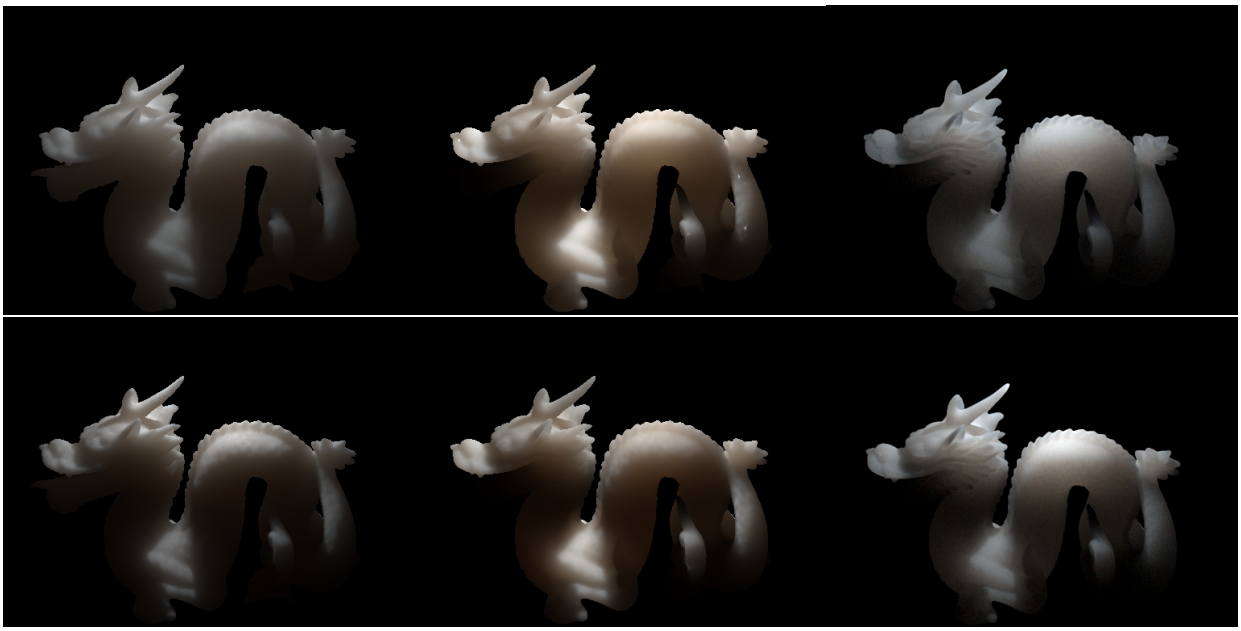


Figure 5: Comparison figure. The dragon model is lit by a point source placed directly above the model, and rendered for scales 20mm (top) and 40mm (bottom). Left: fast dipole approximation [15]. Middle: our method. Right: reference Monte Carlo solution. Our method is capable of resolving internal visibility, contrary to the dipole solution. For instance, the darkening at the lower half of the mouth is caused by visibility issues.

or for dense materials), the response to incident light cannot be represented faithfully. To deal with this case, a more efficient subdivision scheme is required. Aside from aliasing, such a scheme will also avoid the chromatic bias in the source term (see figure 5).

The method is limited by the assumptions inherent to the diffusion equation. Most importantly, low order scattering is important in certain cases, currently ignored by our model (see figure 5).

Acknowledgements

We would like to thank Jensen et al. [15] for publishing their acquired material parameters, Xin Tong for his assistance, and the anonymous reviewers for their constructive comments. The "sculpture" model in figure 6 is courtesy of FarField technology. The first and second author gratefully acknowledge funding by "tUL impulsfinanciering" and "Interdisciplinair instituut voor BreedBandTechnologie" (IBBT), respectively.

A Diffusion Approximation

The diffusion approximation is based on the observation that, in highly-scattering optically thick media, the angular dependence light distribution tends to be smoothed out due to the large number of scattering events [28]. Because of the weak dependence on direction, the diffuse intensity can be accurately approximated using 0th and 1st order spherical harmonics:

$$L_d(x,\vec{\omega}) = U_d(x) + \frac{3}{4\pi}\vec{F}_d(x)\vec{\omega} \tag{6}$$

where $U_d(x) = (1/4\pi)\int_{S^2} L_d(x,\omega)d\omega$ is the radiant fluence and $\vec{F}_d(x) = \int_{S^2} L_d(x,\vec{\omega})\vec{\omega}d\omega$ the vector irradiance.

The diffusion equation follows from this approximation (details Ishimaru's book [11]). The resulting equation is:

$$\vec{\nabla}\cdot[\frac{1}{3\sigma_{tr}}\vec{\nabla}U_d] = 4\pi\sigma_a U_d - S \tag{7}$$

With the incident radiance term S defined as (assuming no internal sources):

$$S(x) = 4\pi\sigma_s U_{ri}(x) - \vec{\nabla}\cdot[\frac{1}{3\sigma_{tr}}\vec{Q}_{ri}(x)]$$

At each location in the medium, the reduced incident radiance is defined as the attenuated radiance coming from external light sources, and is incorporated in the equation by its 0th and 1st order spherical harmonics projections U_{ri} and $\vec{Q}_{ri}$, respectively. The absorption and scattering transport coefficients (σ_a and σ_s respectively) characterize the medium and vary spatially for heterogeneous materials. The effective transport coefficient σ_{tr} is

defined as $\sigma_a + (1-g)\sigma_s$ where g controls the anisotropy of the scattering ($g < 0$, $g > 0$ and $g = 0$ corresponds to backward, forward and isotropic scattering, respectively).

The boundary condition is formulated as: [30, 11]

$$U_d \; - \frac{2A}{3\sigma_{tr}} \vec{n} \cdot \vec{\nabla} U_d + \frac{2A}{3\sigma_{tr}} \vec{n} \cdot \vec{Q}_1 = 0 \qquad (8)$$

where n is the inward-pointing surface normal.

A is a constant that depends on the indices of refraction at both sides of the surface.

References

[1] M. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. In *Journal of Computational Physics*, pages 484–512, 1984.

[2] Nathan A. Carr, Jesse D. Hall, and John C. Hart. Gpu algorithms for radiosity and subsurface scattering. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 51–59. Eurographics Association, 2003.

[3] Yanyun Chen, Xin Tong, Jiaping Wang, Stephen Lin, Baining Guo, and Heung-Yeung Shum. Shell texture functions. *ACM Trans. Graph.*, 23(3):343–353, 2004.

[4] Carsten Dachsbacher and Marc Stamminger. Translucent shadow maps. In P. Christensen and D. Cohen-Or, editors, *Proceedings of the 14th Eurographics Workshop on Rendering*, pages 197–201, Aire-la-Ville, Switzerland, June 25–27 2003. Eurographics Association.

[5] M. Day, P. Colella, M. Lijewski, C. Rendleman, and D. Marcus. Embedded boundary algorithms for solving the poisson equation on complex domains, 1998. Technical Report Lawrence Berkeley National Laboratory, LBNL-41811.

[6] D. DeZeeuw. A quadtree-based adaptively refined cartesian-grid algorithm for solution of the euler equations, 1993.

[7] Julie Dorsey, Alan Edelman, Henrik Wann Jensen, Justin Legakis, and Hans Kuhling Pedersen. Modeling and rendering of weathered stone. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 225–234. ACM Press/Addison-Wesley Publishing Co., 1999.

[8] Michael Goesele, Hendrik P. A. Lensch, Jochen Lang, Christian Fuchs, and Hans-Peter Seidel. Disco: acquisition of translucent objects. *ACM Trans. Graph.*, 23(3):835–844, 2004.

[9] Pat Hanrahan and Wolfgang Krueger. Reflection from layered surfaces due to subsurface scattering. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 165–174. ACM Press, 1993.

[10] Xuejun Hao and Amitabh Varshney. Real-time rendering of translucent meshes. *ACM Trans. Graph.*, 23(2):120–142, 2004.

[11] Akira Ishimaru. *Wave Propagation and Scattering in Random Media*. IEEE, 2002.

[12] J. Saltzman J. Bell, M. J. Berger and M. Welcome. Three dimensional adaptive mesh refinement for hyperbolic conservation laws. In *J. Sci. Comput.*, pages 127–138, 1994.

[13] Henrik Wann Jensen. Global Illumination Using Photon Maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 21–30, New York, NY, 1996. Springer-Verlag/Wien.

[14] Henrik Wann Jensen and Juan Buhler. A rapid hierarchical rendering technique for translucent materials. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 576–581. ACM Press, 2002.

[15] Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 511–518. ACM Press, 2001.

[16] H. Johansen and P. Colella. A cartesian grid embedded boundary method for poisson's equation on irregular domains. In *J. Comput. Phys. 147, 60*, 1998.

[17] Eric P. Lafortune and Yves D. Willems. Rendering Participating Media with Bidirectional Path Tracing. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 91–100, New York, NY, 1996. Springer-Verlag/Wien.

[18] Hendrik P. A. Lensch, Michael Goesele, Philippe Bekaert, Jan Kautz, Marcus A. Magnor, Jochen Lang, and Hans-Peter Seidel. Interactive rendering of translucent objects. In *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, page 214. IEEE Computer Society, 2002.

[19] Tom Mertens, Jan Kautz, Philippe Bekaert, Hans-Peter Seidel, and Frank Van Reeth. Efficient rendering of local subsurface scattering. In Reinhard Klein Jon Rokne and Wenping Wang, editors, *Proceedings of Pacific Graphics*, pages 51–58, 2003.

[20] Tom Mertens, Jan Kautz, Philippe Bekaert, Hans-Peter Seidel, and Frank Van Reeth. Interactive rendering of translucent deformable objects. In P. Christensen and D. Cohen-Or, editors, *Proceedings of the 14th Eurographics workshop on Rendering*, pages 130–140. Eurographics Association, 2003.

[21] Mark Pauly, Thomas Kollig, and Alexander Keller. Metropolis light transport for participating media. In B. Peroche and H. Rushmeier, editors, *Rendering Techniques 2000 (Proceedings of the Eleventh Eurographics Workshop on Rendering)*, pages 11–22, New York, NY, 2000. Springer Wien.

[22] Matt Pharr and Pat Hanrahan. Monte carlo evaluation of non-linear scattering equations for subsurface reflection. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 75–84. ACM Press/Addison-Wesley Publishing Co., 2000.

[23] Stphane Popinet. Gerris: A tree-based adaptive solver for the incompressible euler equations in complex geometries.

[24] Simon Premože, Michael Ashikhmin, and Peter Shirley. Path integration for light transport in volumes. In P. Christensen and D. Cohen-Or, editors, *Proceedings of the 14th Eurographics Workshop on Rendering*, pages 52–63, Aire-la-Ville, Switzerland, June 25–27 2003. Eurographics Association.

[25] Simon Premože, Michael Ashikhmin, Jerry Tessendorf, Ravi Ramamoorthi, and Shree Nayar. Practical rendering of multiple scattering effects in participating media. In *Proceedings of the 14th Eurographics Workshop on Rendering*, 2004.

[26] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes*. Cambridge University Press, 2nd edition, 1992.

[27] Holly E. Rushmeier and Kenneth E. Torrance. The zonal method for calculating light intensities in the presence of a participating medium. In *Computer Graphics (ACM SIGGRAPH '87 Proceedings)*, volume 21, pages 293–302, July 1987.

[28] Jos Stam. Multiple Scattering as a Diffusion Process. In P. M. Hanrahan and W. Purgathofer, editors, *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)*, pages 41–50, New York, NY, 1995. Springer-Verlag.

[29] Hackbusch W. *Multigrid Methods and Applications*. Springer-Verlag, Berlin, 1985.

[30] Keijzer M. Star M W and Starchi. Optical diffusion in layered media. *Appl. Opt.*, 27:1820–4, 1988.

Figure 6: Rendering results. ***1-2-3****: Venus, dragon and Buddha models with a mixture of marble; the ratio of minimum and maximum density is 1:10.* ***4****: Backlit dragon model.* ***5****: Chromatic shift for skim milk.* ***6-7****: Sculpture model with skim and whole milk.* ***8-9-10****: Marble dragon with varying anisotropy ($g = 0$, $g = .5$ and $g = .9$, respectively).*