Efficient Rendering of Local Subsurface Scattering

Tom Mertens[1] Jan Kautz[2] Philippe Bekaert[1] Frank Van Reeth[1] Hans-Peter Seidel[2]

Limburgs Universitair Centrum[1]
Diepenbeek, Belgium
{tom.mertens,philippe.bekaert,frank.vanreeth}@luc.ac.be

MPI Informatik[2]
Saarbrücken, Germany
{jnkautz,hpseidel}@mpi-sb.mpg.de

Abstract

A novel approach is presented to efficiently render local subsurface scattering effects. We introduce an importance sampling scheme for a practical subsurface scattering model. It leads to a simple and efficient rendering algorithm, which operates in image-space, and which is even amenable for implementation on graphics hardware. We demonstrate the applicability of our technique to the problem of skin rendering, for which the subsurface transport of light typically remains local. Our implementation shows that plausible images can be rendered interactively using hardware acceleration.

1. Introduction and Related Work

We frequently encounter translucent objects in our daily lives, such as skin, milk, marble, wax, and so on. Translucent objects have a distinct soft look due to light entering the material and scattering inside it. This process is known as subsurface scattering. The unique appearance of translucent objects cannot be achieved with simple surface reflection models. Even for materials that do not seem to be very translucent at first sight, such as skin for example, subsurface scattering is very important. Fine geometric detail, e.g. small wrinkles and bumps, is smoothed out by subsurface scattering and appears less prominent [7, 10].

Traditionally, subsurface scattering has been approximated with a Lambertian diffuse reflection. Obviously, this approximation fails for any material that exhibits noticeable translucency. For these materials, subsurface scattering effects can be rendered offline using a wide range of methods proposed for participating media, including finite element methods [21, 1, 22], (bidirectional) path tracing [4, 15], and photon mapping [9, 3], to diffusion approximations [25].

Recently an analytical model for subsurface scattering [10] was proposed, which eliminates the need for numerical simulation of subsurface light transport in homogeneous highly scattering optically thick materials such as milk and marble. A hierarchical integration techniques was proposed in [8] in order to use this model in global illumination algorithms. Unfortunately, this technique does not appear to allow interactive rendering.

Figure 1. An interactive rendering of a bump-mapped model on graphics hardware using measured BSSRDF parameters of human skin [10]. The image is rendered at roughly 4 to 5 frames per second.

Our goal is to render deformable, translucent objects at interactive rates under varying lighting and viewing conditions (see figure 1 and color page for examples). Lensch et al. [16] proposed a method, which is interactive, but only for rigid objects with fixed, possibly inhomogeneous, subsurface scattering properties. The method by Hao et al. [5] renders translucent but rigid objects in real-time. Recent work on real-time rendering with precomputed radiance transfer [24] can be extended to subsurface scattering [23], but also assumes static models. In comparison to our technique, these methods all require a costly precomputation phase.

Our computations are done from scratch each frame, allowing us to change virtually everything.

Real-time rendering with complex surface BRDFs has been widely studied, e.g. [6, 12, 17]. Unfortunately, these techniques cannot handle subsurface scattering because they assume that light is directly reflected and not scattered inside the object.

In contrast to the above cited techniques, we propose a method to interactively render arbitrary dynamic models with local subsurface scattering. In many cases, the effect of scattering remains local, depending on the material and the scale of the geometry. For instance, human skin scaled at 10 to 20cm does not exhibit much *global* response to the incoming light. The local response however is more dominant present in this case. If scaled large enough, other materials will behave in the same way.

To handle this case efficiently, we show that the model for subsurface scattering by Jensen et al. [10] can be rewritten as an integral in image-space instead of a (hierarchical [8]) integral over the object. We combine this with importance sampling of the BSSRDF, which allows for a very efficient computation of subsurface scattering. The new technique can be implemented in software as well as on graphics hardware. When running on graphics hardware, we achieve interactive frame rates for translucent, dynamic models. An example is depicted in figure 1, where our method was applied to a head model.

In concurrent work, Dachsbacher et al. [2] presented translucent shadow maps, which is the same in spirit as our technique. They render an irradiance texture from the light source view, which is filtered using mipmapping to perform the integration. In contrast, our technique renders an irradiance map from the eye view, while importance sampling acts as a filter.

2. Background

In this section, we first introduce the necessary background on subsurface scattering and the dipole source approximation [10, 8].

2.1. The Subsurface Reflection Equation

The following integral computes the shade of a surface point x_o on a translucent object (viewed from a direction ω_o):

$$L^e(x_o,\omega_o) = \int_{\mathsf{S}}\int_{\Omega^+(x_i)} L^i(x_i,\omega_i)S(x_i,\omega_i;x_o,\omega_o)\cdot (\omega_i\cdot N_i)d\omega_i dx_i, \quad (1)$$

$L^e(x_o,\omega_o)$ and $L^i(x_i,\omega_i)$ denote exitant/incident radiance respectively. S is the object's surface, $\Omega_+(x_i)$ is the hemisphere above x_i in normal direction N_i, and $S(x_i,\omega_i;x_o,\omega_o)$ is the bidirectional surface scattering distribution function (BSSRDF). In general, the BSSRDF is an eight-dimensional function, expressing what fraction of (differential) light energy entering the object at a location x_i from a direction ω_i leaves the object at a second location x_o into direction ω_o. Because of its high dimensionality, it is infeasible to precompute and store this term, especially since it depends on the object's geometry.

It has been shown [4, 10] that subsurface scattering can be modelled adequately using two components: single scattering and multiple scattering. Single scattering accounts for light that is scattered only once inside the medium. Only a small fraction of the outgoing radiance of a highly scattering translucent material, such as marble, milk, ..., is due to single scattering. The dominating term is the multiple scattering, which we are concerned with.

Multiple scattering diffuses the incident illumination, such that there is almost no dependence on the incident and outgoing direction anymore. Therefore it can be represented at high accuracy as a four-dimensional function $R_d(x_i,x_o)$, which only depends on the incident and outgoing positions. Additionally accounting for the Fresnel transmittance when light enters and leaves the material, we get the following subsurface scattering reflectance function:

$$S(x_i,\omega_i;x_o,\omega_o) = \frac{1}{\pi}F_t(\eta,\omega_o)R_d(x_i,x_o)F_t(\eta,\omega_i). \quad (2)$$

Substituting this into Equation 1, we get:

$$L^e(x_o,\omega_o) = \frac{1}{\pi}F_t(\eta,\omega_o)B(x_o) \quad (3)$$

$$B(x_o) = \int_{\mathsf{S}} E(x_i)R_d(x_i,x_o)dx_i \quad (4)$$

$$E(x_i) = \int_{\Omega^+(x_i)} L^i(x_i,\omega_i)F_t(\eta,\omega_i)(N_i\cdot\omega_i)d\omega_i \quad (5)$$

In order to render translucent objects efficiently, one needs an efficient way to solve for L^e at every surface point.

2.2. Dipole Source Approximation

First, we need to choose a model for the BSSRDF, and thus a model for R_d. This model has to fulfill two main criteria: it should be adaptable to different materials and should allow for importance sampling to speed up the computation later on.

$R_d(x_i;x_o)$ can be accurately determined using a full simulation. Many different techniques have been proposed (coming from the area of participating media), e.g. [4, 22, 15, 9]. While these techniques are effective, they do not fulfill the above criteria.

We chose to use a recently introduced model for the BSSRDF in homogeneous media as the basis for our work

$$
\begin{aligned}
R_d(x_i, x_o) &= \frac{\alpha'}{4\pi}\left[z_r(1+\sigma s_r)\frac{e^{-\sigma s_r}}{s_r^3} + z_v(1+\sigma s_v)\frac{e^{-\sigma s_v}}{s_v^3}\right] \\
z_r &= 1/\sigma_t', \qquad z_v = z_r + 4AD \\
s_r &= \|x_r - x_o\|, \text{ with } x_r = x_i - z_r \cdot N_i \\
s_v &= \|x_v - x_o\|, \text{ with } x_v = x_i + z_v \cdot N_i \\
A &= \frac{1+F_{dr}}{1-F_{dr}} \\
F_{dr} &= -\frac{1.440}{\eta^2} + \frac{0.710}{\eta} + 0.668 + 0.0636\eta \\
D &= 1/3\sigma_t', \qquad \sigma = \sqrt{3\sigma_a\sigma_t'} \\
\sigma_t' &= \sigma_a + \sigma_s', \qquad \alpha' = \sigma_s'/\sigma_t' \\
\sigma_s' &= \text{reduced scattering coefficient (given)} \\
\sigma_a &= \text{absorption coefficient (given)} \\
\eta &= \text{relative refraction index (given)} \\
F_t(\eta, \omega) &= \text{Fresnel transmittance factor}
\end{aligned}
$$

Table 1. The dipole source BSSRDF model. This paper presents an efficient importance sampling strategy for computing integrals of this model.

[10]. This model is based on a dipole source approximation for the solution of a diffusion equation [7, 25] that models light transport in densely scattering media. We shall see that it fulfills the above requirements.

The key idea behind the dipole source approximation for $R_d(x_i, x_o)$ [10], is elegant and simple. An incoming ray at position x_i on a homogeneous, planar, and infinitely large and thick medium is converted into a dipole source (i.e. two sources) at the same position. One source of the dipole is placed at a distance z_v above the surface and the second source at a distance z_r below the surface at x_i. $R_d(x_i, x_o)$ is then obtained by summing a sort of illumination contribution at x_o, due to the two sources near x_i. The result is a function of the distance r (between x_i and x_o) only; see Table 1.

Although the dipole source approximation is only valid for planar surfaces [7, 14], using it for curved surfaces yields highly plausible renderings, as has been shown in several complex examples by Jensen et al. [10, 8]. We will use it in the same spirit. The model is also inherently limited to homogeneous materials. The appearance of heterogeneous materials was simulated by means of texture mapping techniques in previous work and is done in the work presented here too.

3. Overview

The most expensive part of computing the shading contribution due to subsurface scattering is the evaluation of equation 4. One has to integrate over the whole surface of an object and sum up all the contributions due to subsurface scattering.

A straightforward implementation simply performs standard Monte Carlo integration using randomly (and uniformly) distributed samples over the object. A speed-up can be achieved by building a spatial hierarchy on the samples [8]; samples higher up in the hierarchy are taken, if they are far away. The drawback of this method is, that one has to build the hierarchy first.

A more efficient integration can be performed using importance sampling [11]. Importance sampling distributes the samples according to the function to be integrated, in our case R_d. To this end, we propose a method for generating such importance samples in section 4.

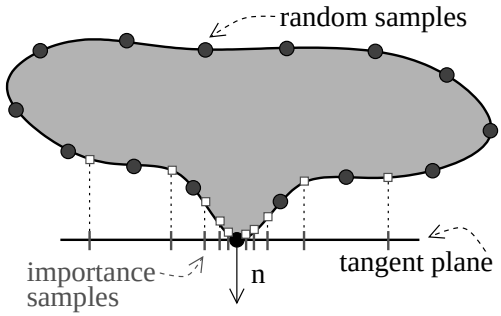

Figure 2. Overview of the proposed technique: we locally sampling incoming lighting in the tangent plane, according to the importance of the BSSRDF function.

In order to further speed up the computation, we do not (locally) integrate over the surface of the object, but instead integrate the irradiances in image-space (see section 5). The basic algorithm is depicted in figure 2. Importance samples are generated on the tangent plane at x_o. These are then projected on the object (in case of a perspective viewer with a perspective projection), and the irradiances from there are summed up. This makes the new algorithm amenable for implementation on graphics hardware.

In section 6, we detail how the new algorithm can be implemented using graphics hardware. Section 7 shows various results, that can be achieved with this method. Finally, we will conclude in section 8.

4. Importance Sampling of the BSSRDF

In this section we derive an exact importance sampling scheme for the BSSRDF model presented by Jensen et al. [10] (see table 1), which allows us to evaluate equation 4 more efficiently. The goal is to find sample distances r_i, which are distributed according to $R_d(r)r$ (this will become

clear in the following section), with $R_d(r)$ defined as:

$$\begin{aligned} R_d(r) &= \frac{\alpha'}{4\pi}\left[R_d^r(r) + R_d^v(r)\right], \\ R_d^r(r) &= z_r(1+\sigma s_r)\frac{e^{-\sigma s_r}}{s_r^3}, \\ R_d^v(r) &= z_v(1+\sigma s_v)\frac{e^{-\sigma s_v}}{s_v^3}. \end{aligned}$$

We now integrate one of the two last terms over the 2D plane. By substituting s with $\sqrt{r^2+z^2}$ we obtain:

$$\begin{aligned} \int_0^\infty z(1+\sigma s)\frac{e^{-\sigma s}}{d^3} r\, dr &= z\sigma\left[-\frac{1}{u}e^{-u}\right]_{u=\sigma z}^{u=\infty} \quad (6) \\ &= e^{-\sigma z}, \quad (7) \end{aligned}$$

with z either z_r or z_v and s either s_r or s_v. The reflectance thus is [10]:

$$\rho = \frac{\alpha'}{2}\left[e^{-\sigma z_r} + e^{-\sigma z_v}\right]$$

Exact sampling is performed as follows. Draw a uniform random number ξ from the unit interval. We must find a distance $r(\xi)$ such that the probability of finding $r(\xi)$ in an interval $[a,b]$ equals $\frac{2\pi}{\rho}\int_a^b R_d(r) r\, dr$, i.e. the probability density function is defined as: $p(r) := \frac{2\pi}{\rho} R_d(r) r\, dr$.

Now, pick one of the two terms R_d^r or R_d^v by testing whether

$$\xi < \frac{e^{-\sigma z_r}}{e^{-\sigma z_r} + e^{-\sigma z_v}}.$$

This threshold corresponds with the magnitude of each integrated term (see equation 7). The chosen term is called $R_d^\star$.

We continue with inverting its cumulative distribution function:

$$\int_0^{r(\xi)} R_d^\star(r) r\, dr = \xi \int_0^\infty R_d^\star(r) r\, dr.$$

Using equation 6 and 7, we obtain:

$$(1-\xi)e^{-\sigma z} = \frac{1}{\sqrt{1+\left(\frac{r}{z}\right)^2}} e^{-\sigma z\sqrt{1+\left(\frac{r}{z}\right)^2}}.$$

Substituting $u = \sqrt{1+\left(\frac{r}{z}\right)^2}$ gives us the following equality:

$$\sigma z(u-1) + \log u + \log(1-\xi) = 0,$$

which can be solved numerically for u. Note that this function is a smooth, monotonically increasing function for which we can find its root rapidly with Newton's method for instance. Finally, we get an importance sample distance r with:

$$r = z\sqrt{u^2-1}.$$

5. Integration Over the Surface

In this section we describe how we employ the importance sampling scheme.

Two problems arise in this context. Firstly, for a given importance sampling distance at a point on the surface, it is not trivial to construct a location at this distance directly on the surface. This would require access to the local geometry, as well as a complex search routine. Therefore we simply construct the samples in the tangent plane of that point.

The second problem is the acquisition of irradiance information over the surface. To this end, we render the irradiance once for one (or possibly more) reference views into a 2D image or texture map. In our implementation, we take the camera view to generate this irradiance texture. Another possibility might be to use the light view, similar to shadow mapping [27]. However, this can cause artifacts at locations which are oriented perpendicularly to the light due to severe undersampling. Also, rendering irradiance once for the camera view enables us to efficiently handle more lights without much extra cost, or more complex lighting (e.g. from an environment map).

The resulting integration procedure is simple: for each point to shade, construct a set of samples in the tangent plane, project this point on the surface w.r.t. the center of projection of our irradiance texture (see figure 3). Essentially, the integral is solved in *image space*.

Consider the local scattering integral (equation 4):

$$\begin{aligned} I &= \int_A R_d(r) E(p) dA_p \\ &= \int_{A'} R_d(r) E(p') \left|\frac{dA_p}{dA_{p'}}\right| dA_{p'} \quad (8) \end{aligned}$$

For each sample point we can then easily look up the irradiance $E(p')$ in the irradiance texture. Since we assume local scattering, we take A to be a fraction of the surface S, and A' the corresponding area on the tangent plane. For simplicity, A' is considered to be a disc of radius R'.

The transformation implies multiplying with a Jacobian $\left|\frac{dA_p}{dA_{p'}}\right|$. We derive it based on the notion that the solid angles occupied by the differential areas surrounding p and p' w.r.t. e (the center of projection), are equal:

$$\delta A_p \frac{\cos\alpha}{d^2} = \delta A_{p'} \frac{\cos\alpha'}{d'^2},$$

with $d = \|e-p\|$ and $d' = \|e-p'\|$. Substituting this in equation 8 yields:

$$\begin{aligned} I &= \int_{A'} R_d(r) E(p') \frac{\cos\alpha}{\cos\alpha'} \frac{d'^2}{d^2} dA_{p'} \\ &= \int_0^{2\pi}\int_0^{R'} R_d(r) E(p') \frac{\cos\alpha}{\cos\alpha'} \frac{d'^2}{d^2} r\, dr\, d\phi \end{aligned}$$

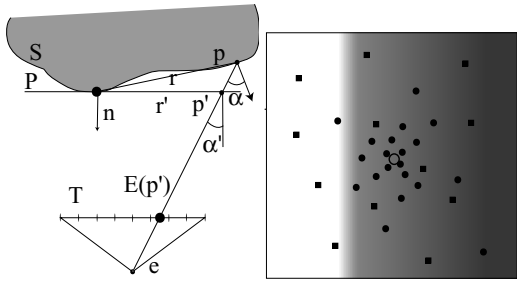

Figure 3. Left: geometry for sampling the irradiance in the tangent plane. The irradiance at the projected sample point p can be retrieved in the irradiance texture T. Right: combined sampling of importance samples (dots) and uniform samples (squares) for a point (circle) near a shadow border. Almost none of the importance samples will get a non-zero contribution. The uniform samples have a higher chance of getting a significant contribution.

The last step results from a transformation to polar coordinates, which stems with the distribution importance sampling routine.

The integral is solved numerically using a set of N samples $p_i = (\phi_i, r_i)$, where r_i is generated as described in section 4 and ϕ_i is uniform:

$$I = \frac{\rho}{2\pi N} \sum_{i}^{N} \frac{R_d(r_i)}{R_d(r'_i)} E(p'_i) \frac{\cos\alpha_i}{\cos\alpha'_i} \frac{{d'_i}^2}{{d_i}^2}. \qquad (9)$$

Since the importance samples are generated in tangent space, i.e., we generate r'_i, but we actually take samples with distances r_i, the samples are weighted by $R_d(r_i)/R_d(r'_i)$.

Final exit radiance is computed by multiplying I with the Fresnel transmittance for the current viewing direction, see equation 3.

6. Implementation

6.1. Variance Reduction

Artifacts caused by variance are inherent to Monte Carlo techniques. Some measures can be taken to alleviate this:

Stratified Sampling We employ a deterministic pseudo random sampling pattern as input for our importance sampling algorithm. Interleaved sampling [13] is used to avoid banding artifacts, and gracefully transforms noise into a dither-like pattern. Also, it is more amenable for a hardware implementation.

Combined Sampling The importance sampling algorithm behaves well as long as irradiance does not vary quickly. This is not a safe assumption. For instance, figure 3 depicts how in unlit regions near shadow borders, few importance samples will ever have a non-zero contribution. Uniform samples will perform better in this case.

To combat this, we generate a set of uniform samples together with the importance sample set. Using the balance heuristic proposed in [26], we can safely combine the two sets in an unbiased fashion.

6.2. Using Graphics Hardware

Our algorithm can be implemented on modern consumer graphics hardware. Current graphics technology offers a high degree of programmability at the fragment level, and provides full floating point precision throughout its render pipeline. These features make the implementation of our algorithm feasible in hardware.

Before rendering, we first compute a set of importance sampled distances r_i and then precompute an offset vector for each importance sample, which is used to quickly construct a sample from the tangent basis vectors at each pixel. Also the inverse probability density function is precomputed as a weight for each sample, as well as the weight for the balance heuristic [26].

On an ATI Radeon 9700Pro, which have used for our implementation, the algorithm requires $N + 2$ passes, where N is the number of samples. The first pass renders an irradiance texture in an off-screen buffer for the camera view. Aside from the radiance, the depth buffer from this view is also stored in a separate texture. An additional depth buffer is employed to perform shadow mapping [27].

The accumulation of the N samples requires two floating point buffers. In one of the passes, one acts as render target and the other as temporary storage of previous passes. In a subsequent pass, the roles are switched in order to avoid a costly texture copy. The alpha channel is used as a counter for the number of samples. It is only incremented if a valid irradiance sample is found, i.e., if the depth of the fetched irradiance pixel is closer than the far plane. The components of the summation in equation 9 are computed as follows in the fragment shader:

- The sample weights are passed as a single (global) parameter to the fragment shader.
- The irradiance $E(p'_i)$ is retrieved using a projective texture fetch.
- The function R_d itself is stored as a 1D floating point texture for a certain range. Using the information in the depth texture (projective fetch), the location p_i on the surface is computed, in order to derive r_i. Now we weight the irradiance sample appropriately (see equation 9).

- We omit the Jacobian for simplicity. For most cases this is a safe assumption, since the tangent plane approximates the surface for a small area A.

The resulting fragment shader contains 37 instructions. The final pass consists of normalizing the resulting pixels with the sample counter in the alpha channel, multiplying with the Fresnel coefficient, and applying the simple tone mapping operator from [20].

On future hardware, such as ATI's Radeon 9800, it will be possible to implement this algorithm in a single pass, since an infinite number of instructions is allowed. This will make the accumulation approach unnecessary, thereby saving a lot of memory access overhead.

7. Results

The quality of our algorithm can be judged from the image in figure 1. A head model scaled to 10cm was rendered using the scattering parameters for human skin [10]. A closeup can be seen in figure 4. The used irradiance map can be seen in the same figure. A bump map was applied when computing the irradiances. Subsurface scattering smoothes out these bump considerably, as has been noted before [10]. The renderings in figure 1 and 4 have an additionally applied base map (e.g., for the eye brows) and a gloss map.

Rendering speed is about 4 to 5 frames per second for a 500 by 500 image on an ATI Radeon 9700Pro. Our implementation was also tested on an NVIDIA GeForceFX 5800 board, for which we obtained similar timings. Rendering speed degrades roughly linearly with the number of pixels rendered. We expect that our inefficient proof-of-concept implementation can be further optimized for faster rendering.

Closeups of the head model can be found in figure 4. The irradiance texture used in the first rendering can be seen next to it. The bottom-left image shows the shadow boundary from the neck region. The original shadow has actually a sharp boundary, as seen in the irradiance image. Subsurface scattering causes the shadow boundary to be diffused over a larger area. The dithering structure comes from interleaved sampling [13]. The two top-right images show the region on the forehead under different illumination conditions. The typical reddish color shifts are very prominent in this example. More complex lighting can be applied: in the bottom row a rendering is shown using a projective stained glass texture.

Also in figure 4, example renderings of two other materials can be seen. The left image was rendered with milk and the right image with marble. Here we did not apply any base or gloss map for the renderings. Even the very translucent milk is rendered without obvious artifacts. All the shadow regions show considerable subsurface scattering (brownish tint).

Figure 5 shows a comparison of Jensen et al.'s method [8] with our method. The results are virtually the same, apart from some small scaling factor. This is due to the global undershoot from not taking the whole surface into account in the integration. Small differences can be seen at sharp boundaries. E.g., the boundary of the lower lip is not as smooth as with Jensen et al.'s method. This is because no contributions can be gathered from underneath the lip as this part not visible in the irradiance image. Noise at grazing angles of the head is hardly noticeable due to the use of interleaved sampling.

The appearance of spatially varying materials can be simulated by altering the BSSRDF parameters per pixel according to a texture function. In figure 5 we demonstrate this by lighting a marble plane with a checkered irradiance pattern. Subtle lighting effects suggest the idea of some internal structure. Even temporal variation in the texture can be rendered at interactive rates. We precompute a set of importance sample offsets, which are generated for an interpolated set of BSSRDF parameters, and are stored in texture memory. Note that we cannot interpolate the offsets themselves, due to the non-linear effect of altering the BSSRDF parameters. Since the importance sampling algorithm is simple enough, the offsets might also be generated on the GPU.

7.1. Discussion

Our method allows a *full* rendering at interactive rates, i.e., one can alter the viewpoint, lighting, material properties, geometry and even topology.

The whole procedure is executed entirely on the GPU, and virtually no precomputation is required on the host CPU, except for the marginal cost of generating importance samples and the R_d texture.

In a situation where the object is lit from behind, light can scatter through thin geometry (e.g. the ears on a head), but the irradiance necessary to compute this may not be available in the irradiance image. A simple solution to this problem is to sample multiple irradiance images and combining them (e.g. using the balance heuristic [26]). This also means that the rendering cost will be multiplied by the number of irradiance images, roughly. In the case of a point source, a combination of the eye view and the light source view should cover every contribution sufficiently.

Also, in certain cases banding and *ghosting* artifacts may occur due to undersampling. Especially when a too translucent material is chosen and the integration area A is too large, variance causes disturbing artifacts. In any case, the response should be limited. Contrary to the method proposed in this paper, the interactive technique in [18] handles the global response efficiently, but does not compute the local response very accurately. In other words, the two meth-

ods are perfectly complementary, and a combination would be able to render a wide variety of scattering effects.

We noted that the performance of our method is primarily bandwidth limited, since many texture lookups and memory writes need to be performed. Future hardware will have faster and wider memory interfaces, which will be beneficial to our method.

8. Conclusions and Future Work

We have presented a simple but efficient algorithm to render the local effect of subsurface scattering. It is based on an exact importance sampling scheme for the BSSRDF. Local integration over the surface is performed in the tangent plane for each pixel. The associated irradiance for each sample in the tangent plane is looked up from a precomputed texture for the camera view. Our algorithm is simple enough to be implemented on modern consumer graphics hardware. Results show that plausible subsurface scattering can be rendered interactively for varying viewpoint and lighting. Also, material properties, geometry and even topology can be changed without any additional cost.

We would like to investigate more sophisticated variance reduction techniques in order to reduce the number of integration passes, possibly achieving real-time rates.

A single scattering term, as well as realistic local reflection similar to the demonstration software of NVIDIA [19], can be added to further improve realism.

Acknowledgements

The head model in figures 1 and 4 is courtesy of NVIDIA. This work is partly funded by the European Commission (European Regional Development Fund) and the Flemish Government (IWT).

References

[1] P. Blasi, B. L. Saëc, and C. Schlick. A Rendering Algorithm for Discrete Volume Density Objects. *Computer Graphics Forum*, 12(3):201–210, 1993.

[2] C. Dachsbacher and M. Stamminger. Translucent shadow maps. In *Proceedings of Eurographics Symposium on Rendering*, pages 197–201, jun 2003.

[3] J. Dorsey, A. Edelman, J. Legakis, H. W. Jensen, and H. K. Pedersen. Modeling and Rendering of Weathered Stone. In *Proceedings of SIGGRAPH 99*, pages 225–234, 1999.

[4] P. Hanrahan and W. Krueger. Reflection from layered surfaces due to subsurface scattering. In *Proceedings of SIGGRAPH 93*, pages 165–174, 1993.

[5] X. Hao, T. Baby, and A. Varshney. Interactive Subsurface Scattering for Translucent Meshes. In *Proceedings 2003 ACM Symposium on Interactive 3D Graphics*, page to appear, april 2003.

[6] W. Heidrich and H.-P. Seidel. Realistic, Hardware-accelerated Shading and Lighting. In *Proceedings of SIGGRAPH 99*, pages 171–178, Aug. 1999.

[7] A. Ishimaru. *"Wave Propagation and Scattering in Random Media"*, volume 1. Academic Press, 1978.

[8] H. W. Jensen and J. Buhler. A Rapid Hierarchical Rendering Technique for Translucent Materials. *ACM Transactions on Graphics*, 21(3):576–581, July 2002.

[9] H. W. Jensen and P. H. Christensen. Efficient Simulation of Light Transport in Scenes With Participating Media Using Photon Maps. In *Proceedings of SIGGRAPH 98*, pages 311–320, 1998.

[10] H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan. A Practical Model for Subsurface Light Transport. In *Proceedings of SIGGRAPH 2001*, pages 511–518, August 2001.

[11] M. H. Kalos and P. A. Whitlock. *Monte Carlo Methods, Volume 1: Basics*. Wiley, 1986.

[12] J. Kautz and M. D. McCool. Interactive Rendering with Arbitrary BRDFs using Separable Approximations. In D. Lischinski and G. W. Larson, editors, *Tenth Eurographics Rendering Workshop 1999*, pages 281–292, June 1999.

[13] A. Keller and W. Heidrich. Interleaved sampling. In *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering*, pages 269–276, June 2001.

[14] J. J. Koenderink and A. J. van Doorn. Shading in the Case of Translucent Objects. In *Human Vision and Electronic Imaging VI*, pages 312–320. SPIE, 2001.

[15] E. P. Lafortune and Y. D. Willems. Rendering Participating Media with Bidirectional Path Tracing. In *Eurographics Rendering Workshop 1996*, pages 91–100, 1996.

[16] H. P. A. Lensch, M. Goesele, P. Bekaert, J. Kautz, M. A. Magnor, J. Lang, and H.-P. Seidel. Interactive Rendering of Translucent Objects. In *Proceedings of Pacific Graphics 2002*, pages 214–224, October 2002.

[17] D. McAllister, A. Lastra, and W. Heidrich. Efficient Rendering of Spatial Bi-directional Reflectance Distribution Functions. In *Proceedings of Graphics Hardware 2002*, pages 79–88, September 2002.

[18] T. Mertens, J. Kautz, P. Bekaert, F. V. Reeth, and H.-P. Seidel. Interactive rendering of translucent deformable objects. In *Proceedings of Eurographics Symposium on Rendering*, pages 130–140, jun 2003.

[19] Nvidia home page. `http://www.nvidia.com`.

[20] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda. Photographic tone reproduction for digital images. *ACM Transactions on Graphics*, 21(3):267–276, July 2002.

[21] H. E. Rushmeier and K. E. Torrance. Extending the Radiosity Method to Include Specularly Reflecting and Translucent Materials. *ACM Transactions on Graphics*, 9(1):1–27, 1990.

[22] F. X. Sillion. A unified hierarchical algorithm for global illumination with scattering volumes and object clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3):240–254, 1995.

[23] P.-P. Sloan, J. Hall, J. Hart, and J. Snyder. Clustered Principal Components for Precomputed Radiance Transfer. In *Proceedings of SIGGRAPH 2003*, page to appear, July 2003.

Figure 4. Top row (from left to right): closeup of the head model with the corresponding irradiance map next to it. Notice how the roughness of the surface is washed away due to the subsurface scattering. Next two: forehead lit from above and forehead lit from the side. Note the obvious color shifts due to scattering. Bottom row: shadow region on neck. Next: the same model with skin lit by a stained glass texture. Last two: milk and marble materials applied to the model.

Figure 5. Comparison of Jensen et al.'s method (left) with our method (middle). The results are virtually the same, apart from a small scaling factor which we accounted for here. Right: rendering with spatially and temporally varying BSSRDF parameters.

[24] P.-P. Sloan, J. Kautz, and J. Snyder. Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments. In *Proceedings of SIGGRAPH 2002*, pages 527–536, July 2002.

[25] J. Stam. Multiple scattering as a diffusion process. In *Eurographics Rendering Workshop 1995*, pages 41–50, 1995.

[26] E. Veach and L. J. Guibas. Optimally combining sampling techniques for monte carlo rendering. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, pages 419–428, Aug. 1995.

[27] L. Williams. Casting curved shadows on curved surfaces. In *Computer Graphics (SIGGRAPH '78 Proceedings)*, volume 12, pages 270–274, Aug. 1978.