Interactive Subsurface Scattering for Translucent Meshes

Xuejun Hao Thomas Baby Amitabh Varshney

Department of Computer Science and UMIAC
University of Maryland at College Par
College Park, MD 20742
{hao, thomas, varshney}@cs.umd.edu

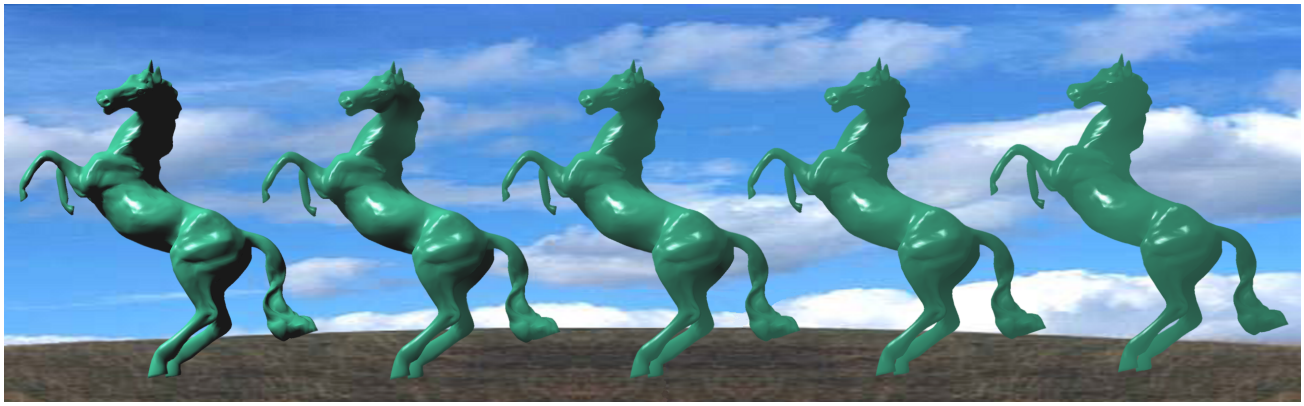


Figure 1: Different levels of Subsurface Scattering (increasing from left to right) on the model of a horse (14,521 vertices)

Abstract

We propose a simple lighting model to incorporate subsurface scattering effects within the local illumination framework. Subsurface scattering is relatively local due to its exponential falloff and has little effect on the appearance of neighboring objects. These observations have motivated us to approximate the BSSRDF model and to model subsurface scattering effects by using only local illumination. Our model is able to capture the most important features of subsurface scattering: reflection and transmission due to multiple scattering.

In our approach we build the neighborhood information as a preprocess and modify the traditional local illumination model into a run-time two-stage process. In the first stage we compute the reflection and transmission of light on the surface. The second stage involves bleeding the scattering effects from a vertex's neighborhood to produce the final result. We then show how to merge the run-time two-stage process into a run-time single-stage process using precomputed integral. The complexity of our run-time algorithm is $O(N)$, where N is the number of vertices. Using this approach, we achieve interactive frame rates with about one to two orders of magnitude speedup compared with the state-of-the-art methods.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms; I.3.6 [Computer Graphics]: Methodology and Techniques—Graphics data structures and data types; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture.

Additional Keywords: Reflection models, subsurface scattering, local illumination, BSSRDF.

1 Introduction

Photo-realistic image synthesis remains one of the primary goals of graphics. To achieve this, it is necessary to model the interaction of light with objects in a physically correct manner. Several illumination models have been developed for image synthesis. Most of them model the bidirectional reflectance distribution function (BRDF). A good BRDF model, either derived or measured, can give highly realistic visual effects. However, the basic assumption of BRDF models, that light enters and exits the object surface at the same point, is sometimes not valid. For example, when subsurface scattering is involved the light enters the object at one point and exits at another. Thus, the BRDF models are just approximations of the more general bidirectional surface scattering reflectance distribution function (BSSRDF). BRDF models are inadequate to simulate the appearance of materials with high subsurface scattering.

Hanrahan and Krueger [1993] have modelled the subsurface scattering in layered surfaces in terms of one-dimensional linear transport theory, and derived analytical expressions for single scattering events. They incorporated their results into a BRDF model. The model is fast but also has the shortcoming of the BRDF assumption. More recently, Dorsey *et al.* [1999] simulated the subsurface trans-

fer by solving the radiative transfer equation using photon maps. Koenderink and van Doorn [2001] proposed to model the light scattering in translucent objects as a diffusion process. Stam [2001] used a discrete-ordinate solution of the radiative transfer equation to model the multiple anisotropic scattering for the human skin layer bounded by two rough surfaces. Another contribution of [Stam 2001] is the derivation of a bidirectional transmittance distribution function (BTDF) to complement the BRDF models. Pharr and Hanrahan [2000] have taken a different approach. Instead of simulating the energy transport, they have focused on the scattering behavior and solve a non-linear integral scattering equation using Monte Carlo evaluation. Also Jensen *et al.* [1999] have used path tracing to simulate subsurface scattering in wet materials. These approaches are able to simulate all the effects of subsurface scattering and generate impressive images, but are slow.

Jensen *et al.* [2001] have suggested a simple and more efficient approach to simulate scattering media. They approximate the BSSRDF with an exact solution for single scattering and a dipole point source diffusion approximation for multiple scattering. Their algorithm is impressively fast. For example, for one scene they reduced the rendering time from 1250 minutes using full Monte Carlo simulation to 5 minutes with nearly indistinguishable visual difference. This makes the practical simulation of the subsurface scattering phenomena feasible. Later, by ignoring the single scattering events, and including only the multiple scattering diffusion approximation for the modelling of translucent material, Jensen and Buhler [2002] achieve up to 7 seconds per frame for a complicated lighting environment. Lensch *et al.* [2002] have used a preprocessing step to compute the impulse response for each surface point under subsurface scattering. They store it in two ways depending on whether it is a local or global effect. The local effect is modelled as a filter kernel and stored in a texture map and global response is stored as vertex-to-vertex throughput factors. During runtime the local and global responses are combined to form the final image, and they achieve speeds similar to [Jensen and Buhler 2002]. In this paper, we take the next step to enable the subsurface scattering effects for interactive rendering.

We note that subsurface scattering, although a global effect, is largely a local one due to its exponential falloff, which limits the volume it can affect. Therefore even though the light does not necessarily exit an object at the same point it enters, as required by the BRDF model, it will for all practical purposes exits within a short distance of its entry point. Thus, light that enters at one point can only affect the intensity at nearby surface points due to subsurface scattering and will have little effect on the appearance of distant regions on the same object or other objects in the same scene. This observation makes it likely that one should be able to simulate the subsurface scattering effects by making modifications to the existing local illumination models and thereby progress towards the goal of interactively rendering subsurface scattering effects. This paper describes our efforts to achieve that goal.

We approximate the BSSRDF for subsurface scattering based on both, the underlying physics processes and the visual appearance. As Jensen and Buhler [2002] have shown the visual appearance for translucent materials can be almost entirely simulated by only considering multiple scattering. Therefore in this paper we shall focus only on the multiple scattering events. The model we propose here is an empirical one in the sense that it is not completely based on microscopic physics unlike most approaches described above that are. Our model is rather a macroscopic appearance-driven one. It captures most of the important features of subsurface scattering: multiple scattering reflection and transmission. It can generate visually appealing images, and more importantly, is very fast due to its local illumination characteristics and the preprocessed local neighborhood information. The main contributions of this paper are:

1. We show that it is possible to incorporate subsurface scattering effects into a local illumination model, which makes it easy and efficient to simulate the phenomena in many applications that use local illumination models and cannot afford the global ray-tracing approach.

2. We provide, with reasons, methods to approximate the main features of subsurface scattering, i.e., reflection and transmission due to multiple scattering, for generating realistic visual effects.

3. We modify the local illumination process into a run-time two-stage process: a traditional local lighting stage and a scatter-bleeding stage. We then show how to merge the run-time two-stage process into a run-time single-stage one by using pre-computed integral. We improve the complexity of the run-time algorithm from $O(N^2)$ to $O(N)$. This allows us to achieve interactive frame rates for simulating the subsurface scattering effects.

2 Previous and Related Work

Illumination models for image synthesis can be empirically based or physically based. The Phong illumination model is an example of an empirically-based model [Phong 1975]. Physically-based models are derived from the principles of light-object interaction, using either geometrical optics or wave optics.

Cook-Torrance [1981] model is an example of the physically-based models using geometrical optics. It can compute the directional distribution of light, as well as the color shift with different angles of incidence and different kinds of materials. Other geometrical-optics-based models include microfacet-based approaches [Ashikhmin et al. 2000; Blinn 1977]. Inverse rendering methods can produce high-quality illumination models from images [Cabral et al. 1987; Debevec et al. 2000; Ramamoorthi and Hanrahan 2001; Sato et al. 1997; Yu et al. 1999]. Significant efforts have been devoted to determining the BRDF of an object. Some of the methods to directly measure the BRDF include [Greenberg et al. 1997; Marschner et al. 1999; Ward 1992]. The wave-optics-based models are usually less intuitive, but have the advantage of being able to model several phenomena, such as interference and diffraction, that cannot be directly modelled using geometrical optics. Kajiya [1985] has used scalar-form Kirchhoff approximation to compute the BRDF of surfaces with anisotropy. He *et al.* [1991] have presented a general local reflection model based on vector-formed Kirchhoff wave diffraction theory and have given an analytical formula to compute the BRDF for surfaces with roughness, including the polarization and directional Fresnel effects. Bahar and Chakrabarti [1987] have computed the differential scattering cross-section of a wave from rough metallic surfaces using electromagnetic theory. Stam [1999] and Sun *et al.* [2000] have extended the He-Torrance model [1991] to handle anisotropic reflections and demonstrated the diffraction effects for a compact disk. All these models for computing BRDF assume that light enters and exits on the same surface point. In most cases this assumption is valid and the resulting BRDF models provide convincing visual appearance.

Although most visual effects can be simulated using BRDF models, some, such as subsurface scattering effects, are hard. In such cases, normally we have to go back to the more general BSSRDF model. As we mentioned in the previous section, researchers [Dorsey et al. 1999; Hanrahan and Krueger 1993; Jensen et al. 1999; Jensen et al. 2001; Jensen

and Buhler 2002; Koenderink and van Doorn 2001; Lensch et al. 2002; Pharr and Hanrahan 2000; Stam 2001] have done an impressive job of modelling this phenomena. In this paper, we attempt to build a simple, approximate model based on previous methods and accommodate it into a local illumination model to make the effects more widely accessible for different applications, especially those that cannot afford the global ray tracing or Monte Carlo approaches.

3 Our Simplified Subsurface Scattering Model

As mentioned earlier, subsurface scattering for highly-scattering materials cannot be modelled with BRDF models, and the more general BSSRDF model is needed. The BSSRDF model relates the illumination of a surface point with the light distribution at other points. As stated in [Jensen et al. 2001]:

$$dL_o(x_o, \vec{\omega_o}) = S(x_i, \vec{\omega_i}; x_o, \vec{\omega_o}) d\Phi_i(x_i, \vec{\omega_i})$$

where $L_o(x_o, \vec{\omega_o})$ is the outgoing radiance at point x_o in direction $\vec{\omega_o}$, $\Phi_i(x_i, \vec{\omega_i})$ is the incident flux at point x_i in direction $\vec{\omega_i}$, and $S(x_i, \vec{\omega_i}; x_o, \vec{\omega_o})$ is the BSSRDF.

The total outgoing radiance is computed by an integral over all the incoming directions and the area A [Jensen et al. 2001]:

$$L_o(x_o, \vec{\omega_o}) = \int_A \int_{2\pi} S(x_i, \vec{\omega_i}; x_o, \vec{\omega_o}) L_i(x_i, \vec{\omega_i})(\vec{n_i} \cdot \vec{\omega_i}) d\omega_i dA(x_i)$$

where $\vec{n_i}$ is the normal at x_i. The effect of BSSRDF for subsurface scattering can be seen in Figure 2.

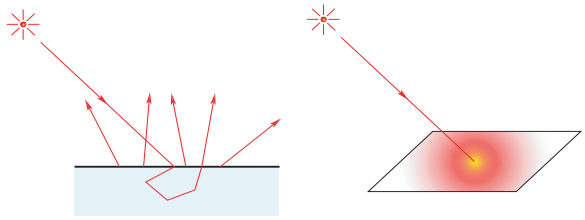


Figure 2: Scattering of light in BSSRDF models (based on [Jensen *et al.*,2001])

3.1 Locality of Subsurface Scattering Effects

The main rationale behind a possible combination of local illumination model with subsurface scattering effects is based on the key observation that the effects are well localized. Subsurface scattering, although a global illumination property in the sense that the illumination on a surface point is affected by the illumination on other surface points, is largely a local effect. This means two things. First, subsurface scattering within one object will have little effect on the appearance of another object; the influence between different objects can be well described by the reflectance values on their surfaces only. Therefore unlike the situation addressed by radiosity methods where every patch has an effect on every other patch in the scene, subsurface scattering only has effect within an object. Second, even within the same object, subsurface scattering due to light entering from one surface point will have little effect on another surface point on the same object if the distance between the two points is large. This property is due to the exponential falloff of light intensity due to absorption and scattering.

Based on the above observations, we conclude that in order to model the appearance of a surface point due to subsurface scattering to a first approximation, we only need to know its local neighborhood and associated material properties.

3.2 Approximate Volume Scattering Effects

The complete BSSRDF model S is a sum of single scattering $S^{(1)}$ and multiple scattering S_d terms [Jensen et al. 2001]:

$$S(x_i, \vec{\omega_i}; x_o, \vec{\omega_o}) = S^{(1)}(x_i, \vec{\omega_i}; x_o, \vec{\omega_o}) + S_d(x_i, \vec{\omega_i}; x_o, \vec{\omega_o})$$

Jensen and Buhler [2002] have shown that for highly-scattering translucent materials, multiple scattering alone can sufficiently simulate the visual appearance. We follow their results and focus here on modelling the multiple scattering effects.

Jensen *et al.* [2001] have shown that the dipole method is a good approximation for volumetric effects due to subsurface multiple scattering. The diffusion term of the BSSRDF for subsurface multiple scattering can be well approximated by the following formula [Jensen et al. 2001]:

$$S_d(x_i, \vec{\omega_i}; x_o, \vec{\omega_o}) = \frac{1}{\pi} F_t(\eta, \vec{\omega_i}) R_d(\|x_i - x_o\|) F_t(\eta, \vec{\omega_o})$$

where F_t is the Fresnel transmission term and R_d is the single dipole approximation for multiple scattering [Jensen et al. 2001; Jensen and Buhler 2002]:

$$\begin{aligned} R_d(r) &= -D \frac{(\vec{n} \cdot \vec{\nabla} \phi(x_s))}{d\Phi_i} \\ &= \frac{\alpha'}{4\pi} [z_r(\sigma_{tr} + \frac{1}{d_r}) \frac{e^{-\sigma_{tr} d_r}}{d_r^2} + z_v(\sigma_{tr} + \frac{1}{d_v}) \frac{e^{-\sigma_{tr} d_v}}{d_v^2}] \end{aligned}$$

where D is the diffusion constant, ϕ is the radiant fluence, Φ_i is the incident flux, α' is the reduced albedo, σ_{tr} is the effective transport coefficient,coefficient, z_r and z_v are the distance from the dipole lights to the surface, d_r is the distance from x to the real source, and d_v is the distance from x to the virtual source. From this equation, we see that if the scattering property of the material is homogeneous, i.e., the scattering cross-sections are constant, then the formula relates the reflectance at one surface point to the incident flux at other surface points. Since the subsurface scattering has a limited effective range, we can obtain the reflectance of a surface point due to multiple scattering by integrating flux incident at points within a certain distance.

We note that the dipole factor, $R_d(r)$, only depends on the distance between the two points and decays exponentially with the distance. This allows us to compute the multiple scattering contribution from the *neighborhood* of each

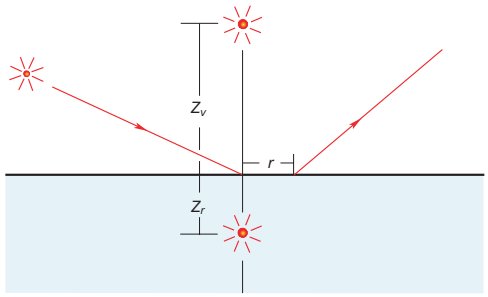


Figure 3: Dipole Approximation of Multiple Scattering (based on [Jensen *et al.*,2001])

vertex during the pre-processing stage. The neighborhood of a vertex x_o, $N(x_o)$, is defined to include all vertices x_i of that object that lie within the effective scattering range from x_o. Every such neighboring vertex x_i is then considered to represent a small surface area whose size can be approximately defined. We then assign the integral of $R_d(\|x_i - x_o\|)$ over this small surface area as the contribution to the multiple scattering at x_o due to x_i and append this information to x_o's list of multiple scattering contributors. Then at rendering time, once we have the $F_t(\eta, \overrightarrow{\omega_i})$ and $F_t(\eta, \overrightarrow{\omega_o})$ from the local illumination computation, the contribution of point x_i to x_o due to subsurface scattering is just the multiplication of $F_t(\eta, \overrightarrow{\omega_i})$ with $F_t(\eta, \overrightarrow{\omega_o})$ and the pre-computed $R_d(\|x_i - x_o\|)$ factor of x_i from x_o's neighborhood list.

3.3 Run-Time Two-Pass Local Illumination Model

Traditionally, we compute the outgoing radiance from a surface point according to the lighting direction, the surface normal, and the viewing direction. We accommodate subsurface scattering effects into the local illumination model by extending the model into a run-time two-pass one. The first pass generates the reflection and transmission radiance at each surface point as if there is no subsurface scattering, using the Fresnel terms for reflection or transmission.

After we compute the illumination at all surface points, we come to the second pass, i.e., the bleeding pass. During this pass, we combine the surface reflection with the subsurface scattering to get the total radiance at the exterior surface points according to the multiple scattering factors given in Section 3.2, using each point's weighted contributions from its neighbors. This bleeding pass adds the subsurface reflection and transmission effects on the surface.

We note that Jensen and Buhler [2002] have also used a two-pass process to model the multiple scattering of translucent materials. The two approaches, though similar, are still quite different. First, our run-time two-pass scheme is flexible, and can be simplified to a run-time single-pass using local illumination model to improve the efficiency, which we describe in the next section. Second, our pre-computed integrals are built at the preprocessing stage, so bleeding the neighboring effects due to scattering in the second pass is quite efficient, instead of traversing the hierarchical N-body data structure for each frame as in [Jensen and Buhler 2002]. These differences, when combined with the acceleration techniques we will describe in the following section, enable our approach to achieve interactive frame rates for simulating subsurface scattering effects. However, as we stated earlier, our model is an appearance-driven one using local illumination, so its accuracy sometimes can not exactly match the one proposed in [Jensen and Buhler 2002].

4 Improving Efficiency

As mentioned in previous sections, we build the neighborhood information during a preprocessing phase. At runtime, after computing the light flux of reflection and transmission at each vertex, we use the computed neighborhood information to do the bleeding. This approach has $O(N^2)$ complexity, where N is the number of surface points, assuming the size of the object and the scattering properties remain constant. This is due to the fact that the number of vertices at which we have to perform the bleeding step is N and the neighborhood size is proportional to the surface point density, which in turn is proportional to the number of surface points. Instead of building a hierarchical $O(NlogN)$ data structure to solve the inherent $O(N^2)$ complexity problem as done in [Jensen and Buhler 2002], we propose a quantized light source scheme to merge the two stages of our lighting process into a single stage and further improve the efficiency of our algorithm. We thus reduce the complexity of our run-time algorithm to $O(N)$ with quite small constant factors.

4.1 Quantized Light Sources

As we have mentioned earlier, due to the roughness of real surfaces, each surface point in the neighborhood of another surface point represents a small area on the surface. Therefore, we can make further simplifications to reduce the complexity of the algorithm.

If the light source is directional, then the subsurface scattering contribution to the appearance of a surface point can be preprocessed as the following:

$$
\begin{aligned}
& L_o(x_o, \overrightarrow{\omega_o}) \\
= & \int_A S(x_i, \overrightarrow{\omega_i}; x_o, \overrightarrow{\omega_o}) L_i(x_i, \overrightarrow{\omega_i})(\overrightarrow{n_i} \cdot \overrightarrow{\omega_i}) dA \\
\approx & \int_A S_d(x_i, \overrightarrow{\omega_i}; x_o, \overrightarrow{\omega_o}) L_i(x_i, \overrightarrow{\omega_i})(\overrightarrow{n_i} \cdot \overrightarrow{\omega_i}) dA \\
= & \int_A F_t(\eta, \overrightarrow{\omega_i})[\frac{1}{\pi} R_d(\|x_i - x_o\|)] F_t(\eta, \overrightarrow{\omega_o}) \cdot L_i(x_i, \overrightarrow{\omega_i})(\overrightarrow{n_i} \cdot \overrightarrow{\omega_i}) dA \\
= & \{\int_A F_t(\eta, \overrightarrow{\omega_i})(\frac{1}{\pi} R_d) L_i(x_i, \overrightarrow{\omega_i}) \overrightarrow{n_i} dA\} \cdot \overrightarrow{\omega_i} \cdot F_t(\eta, \overrightarrow{\omega_o}) \\
\equiv & \overrightarrow{Q}(\eta, x_o, \overrightarrow{\omega_i}) \cdot \overrightarrow{\omega_i} \cdot F_t(\eta, \overrightarrow{\omega_o})
\end{aligned}
$$

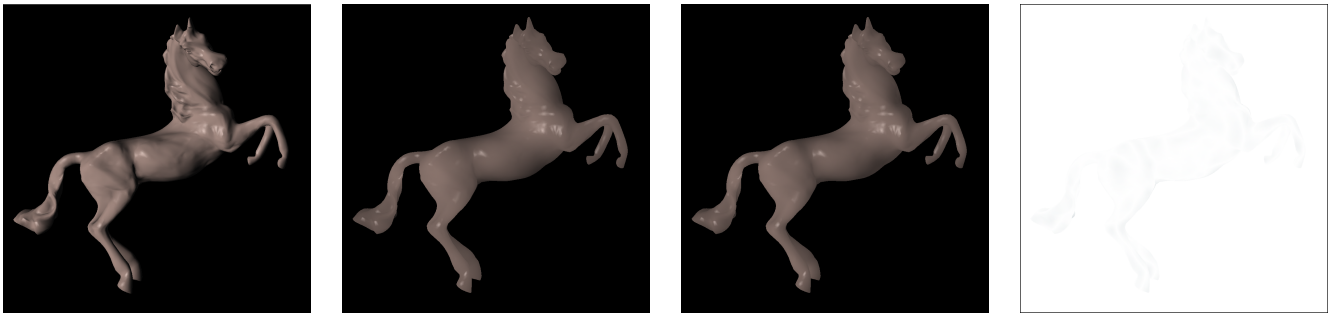
This means we can pre-compute the integral $\overrightarrow{Q}(\eta, x_o, \overrightarrow{\omega_i})$ for the scattering factor during the preprocessing stage, and at run time perform the dot-product and multiplication operations.

Due to the discrete nature of the input mesh geometry, the integral above can be expressed as a summation over all the vertices by the following expression:

$$
\overrightarrow{Q}(\eta, x_o, \overrightarrow{\omega_i}) = \sum_{x_i \in N(x_o)} F_t(\eta, \overrightarrow{\omega_i})(\frac{1}{\pi} R_d) L_i(x_i, \overrightarrow{\omega_i}) \overrightarrow{n_i} \Delta A(x_i)
$$

where the summation is over all the neighboring vertices. $\Delta A(x_i)$ is the area represented by vertex x_i, which is a constant if the vertices are distributed uniformly as in [Jensen and Buhler 2002]. For non-uniformly distributed vertices, we can either resample the geometry, or use one third of the total area of the triangles sharing the vertex as ΔA at the vertex. So we actually pre-compute the summation $\overrightarrow{Q}(\eta, x_o, \overrightarrow{\omega_i})$ for each vertex. Note, if a vertex at x_i in the neighborhood of x_o is in shadow, then it will not contribute to $\overrightarrow{Q}(\eta, x_o, \overrightarrow{\omega_i})$, because x_i receives no direct irradiance from the light source. The summation will not be affected by the presence of shadow on x_o, though. We use a technique similar to shadow map to determine if a vertex is in shadow. We first generate a depth image of the scene as seen by the light source. Then for each vertex, we transform it into light space and compare its depth value against the value on the depth image. If the depth value of the vertex is bigger, the vertex is in the shadow.

To cover all the possible light source directions, we precompute a set of uniformly distributed directional light sources. For each light source j within the set, we compute the neighborhood integral $\overrightarrow{Q_j}$ at each vertex during the preprocessing stage. At run time, we approximate the scattering integral $\overrightarrow{Q}$ using quaternion-based vector interpolation [Pletinckx 1989] of the integrals of its four closest $\overrightarrow{Q_j}$'s in the set. We compute the dot-product of the interpolated scattering integral $\overrightarrow{Q}$ with the real light source direction. This interpolation is similar to the normal interpolation scheme used in Phong shading, though the quaternion interpolation gives a more accurate result and avoids a vector re-normalization step. Another possibility is to interpolate



(a) No scattering (b) Scattering using $\overrightarrow{Q}$ (c) Scattering using q (d) Difference of (b) and (c)

Figure 4. Subsurface scattering using $\overrightarrow{Q}$ and q on the horse model (14,521 vertices)

the computed dot-products, which is similar to the interpolation used in the Gouraud shading algorithm. Thus we can reduce the memory usage by just storing a scalar dot-product value $q(\eta, x_o, \overrightarrow{\omega_i})$ instead of a vector $\overrightarrow{Q}(\eta, x_o, \overrightarrow{\omega_i})$:

$$
\begin{aligned}
& L_o(x_o, \overrightarrow{\omega_o}) \\
= & \left\{ \sum_{x_i \in N(x_o)} F_t(\eta, \overrightarrow{\omega_i})(\frac{1}{\pi} R_d) L_i(x_i, \overrightarrow{\omega_i})(\overrightarrow{n_i} \cdot \overrightarrow{\omega_i}) \Delta A(x_i) \right\} \cdot F_t(\eta, \overrightarrow{\omega_o}) \\
\equiv & \; q(\eta, x_o, \overrightarrow{\omega_i}) \cdot F_t(\eta, \overrightarrow{\omega_o})
\end{aligned}
$$

Our experiments show that a set of about 200 light sources can give robust and continuous appearance of subsurface scattering effect. We have also observed that for our models, the visual difference between using $\overrightarrow{Q}$ and q is insignificant. This can be attributed to the diffuse nature of the subsurface scattering. Hence we are currently using the pre-computed scalar dot-products. Figure 4(b) shows the image generated using $\overrightarrow{Q}$ on the horse model, and Figure 4(c) shows the image generated using q on the same model. The difference image is shown in Figure 4(d). The image space root-mean-square error between Figure 4(b) and 4(c) is 5.26×10^{-3}.

During the rendering of the scene at run-time, we combine the scattering effects with the direct on-surface-reflected light to give the final appearance of each vertex. As an example, for a light source in direction $\overrightarrow{\omega_i}$, the scattering amount for vertex x_o along viewing direction $\overrightarrow{\omega_o}$ will be $F_t(\eta, \overrightarrow{\omega_o})$ multiplied with the pre-computed factor $q(\eta, x_o, \overrightarrow{\omega_i})$, and scaled by this light source's actual intensity. While for the direct on-surface-reflected light, we can simply use a local illumination model.

The light flux at a vertex on the surface due to direct reflection and subsurface scattering can now be computed at the same time under a local illumination model. Thus with the pre-computed integral, the run-time two-pass algorithm we suggested before now becomes a run-time single-pass algorithm. Furthermore, this pre-computed integral scheme also indicates that the computation of the scattering effect on a vertex only depends on the size of the light set we have selected, which is a constant, and not related to the surface point density. So the complexity of the algorithm becomes $O(N)$, instead of $O(N^2)$, where N is the number of vertices, which is proportional to the surface point density for a given geometric shape. The original $O(N^2)$ complexity is because we have to compute the scattering for all the N vertices, and for each vertex, the neighbors that need to be considered is also proportional to N. The run-time computational complexity of our algorithm stays $O(N)$ even if the subsurface scattering property increases and results in a larger effective range with more neighborhood surface points that have to be considered for each vertex. This can happen when either the translucency of the material increases or the physical size of the object decreases. This is because all of these can be pre-computed.

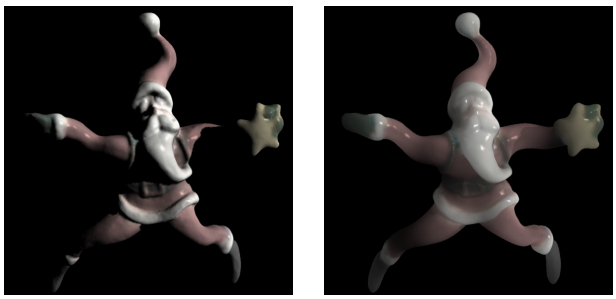
This directional quantization scheme can also be extended to include point light sources. We can add one more dimension to the interpolation, i.e., we quantize the distance of the light source to the object along with the quantization of its direction. Then we can quadralinearly interpolate 16 nearest neighbors to get an $O(N)$ complexity algorithm for directional and point light sources.

Here we limit ourselves for local illumination, so we ignore the multiple on-surface inter-reflections between vertices during the computation of the pre-computed integral of $q(\eta, x_o, \overrightarrow{\omega_i})$. If we use ray-tracing in the preprocessing stage, we can incorporate it in our algorithm and get more accurate $q(\eta, x_o, \overrightarrow{\omega_i})$.

5 Results and Discussion

We demonstrate our simplified subsurface scattering model by showing the visual effects we can generate on polygonal datasets. The results are summarized in Table 1 and in Figures 1, 5– 7, and 9. The results presented here have been obtained from our implementation running on a 2GHz Pentium 4 PC running Windows 2000 with a nVIDIA Geforce3 graphics card.

Table 1 shows the timing for generating the images in Figures 1, 5– 7, and 9. We compute the BSSRDF for all the sample vertices as in Table 1. The color of the vertex, both subsurface scattered and non-scattered, is computed on the CPU for a fair comparison. As one can see, the images are generated by our scattering model within a few tenths of a second, which is about one to two orders of magnitude faster than the previous fastest method [Jensen and Buhler 2002]. Our algorithm has an effective $O(N)$ complexity, where N is the number of vertices. From this table, one can see that our algorithm is only about 30% slower compared with local illumination without subsurface scattering. This



(a) Without Scattering (17.2 fps) (b) With Scattering (12.7 fps)

Figure 5: Santa without and with subsurface scattering (75,781 vertices, 1024×1024 pixels)

Model Name	No. of Vertices	No. of Triangles	Frame rate(fps)	
			with scattering	without scattering
Horse	14,521	29,054	69.3	98.2
Venus	42,656	90,044	23.4	33.9
Santa	75,781	151,558	12.7	17.2
Teapot	150,510	292,168	7.5	10.6
Dragon	437,645	871,414	2.4	3.4
Buddha	543,652	1,087,716	1.9	2.8

Table 1: Total rendering times for our approach

Figure 6: Utah teapot without scattering (10.6 fps, 150,510 vertices, 1024×624 **pixels)**

small overhead will give most applications the opportunity to include the subsurface scattering effects for more photo-realistic rendering without sacrificing the interactive frame rates.

To validate our algorithm, we have run our algorithm on the Utah teapot and compared the result with the one generated by Jensen and Buhler [2002](Figure 8) under similar viewing and lighting conditions. Figure 6 is the teapot without subsurface scattering, Figure 7 is the one with the subsurface scattering using our algorithm. Our approach gives similar effects while improving the speed from 7 seconds per frame as reported in [Jensen and Buhler 2002] to 7.5 frames per second, which results in a factor of more than 50 speedup. Figures 6, 7, and 8 each has $150K$ sample points. The computation of the pre-computed integral $q(\eta, x_o, \vec{\omega_i})$ for all the vertices with 200 light sources takes about 40 minutes for the teapot model.

A video sequence of the subsurface scattered teapot by our algorithm is included in the accompanying video. There the translucency can be observed clearly from the neighborhood bleeding and soft shadow effects on the appearance. Most of the shadows thus produced are very soft due to the subsurface scattering, though in a few places, one will observe sharp shadows. Sharp shadows occur when a specular highlight overlaps with the shadow region. This is because specular reflection is by definition a superficial effect, so the light taking part in it does not scatter inside the material.

For the Venus model we used the Perlin noise function [Ebert et al. 1998; Perlin 1985] to generate the marble texture. Here we have made the assumption the marble texture is on the surface, and will affect both x_i and x_o. Figures 1 and 9 show how the object will appear if either its size shrinks or its material property changes to allow greater subsurface scattering.

We should also mention that for a set of 200 lights, we need to store 200 integrals per vertex. Instead of storing a floating-point value per integral, we store a normalized unsigned byte value to serve as an index to a lookup table. Thus we need 200 bytes of extra storage per vertex. Vertices

Figure 7: Utah teapot with scattering by our algorithm (7.5 fps, 150,510 vertices, 1024×624 **pixels)**

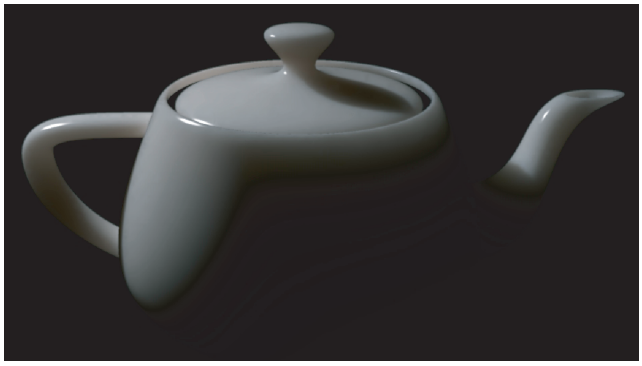


Figure 8: Image from paper [Jensen and Buhler 2002] (7 seconds/frame)

normally need three numbers each for position, normal direction, diffuse color, specular color, and the final blended color. If we assume floating-point numbers to store these values, we will need 60 bytes. So the extra storage needed for the pre-computed integral will be less than four times of the storage for regular vertex structure. The number can be reduced though. Due to the diffuse-like nature of the subsurface scattering effects, we expect that the spherical harmonic functions as used by [Ramamoorthi and Hanrahan 2001; Ramamoorthi and Hanrahan 2002; Sloan et al. 2002] can be applied effectively to compress the directional integrals. If a total of 25 spherical basis functions are needed, and each is represented by a normalized short coefficient, then the extra storage needed per vertex will be 50 bytes, comparable to the size of a regular vertex. We plan to explore this issue in the future.

6 Conclusions

In this paper we show that the subsurface scattering can be simulated efficiently within the local illumination framework. The algorithm delivers about one to two orders of magnitude speedup over the previous approaches simulating the subsurface scattering effect. The results capture the most important effects of subsurface scattering, such as neighborhood bleeding and smooth transitions between regions separated by sharp edges. Our method has an efficient $O(N)$ run-time complexity and provides a possible approximation of subsurface scattering for applications which need to maintain interactivity without sacrificing the realistic appearance for translucent materials. Our approach, by a little modification, can also be incorporated into shadow

(a) Without Scattering (33.9 fps)

(b) With scattering (23.4 fps) (10% vertices in $N(x_o)$)

(c) With scattering (23.4 fps) (20% vertices in $N(x_o)$)

(d) With scattering (23.4 fps) (30% vertices in $N(x_o)$)

Figure 9: Rendering the Venus model with subsurface scattering increasing from left to right (42,656 vertices, 480×1280 **pixels)**

algorithms to generate soft shadow effects.

7 Acknowledgements

We would like to thank Henrik Wann Jensen, Pat Hanrahan, and Juan Buhler for providing the lighting and material parameters for generating the images in this paper. We would also like to acknowledge the reviewers for their very detailed and constructive comments which have led to a much better presentation of our results. This work has been supported in part by the NSF grants: IIS-00-81847, and ACR-98-12572.

References

Ashikhmin, M., Premoze, S., and Shirley, P. 2000. A microfacet-based BRDF generator. In *SIGGRAPH 2000, Computer Graphics Proceedings*, K. Akeley, Ed., Annual Conference Series, 65–74.

Bahar, E., and Chakrabarti, S. 1987. Full-wave theory applied to computer-aided graphics for 3D objects. *IEEE Computer Graphics and Applications 7*, 7 (July), 46–60.

Blinn, J. F. 1977. Models of light reflection for computer graphics. *Computer Graphics 11*, 2, 192–198.

Cabral, B., Max, N., and Springmeyer, R. 1987. Bidirectional reflection functions from surface bump maps. In *Computer Graphics (SIGGRAPH '87 Proceedings)*, M. C. Stone, Ed., vol. 21(4), 273–281.

Cook, R. L., and Torrance, K. E. 1981. A reflectance model for computer graphics. In *Computer Graphics (SIGGRAPH '81 Proceedings)*, vol. 15(3), 307–316.

Debevec, P., Hawkins, T., Tchou, C., Duiker, H., Sarokin, W., and Sagar, M. 2000. Acquiring the reflectance field of a human face. In *SIGGRAPH 2000, Computer Graphics Proceedings*, K. Akeley, Ed., Annual Conference Series, 145–156.

Dorsey, J., Edelman, A., Legakis, J., Jensen, H. W., and Pedersen, H. K. 1999. Modeling and rendering of weathered stone. In *SIGGRAPH 99, Computer Graphics Proceedings*, A. Rockwood, Ed., Annual Conference Series, 225–234.

Ebert, D., Musgrave, F. K., Peachey, D., Perlin, K., and Worley, S. 1998. *Texturing and Modeling*, second ed. AP Professional.

Greenberg, D. P., Torrance, K. E., Shirley, P., Arvo, J., Ferwerda, J. A., Pattanaik, S., Lafortune, E. P. F., Walter, B., Foo, S.-C., and Trumbore, B. 1997. A framework for realistic image synthesis. In *SIGGRAPH 97, Computer Graphics Proceedings*, T. Whitted, Ed., Annual Conference Series, 477–494.

Hanrahan, P., and Krueger, W. 1993. Reflection from layered surfaces due to subsurface scattering. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, J. T. Kajiya, Ed., vol. 27, 165–174.

He, X. D., Torrance, K. E., Sillion, F. X., and Greenberg, D. P. 1991. A comprehensive physical model for light reflection. In *Computer Graphics (SIGGRAPH '91 Proceedings)*, T. W. Sederberg, Ed., vol. 25(4), 175–186.

Jensen, H. W., and Buhler, J. 2002. A rapid hierarchical rendering technique for translucent materials. In *SIGGRAPH 2002, Computer Graphics Proceedings*, J. F. Hughes, Ed., Annual Conference Series, 576–581.

Jensen, H. W., Legakis, J., and Dorsey, J. 1999. Rendering of wet materials. In *Rendering Techniques '99*, Springer Verlag, D. Lischinski and G. W. Larsoi, Eds., 273–282.

Jensen, H. W., Marschner, S., Levoy, M., and Hanrahan, P. 2001. A practical model for subsurface light transport. In *SIGGRAPH*

2001, Computer Graphics Proceedings, E. Fiume, Ed., Annual Conference Series, 511–518.

Kajiya, J. T. 1985. Anisotropic reflection models. In *Computer Graphics (SIGGRAPH '85 Proceedings)*, vol. 15(3), 15–21.

Koenderink, J. J., and van Doorn, A. J. 2001. Shading in the case of translucent objects. In *Proceedings of SPIE*, vol. 4299, 312–320.

Lensch, H., Gosele, M., Bekaert, P., Kautz, J., Magnor, M., Lang, J., and Seidel, H.-P. 2002. Interactive rendering of translucent objects. In *Proc. IEEE Pacific Graphics 2002*, 214–224.

Marschner, S. R., Westin, S. H., Lafortune, E. P. F., Torrance, K. E., and Greenberg, D. P. 1999. Image-based BRDF measurement including human skin. In *Eurographics Workshop on Rendering*, 139–152.

Perlin, K. 1985. An image synthesizer. In *Computer Graphics (SIGGRAPH '85 Proceedings)*, B. A. Barsky, Ed., vol. 19(3), 287–296.

Pharr, M., and Hanrahan, P. 2000. Monte carlo evaluation of non-linear scattering equations for subsurface reflection. In *SIGGRAPH 2000, Computer Graphics Procedings*, K. Akeley, Ed., Annual Conference Series, 75–84.

Phong, B.-T. 1975. Illumination for computer generated pictures. *CACM June 1975 18*, 6, 311–317.

Pletinckx, D. 1989. Quaternion calculus as a basic tool in computer graphics. *The Visual Computer 5*, 1/2 (Mar.), 2–13.

Ramamoorthi, R., and Hanrahan, P. 2001. A signal-processing framework for inverse rendering. In *SIGGRAPH 2001, Computer Graphics Proceedings*, E. Fiume, Ed., Annual Conference Series, 117–128.

Ramamoorthi, R., and Hanrahan, P. 2002. Frequency space environment map rendering. In *SIGGRAPH 2002, Computer Graphics Proceedings*, J. Hughes, Ed., Annual Conference Series, 517–526.

Sato, Y., Wheeler, M. D., and Ikeuchi, K. 1997. Object shape and reflectance modeling from observation. In *SIGGRAPH 97, Computer Graphics Proceedings*, T. Whitted, Ed., Annual Conference Series, 379–388.

Sloan, P., Kautz, J., and Snyder, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH 2002, Computer Graphics Proceedings*, J. Hughes, Ed., Annual Conference Series, 527–536.

Stam, J. 1999. Diffraction shaders. In *SIGGARPH 99, Computer Graphics Proceedings*, A. Rockwood, Ed., Annual Conference Series, 101–110.

Stam, J. 2001. An illumination model for a skin layer bounded by rough surfaces. In *Rendering Techniques '01*, Springer Verlag, S. J. Gortler and K. Myszkowski, Eds., 39–52.

Sun, Y., Fracchia, F. D., Drew, M. S., and Calvert, T. W. 2000. Rendering iridescent colors of optical disks. In *Rendering Techniques '00*, 341–352.

Ward, G. J. 1992. Measuring and modeling anisotropic reflection. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, E. E. Catmull, Ed., vol. 26(2), 265–272.

Yu, Y., Debevec, P., Malik, J., and Hawkins, T. 1999. Inverse global illumination: recovering reflectance models of real scenes from photographs. In *SIGGARPH 99, Computer Graphics Proceedings*, A. Rockwood, Ed., Annual Conference Series, 215–224.

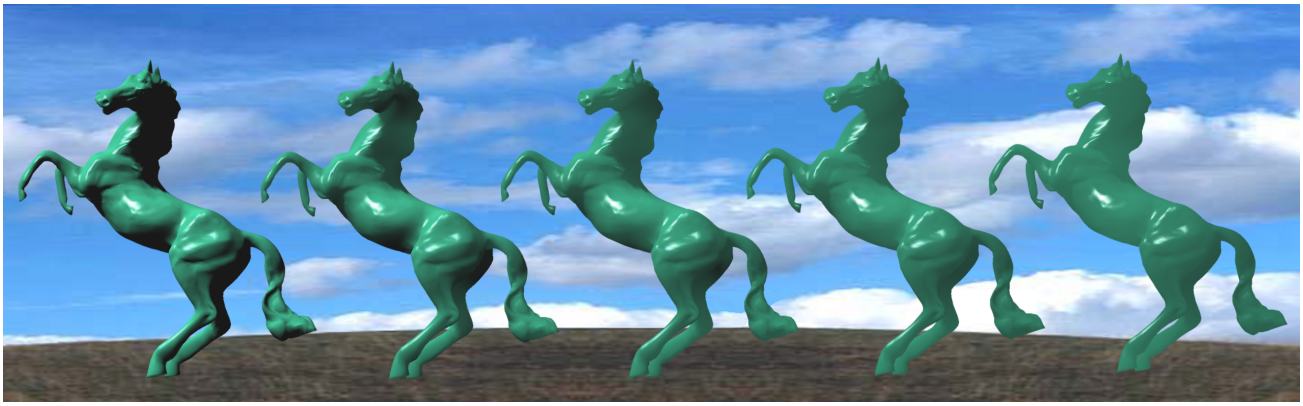


Figure 1: Different levels of Subsurface Scattering (increasing from left to right) on the model of a horse (14,521 vertices)

6. Without Scattering (10.6 fps, 1024×624 pixels)

7. Scattering by our algorithm (7.5 fps, 1024×624 pixels)

8. Image from paper [Jensen and Buhler 2002] (7 seconds/frame)

Figure 6, 7, 8: Utah teapot without and with subsurface scattering (150,510 vertices)

(a) Without Scattering (33.9 fps)

(b) With scattering (23.4 fps) (10% vertices in $N(x_o)$)

(c) With scattering (23.4 fps) (20% vertices in $N(x_o)$)

(d) With scattering (23.4 fps) (30% vertices in $N(x_o)$)

Figure 10: Rendering the Venus model with subsurface scattering increasing from left to right (42,656 vertices, 480×1280 pixels)