SSLPV: Subsurface Light Propagation Volumes

Jesper Børlum Brian Bunch Christensen Thomas Kim Kjeldsen Peter Trier Mikkelsen
Karsten Østergaard Noe Jens Rimestad Jesper Mosegaard*

Alexandra Institute, Denmark

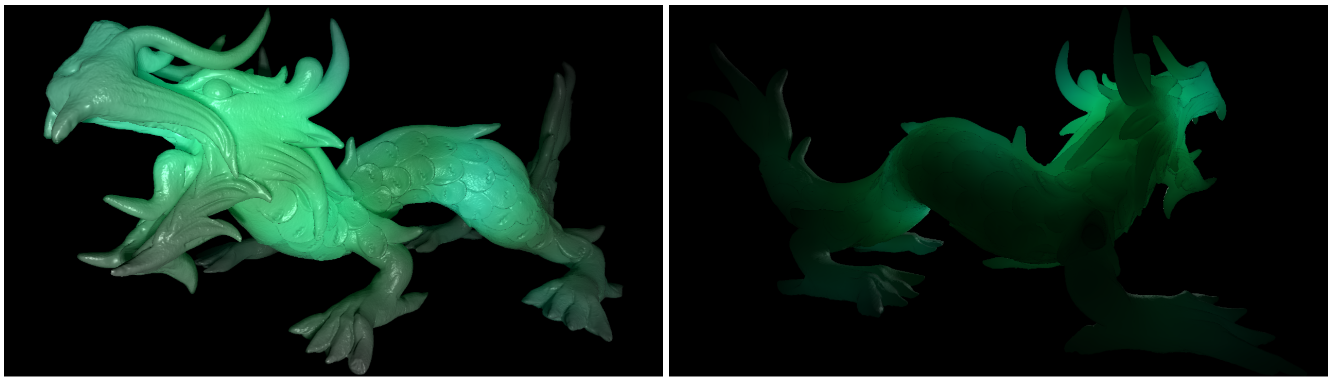

Figure 1: *Our method renders the Stanford Asian Dragon (7.2 M triangles) with heterogeneous materials at 43 frames per second.*

Abstract

This paper presents the Subsurface Light Propagation Volume (SSLPV) method for real-time approximation of subsurface scattering effects in dynamic scenes with changing mesh topology and lighting. SSLPV extends the Light Propagation Volume (LPV) technique for indirect illumination in video games. We introduce a new consistent method for injecting flux from point light sources into an LPV grid, a new rendering method which consistently converts light intensity stored in an LPV grid into incident radiance, as well as a model for light scattering and absorption inside heterogeneous materials. Our scheme does not require any precomputation and handles arbitrarily deforming meshes. We show that SSLPV provides visually pleasing results in real-time at the expense of a few milliseconds of added rendering time.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Radiosity; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Shading; I.3.3 [Computer Graphics]: Picture/Image Generation—Display Algorithms

Keywords: subsurface scattering, real-time rendering, Light Propagation Volumes

1 Introduction

Subsurface scattering is an important visual phenomenon for many categories of translucent materials, such as marble, milk, skin, and teeth. Formally, all non-metallic materials scatter incoming light inside the object such that the light is eventually either absorbed or leaves the object at a different location, in a different direction, and with a different spectral composition. The visual effect is naturally more evident in some materials than others. In both computer games and feature films, recent advances in the field of subsurface scattering have enabled more believable characters due to our sensitivity to the exact appearance of *e.g.* skin. However, the simulation of the complex phenomenon of subsurface scattering remains challenging; the outgoing radiance at any point on a surface essentially depends on the incoming radiance at all points on the entire surface as well as the scattering and absorption properties of the medium. A complete Monte Carlo solution to a general description of light transport in participating media is prohibitively slow, hence, subsurface scattering is often modelled by approximate methods such as the diffusion approximation or the dipole approximation [Jensen et al. 2001].

In the last decade, much research has gone into solving the diffusion equation efficiently as well as achieving similar effects through other approximations. In recent years, the raw computational power of the GPUs has made real-time subsurface scattering feasible through various simplifications, approximations and precomputations. We present *Subsurface Light Propagation Volumes* (SSLPV) as a novel approach to real-time rendering of subsurface scattering. Unlike existing approaches, our method supports both heterogeneous material properties and changes in object topology without any precomputations.

Our method is essentially an extension to the Light Propagation Volume (LPV) method [Kaplanyan and Dachsbacher 2010; Kaplanyan et al. 2011], which was introduced as an efficient technique for real-time indirect illumination. As a secondary result, Kaplanyan and Dachsbacher [2010] note that single scattering from participating media can be included in the model quite easily. The main purpose of the present work is to extend the LPV model to account for a broader range of phenomena in participating media, in particular subsurface scattering. We demonstrate that the significant numerical diffusion of the LPV method plays well together with the simulation of highly scattering materials. However, we do not expect the SSLPV method to be suitable for light transport in highly translucent materials. Our contributions are the following:

- A novel LPV based method for subsurface scattering.

*{jesper.borlum, brian.bunch, thomas.kjeldsen, peter.trier, karsten.noe, jens.rimestad, jesper.mosegaard}@alexandra.dk

©ACM, 2011. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version will be published in the HPG 2011 proceedings.

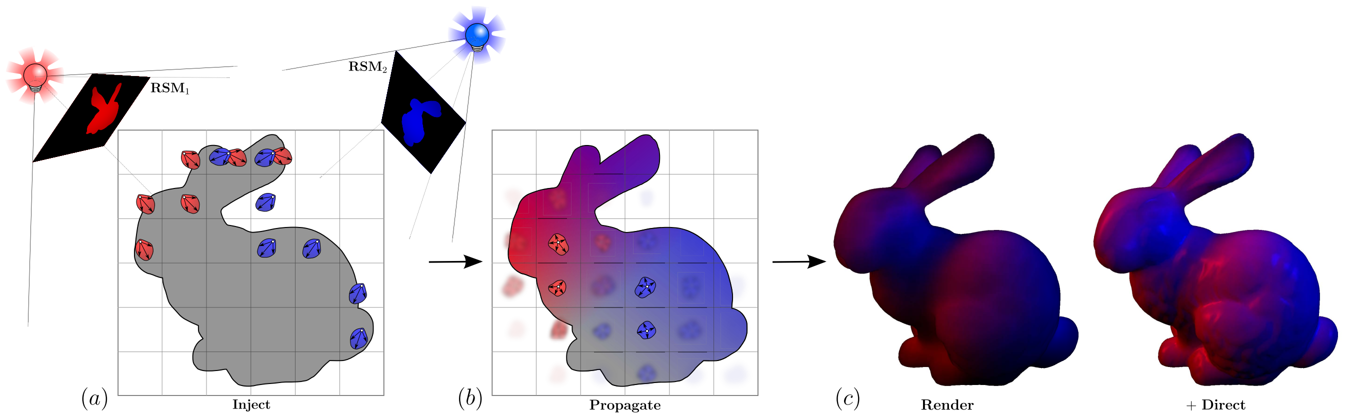

Figure 2: *Conceptual overview of the SSLPV method. (a) A reflective shadow map is created for each light source. The shadow maps are used to create virtual point light sources in the grid cells lying on the boundary of the object. (b) Light is propagated between neighbouring cells in the propagation step. Propagation is repeated iteratively until all light has escaped the grid cells within the object interior. (c) Light transported through the object surface is collected and used for rendering subsurface scattering. Finally, direct reflections are added.*

- Support for arbitrarily deforming meshes.
- A heuristic model of scattering and absorption effects in LPV.
- A consistent method for rendering using LPV grids and for injecting flux from point lights into LPV grids.

2 Related Work

Multiple scattering is often modelled by the diffusion approximation [Kajiya and Herzen 1984; Stam 1995]. Stam [1995] originally proposed a grid based numerical scheme to solve the diffusion equation for heterogeneous materials. Later Jensen *et al.* [2001] devised a dipole point source diffusion approximation model which formed the basis for many of the following techniques. Jensen and Buhler [2002] improved on the offline rendering times by precomputing an acceleration structure for neighbour vertex lookup and split the light computation into two passes.

Several authors have worked on achieving real-time subsurface scattering. Sloan *et al.* [2003] precompute low frequency light scattering for all vertices and encode the factors into spherical harmonics. Multiple techniques [Hao and Varshney 2004; Lensch et al. 2003; Carr et al. 2003] are based on variants of precomputed vertex-to-vertex diffusion throughput factors. Mertens *et al.* [2003] precompute a hierarchical mesh data structure to reduce the complexity of integrating over the mesh surface. Wang *et al.* [2008] demonstrate how to precompute 1D homogeneous basic scattering profiles and linearly combine them on the fly to achieve real-time manipulation of optical properties. More recently, discretisation on a tetrahedral mesh and a clever GPU implementation were used to solve the diffusion equation in real-time [Wang et al. 2010]. Although rendering is real-time in the work by Wang *et al.* [2008; 2010], they require a significant amount of time for precomputation based on the geometry. Hence, objects that undergo topological changes cannot be handled in real-time.

Some methods do not require preprocessing of light propagation although some rely on the generation of a texture map for each rendered mesh. François *et al.* [2008] use relief texture mapping and a ray marching algorithm for gathering interior single scattering contributions along the view direction. Multiple authors have used translucent shadow maps (TSM) [Dachsbacher and Stamminger 2003] for approximating global subsurface scattering in real-time. TSMs are rendered from each light source and then filtered with a depth dependent kernel. d'Eon *et al.* [2007] use filtering of illumination textures to model local scattering and TSMs for global effects. A similar combined approach is taken by Chang *et al.* [2008] using importance sampling instead of a smoothing kernel. Optimisations of this approach were proposed by Jimenez *et al.* [2008] and Hable *et al.* [2009]. Jimenez *et al.* [2009] further improved performance of the method by doing computations in screen space.

Mertens *et al.* [2005] also proposed a screen space method for rendering *local* subsurface scattering effects in real time. Their work uses importance sampling of an irradiance map which is rendered in either screen space or texture space. Note that the latter approach requires a texture parameterisation. The screen space methods above require no precomputation of a texture parameterisation and they can be used with arbitrary changes of topology. Although a screen space method in nature cannot handle global effects such as light bleeding through, it can be combined with TSMs to approximate such effects. The advantage of our method compared to this approach is the ability to handle both local and large-scale scattering effects while allowing heterogeneous material properties.

The Lattice-Boltzmann lighting model is somewhat similar to our method in terms of implementation [Geist et al. 2004]. This method propagates photon densities in a grid according to a local set of collision rules. Recent GPU based optimisations have accelerated the Lattice-Boltzmann method significantly, however, real-time frame rates have not yet been achieved [Geist and Westall 2011].

3 Method

This section outlines the theoretical foundation for our method in general terms. In Sec. 4 we describe how to apply the method efficiently. Figure 2 shows the conceptual overview of our SSLPV method. The scene geometry is enclosed by a regular lattice, the *LPV grid*, on which point light intensities representing the subsurface lighting contribution are computed and stored. The main steps shown in Fig. 2, injection, propagation, and rendering will be described in detail below.

3.1 Injection

We inject light into the scene by a slight modification of the scheme described by Kaplanyan and Dachsbacher [2010]. They first generate a reflective shadow map (RSM) for each light source [Dachs-

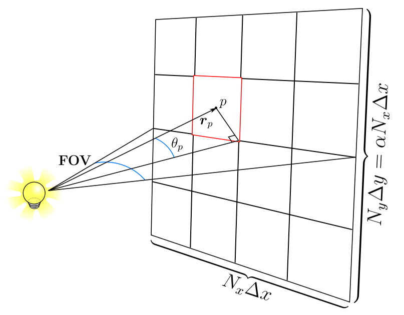

Figure 3: *The shadow map for a point light source is characterised by the map resolution* (N_x, N_y)*, field of view* FOV*, aspect ratio* α*, and the normal vector.*

bacher and Stamminger 2005]. For each texel p in the RSM, they store the world space position $\boldsymbol{x}_p$ and the normal $\boldsymbol{n}_p$. Furthermore, because they use the RSM to generate the directional intensity distribution of reflected light, they also store the *reflected* flux. However, the main focus in our work is subsurface scattering, and therefore we will use a shadow map to create an intensity distribution of light *transmitted* into an object. Thus, instead of storing the reflected flux, we use a shadow map to store the incoming flux Φ_p through each texel. Despite this minor difference in use, we will refer to these shadow maps as RSMs in order to keep our terminology more consistent with previous work.

The shadow map spans a virtual rectangular plane as shown in Fig. 3 and is characterised by the field of view FOV, the height to width aspect ratio α, and the number of texels in the horizontal and vertical directions, N_x and N_y, respectively. For a point light source which emits uniformly in all directions, the flux through a texel is $\Phi_p = I\Delta\omega_p$, where I is the constant intensity and $\Delta\omega_p$ is the solid angle subtended by the texel seen from the light source

$$\Delta\omega_p \approx \frac{A_p \cos\theta_p}{||\boldsymbol{r}_p||^2} = \frac{4\alpha \tan^2\left(\frac{\text{FOV}}{2}\right)\cos^3\theta_p}{N_x N_y}.$$

A_p is the pixel area in the virtual plane, $\boldsymbol{r}_p$ is the vector from the light source to the pixel center, and θ_p is the angle between the plane normal and $\boldsymbol{r}_p$.

For each texel in the shadow map, we use the flux Φ_p to estimate the transmitted intensity of a virtual point light source inside the object. We use a bidirectional transmission distribution function (BTDF) with angular dependence proportional to a clamped cosine lobe centred around the refracted direction obtained from Snell's law, i.e., the virtual point light intensity is

$$I_p(\boldsymbol{\omega}) = \frac{1}{\pi} T(\boldsymbol{\omega}_p, \boldsymbol{\omega}_t)\Phi_p \max\{0, \boldsymbol{\omega}_t \cdot \boldsymbol{\omega}\}, \qquad (1)$$

where T is the Fresnel transmission factor from the incoming direction $\boldsymbol{\omega}_p$ to the refracted direction $\boldsymbol{\omega}_t$. Next, we use the world space position $\boldsymbol{x}_p$ to identify the grid cell that the texel is projected into. This grid cell will lie on the boundary of the scene geometry as shown in Fig. 2(a). In each of these boundary grid cells we accumulate the point light intensities from the RSM to a single point light source in the cell center, *i.e.* $I(\boldsymbol{\omega}) = \sum_{p, \boldsymbol{x}_p \in \text{cell}} I_p(\boldsymbol{\omega})$. This process is referred to as injection of light into the LPV grid.

In principle, we could choose other forms of the BTDF, *e.g.* an ideal specular material should be represented by a BTDF proportional to a delta function $\delta(\boldsymbol{\omega} - \boldsymbol{\omega}_t)$. This type of light distribution is, however, impossible to represent with our method as will be discussed further in Sec. 6.

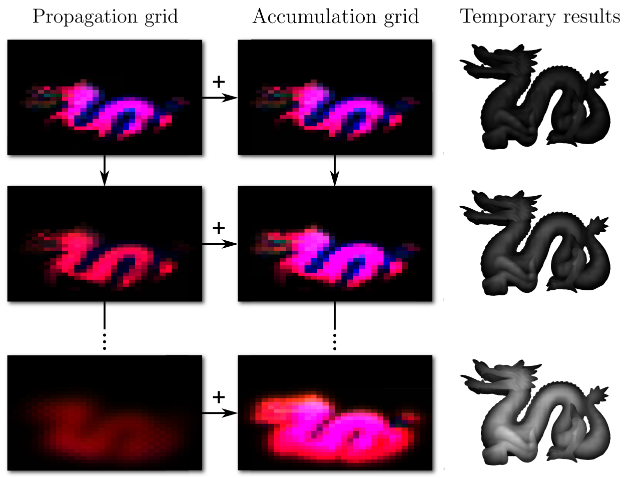

Figure 4: *Iterative propagation. Each propagation step updates the light distribution which is stored in the propagation grid. The accumulation grid collects the distributions from all propagation steps and is used as the source of illumination for the rendering pass.*

Similar to Kaplanyan and Dachsbacher [2010] we use the basis of real spherical harmonics to represent the angular dependence of the intensity distribution, *i.e.*

$$I(\boldsymbol{\omega}) = \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} c_{lm} y_{lm}(\boldsymbol{\omega}),$$

where the expansion coefficients c_{lm} are obtained by projecting Eq. (1) on the spherical harmonics. The upper limit in the summation $l_{\max}$ defines the number of bands included in the basis.

The reflected part of the incoming flux is not used in our SSLPV scheme but is simply added as standard diffuse and specular reflections to the final result, Fig. 2(c). Shadow effects are easily added to the direct illumination since the RSMs are readily available.

3.2 Propagation

Our propagation scheme is similar to the scheme presented by Kaplanyan and Dachsbacher [2010]. For completeness, we briefly outline the method below and refer to the original papers [Kaplanyan and Dachsbacher 2010; Kaplanyan et al. 2011] for further details.

The basic idea is to redistribute light by gathering flux transferred from neighbouring grid cells. In a single propagation pass, we traverse through all grid cells. We propagate light into one (destination) cell by considering the point light intensities in its six neighbouring (source) cells along the axial directions. For one source cell s, the amount of flux that reaches the face f in the destination cell d is $\Phi_{f,d\leftarrow s} = \int_{\Delta\omega_{f,s}} I_s(\boldsymbol{\omega})d\omega$, where $I_s(\boldsymbol{\omega})$ is the intensity distribution in the source cell and $\Delta\omega_{f,s}$ is the solid angle subtended by f seen from the source cell center. We now create a virtual point light source in the destination cell which has an intensity profile proportional to a clamped cosine lobe directed towards f and emits exactly the flux $\Phi_{f,d\leftarrow s}$, *i.e.* $I_{f,d\leftarrow s}(\boldsymbol{\omega}) = \Phi_{f,d\leftarrow s}/\pi \max\{0, \boldsymbol{n}_f \cdot \boldsymbol{\omega}\}$, where $\boldsymbol{n}_f$ is the face normal. We repeat this procedure for all faces, except the bordering face between the source and destination cells, and all six source cells. All these contributions are then summed to a single point light source in the center of the destination cell with the intensity distribution

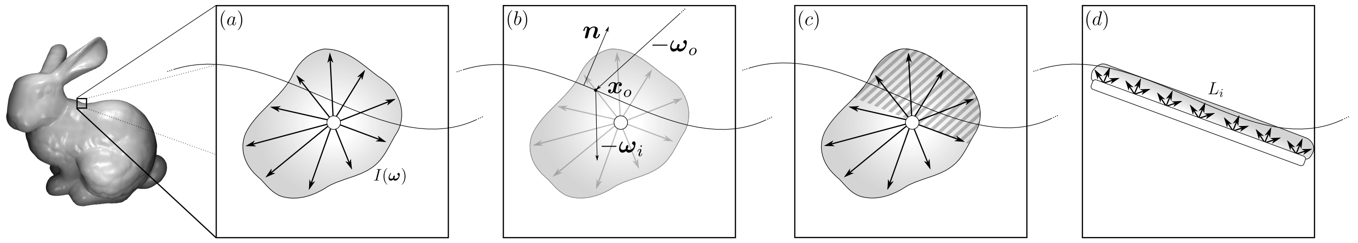

Figure 5: *(a) The point light intensity in a boundary grid cell of the final accumulation grid. (b) The path from the camera refracted through the surface cannot hit an infinitesimal light source. (c) The amount of flux going outwards (shaded dark gray area) is converted to a virtual area light (d).*

$I_d(\boldsymbol{\omega}) = \sum_s \sum_f' I_{f,d\leftarrow s}(\boldsymbol{\omega})$. The primed summation excludes the bordering face. In practice, we can propagate between two cells by a single matrix-vector multiplication as we will show in Sec. 4. The propagation step is repeated iteratively and the point light intensities are continuously updated in a grid, which we will refer to as the *propagation grid*. After each step the point light intensities are accumulated in a separate grid, the *accumulation grid*, which eventually will represent the final distribution of light. The accumulation grid is finally used as the light source for rendering of the scene. Figure 4 illustrates how the light intensities are collected in the accumulation grid. Convergence is achieved when the propagation grid is empty and, correspondingly, the accumulation grid is constant.

In contrast to the standard LPV scheme described above, we aim to model participating media, hence, we need to take absorption and scattering effects into account. Light transport in participating media is usually described by the radiative transport equation (RTE). Unfortunately, such a formulation requires that light transport is characterised by rays carrying radiance. Therefore, we cannot directly use the RTE in our propagation scheme. If the RTE were to be implemented in the SSLPV propagation scheme we would first need to convert our point light intensities to radiance carrying rays. Second, we should integrate the RTE in order to estimate the flux transport to a face in a neighbouring cell. Our main focus is to maintain an efficient implementation so instead of an RTE based approach we model participating media by a simple method which fits straightforwardly in the standard LPV propagation scheme.

3.2.1 Absorption

Absorption is characterised by a colour dependent extinction coefficient σ_t, *i.e.* the fraction of radiance being absorbed or scattered per unit length. This leads to an exponential decay of radiance, known as Beer's law. In our approach, absorption is modelled by attenuating the flux transfer between two adjacent cells by the factor $\exp(-\sigma_t \Delta X_{\text{cell}})$, where ΔX_{cell} is the cell width.

In heterogeneous materials the extinction coefficient will vary in space, and, hence, we use a separate grid to store the RGB components of the extinction coefficient.

3.2.2 Scattering

Our volume scattering model should have the property that flux transfer between neighbouring cells must depend on the scattering properties of the medium. In order to fulfil this requirement, we model scattering by redistributing the intensity distribution according to the phase function $p(\boldsymbol{\omega} \cdot \boldsymbol{\omega}')$ and a dimensionless scattering parameter $\sigma_s' \in [0, 1]$ which accounts for the amount of scattering

$$I_{\text{sc}}(\boldsymbol{\omega}) = \left(1 - \sigma_s'\right) I(\boldsymbol{\omega}) + \sigma_s' \int_{4\pi} p(\boldsymbol{\omega} \cdot \boldsymbol{\omega}') I(\boldsymbol{\omega}') d\omega'. \qquad (2)$$

$I(\boldsymbol{\omega})$ is the intensity distribution obtained according to the reprojection step in the standard LPV scheme and $I_{\text{sc}}(\boldsymbol{\omega})$ is the intensity distribution being propagated to the neighbouring cells. Furthermore, the phase function is assumed to be normalised to unity on the unit sphere. Note that Eq. (2) deceptively resembles the scattering term in the RTE but involves intensity instead of radiance. However, we do not present Eq. (2) as a rigourous approximation to the true volume scattering but rather as an intuitive model with the desired property mentioned above.

The spherical harmonics addition theorem can be used to express the phase function in terms of spherical harmonics $p(\boldsymbol{\omega} \cdot \boldsymbol{\omega}') = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} p_l y_{lm}(\boldsymbol{\omega}') y_{lm}(\boldsymbol{\omega})$, where the multipole moments p_l depend on the scattering properties of the medium. We assume the Henyey-Greenstein form for the phase function characterised by the asymmetry parameter $g \in [-1, 1]$ with g negative (positive) corresponding to backward (forward) scattering [Henyey and Greenstein 1941]. In this case the multipole moments are simply $p_l = g^l$. Note that the addition theorem is often expressed in terms of complex spherical harmonics. It is, however, straightforward to show the corresponding theorem for the real spherical harmonics.

Inserting the spherical harmonics expansion of the intensity distribution in Eq. (2) and using the orthogonality of the spherical harmonics leads to

$$I_{\text{sc}}(\boldsymbol{\omega}) = \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} \left[(1 - \sigma_s') + \sigma_s' g^l\right] c_{lm} y_{lm}(\boldsymbol{\omega}). \qquad (3)$$

Kajiya and Herzen [1984] noted that the phase function is diagonal in the basis of complex spherical harmonics, i.e., scattering does not mix between different spherical harmonics components. Equation (3) shows the equivalent statement in the real basis. This fact allows us to implement Eq. (3) easily in the existing propagation scheme simply by multiplying the expansion coefficient c_{lm} by $(1 - \sigma_s') + \sigma_s' g^l$ prior to the propagation step. Note that complete forward scattering, $g = 1$ cannot be distinguished from no scattering. Another limiting case is isotropic scattering $g = 0$. In this case, all coefficients in the spherical harmonics expansion are reduced by $1 - \sigma_s'$, except the spherically symmetric component.

As for the absorption, we store the scattering and asymmetry parameters in grids in order to account for heterogeneous materials.

3.3 Rendering

In the rendering step we need to convert the light distribution inside the object to a radiance estimate on the outside of the surface. The distribution inside the object is represented by the point light intensities of the boundary grid cells in the final accumulation grid. In general we would obtain the outgoing radiance L_o in the direction

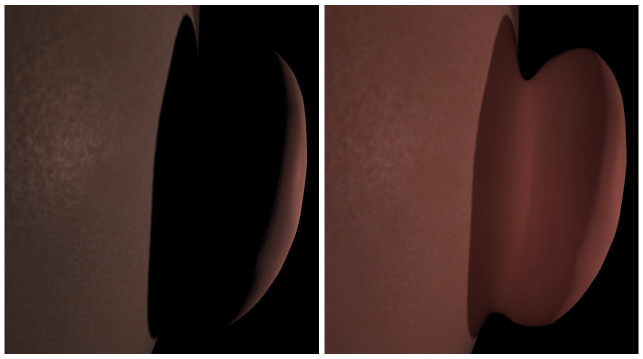

Figure 6: *Comparison of a human ear (32K triangles) rendered without and with SSLPV.*

$\boldsymbol{\omega}_o$ from a surface point $\boldsymbol{x}_o$ to the camera as

$$L_o(\boldsymbol{x}_o, \boldsymbol{\omega}_o) = T(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) L_i(\boldsymbol{x}_o, \boldsymbol{\omega}_i), \quad (4)$$

where L_i is the incident radiance and T is again the Fresnel transmission factor. The incoming direction $\boldsymbol{\omega}_i$ is determined from $\boldsymbol{\omega}_o$ and Snell's law. In our case, however, we cannot use Eq. (4) directly because we store point light intensities in the LPV grid rather than radiances.

One problem with the point light representation is to estimate the incident radiance consistently. Point light radiance is estimated as the intensity divided by the squared distance between the light source and the point of incidence. However, such an estimate is inconsistent in the LPV context because a slight translation of the surface with respect to the grid will lead to a dramatic change in incident radiance, in particular if the surface is close to the light source. Similarly, changing the grid resolution could also lead to significant changes in the radiance estimate. Another problem related to point lights is that the probability for an eye or light ray path to connect the light source and the camera is infinitesimal. Effectively, this means that a ray from the camera which is refracted through the surface will never hit the light source as shown in Fig. 5(b). Thus, in order to solve these issues, we need an alternative method to estimate radiance. Our proposal is to convert the point light intensity to incident radiance by approximating the point light source with a uniform area light which is parallel to the surface and spans a cross section of area A of the grid cell as shown in Figs. 5(c) and (d). Here we assume that the LPV grid has been chosen dense enough that the surface can be considered being locally flat across each grid cell. For consistency, we require that the total flux emitted by the area light $\Phi_{\text{area}} = L_i A\pi$, must be equal to the flux emitted in the hemisphere directed towards the surface by the original point light $\Phi_{\text{point}} = \int_{2\pi} I(\boldsymbol{\omega}) d\omega$, *i.e.* $L_i = \Phi_{\text{point}}/(A\pi)$. If the surface normal coincides with the world space z-axis from which our spherical harmonics are defined, it would be easy to estimate the area light radiance as

$$\begin{aligned} L_i &= \frac{1}{A\pi} \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} c_{lm} \int_0^{2\pi} \int_0^{\pi/2} y_{lm}(\theta, \phi) \sin\theta d\theta d\phi \\ &= \frac{2}{A} \sum_{l=0}^{l_{\max}} c_{l0} \int_0^{\pi/2} y_{l0}(\theta) \sin\theta d\theta. \quad (5) \end{aligned}$$

In the equation above we used that $\int_0^{2\pi} y_{lm}(\theta, \phi) d\phi \propto \delta_{m0}$ and that $y_{l0}(\theta, \phi)$ is independent of the azimuthal angle ϕ. The angular integrals in Eq. (5) are easily calculated analytically.

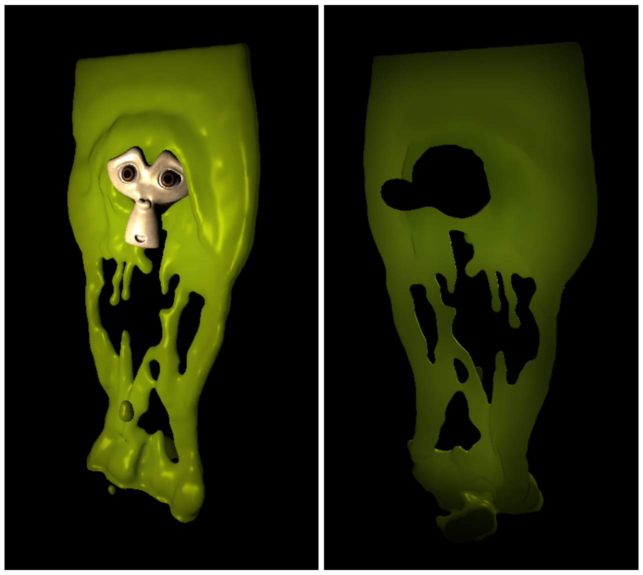

Figure 7: *Animated slime flowing around a monkey head (max. 110K triangles) demonstrates that our method handles changing topology.*

In general, however, the surface normal $\boldsymbol{n}$ will not be aligned along the world space z-axis. In this case, we can still use Eq. (5) provided that we express the expansion coefficients with respect to a set of spherical harmonics defined from the surface normal instead of the world space coordinate system. Let us arrange the expansion coefficients in the world space system in a column vector $\boldsymbol{c} = (c_{00}, c_{1-1}, c_{10}, \cdots, c_{l_{\max} l_{\max}})^T$, and a corresponding vector $\boldsymbol{c}^{(\boldsymbol{n})}$ which contains the expansion coefficients in the surface normal coordinate system. The two sets of coefficients are related through

$$\boldsymbol{c}^{(\boldsymbol{n})} = [\boldsymbol{R}_{\text{SH}}(\alpha, \beta, \gamma)]^{-1} \cdot \boldsymbol{c}, \quad (6)$$

where (α, β, γ) are the zyz Euler angles that define the rotation and $\boldsymbol{R}_{\text{SH}}(\alpha, \beta, \gamma)$ is the spherical harmonics rotation matrix [Green 2003]. Notice that Eq. (6) contains the inverse (adjoint) rotation matrix which corresponds to a passive rotation. The Euler angles fulfil $\tan\alpha = n_y/n_x$, $\cos\beta = n_z$, and $\gamma = 0$, where the normal vector components refer to the world space coordinate system. The final outgoing radiance is then

$$L_o(\boldsymbol{x}_o, \boldsymbol{\omega}_o) = T(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) \frac{2}{A} \sum_{l=0}^{l_{\max}} c_{l0}^{(\boldsymbol{n})} \int_0^{\pi/2} y_{l0}(\theta) \sin\theta d\theta. \quad (7)$$

The radiance estimate presented in this section avoids the two main issues by the point light representation, namely the singularity if the surface accidentally coincides with the point light as well as the vanishing probability for a path to connect the light source and the camera. The trade-off is that we lose directional information from the intensity distribution. For highly scattering media, however, it is well-known that the distribution becomes nearly independent of directions as will be discussed further in Sec. 6 and, hence, we propose that this trade-off is acceptable.

4 Implementation details

We have implemented the SSLPV algorithm using the OpenGL graphics API including the OpenGL Shading Language (GLSL).

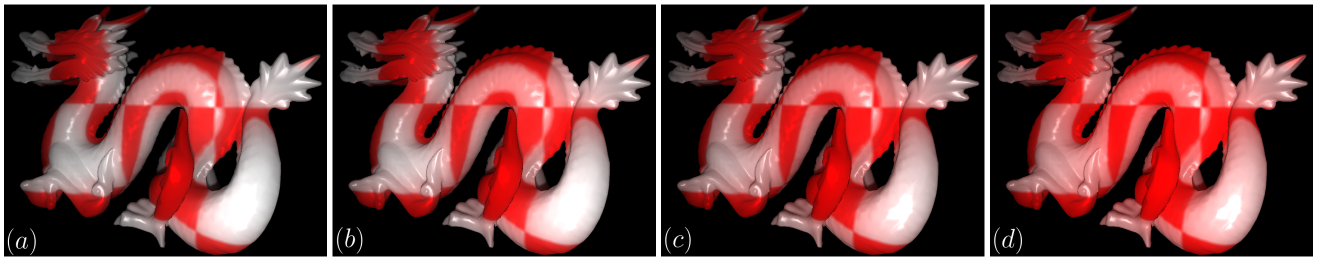

Figure 8: *Comparison of visual quality when rendering the Stanford Dragon (50K triangles) using 8, 16, 32 and 64 propagation steps, respectively. The subsurface scattering contribution has been scaled up by a factor of* 3, 2 *and* 1.2 *in the first three renderings in order to compensate for the light which has not yet left the object. Note that using many propagation steps results in excessive dissipation because our model does not currently remove light which is leaving the object from the propagation grid. We leave this issue as future work.*

We use two spherical harmonics bands in our implementation, i.e., $l_{\max} = 1$. The spherical harmonics are the basis vectors, in a four dimensional subspace

$$\begin{aligned}
\boldsymbol{e}_0 &= y_{00}(\boldsymbol{\omega}) &&= 1/\sqrt{4\pi}, \\
\boldsymbol{e}_1 &= y_{1-1}(\boldsymbol{\omega}) &&= \sqrt{3/(4\pi)}\sin\theta\sin\phi, \\
\boldsymbol{e}_2 &= y_{10}(\boldsymbol{\omega}) &&= \sqrt{3/(4\pi)}\cos\theta, \\
\boldsymbol{e}_3 &= y_{11}(\boldsymbol{\omega}) &&= \sqrt{3/(4\pi)}\sin\theta\cos\phi.
\end{aligned}$$

In this basis, the intensity is represented as the column vector $\boldsymbol{c} = (c_{00}, c_{1-1}, c_{10}, c_{11})^T$. For each colour component we use two RGBA 3D textures as alternating source and target during propagation, and one for the accumulation grid to store the four coefficients across the LPV grid. Typically, we use a grid resolution of 32^3 and store the coefficients as 16 bit floats. Furthermore, we need two 3D textures to store the properties of heterogeneous materials, one for the three colour components of the extinction coefficient σ_t and one for the colour components of the asymmetry parameter g.

Propagation between the source cell s and the destination cell d can be written conveniently in terms of matrix-vector operations in the propagation grid. First, the flux transfer can be expressed as

$$\begin{aligned}
\Phi_{f,d\leftarrow s} &= \int_{\Delta\omega_{f,s}} I_s(\boldsymbol{\omega})d\omega = \sum_{l,m} c_{s,lm} \int_{\Delta\omega_{f,s}} y_{lm}(\boldsymbol{\omega})d\omega \\
&= \boldsymbol{v}_{f,d\leftarrow s}^T \cdot \boldsymbol{c}_s,
\end{aligned}$$

with the elements of the vector $\boldsymbol{v}_{f,d\leftarrow s}$

$$[v_{f,d\leftarrow s}]_{l^2+l+m} = \int_{\Delta\omega_{f,s}} y_{lm}(\boldsymbol{\omega})d\omega.$$

Second, we find the expansion coefficients for the point light in the destination cell by projecting the intensity distribution $I_{f,d\leftarrow s}(\boldsymbol{\omega}) = \Phi_{f,d\leftarrow s}/\pi \max\{0, \boldsymbol{n}_f \cdot \boldsymbol{\omega}\}$ on the spherical harmonics

$$\begin{aligned}
\boldsymbol{c}_{f,d\leftarrow s} &= \begin{pmatrix} \int_{4\pi} I_{f,d\leftarrow s} y_{00}(\boldsymbol{\omega})d\omega \\ \int_{4\pi} I_{f,d\leftarrow s} y_{1-1}(\boldsymbol{\omega})d\omega \\ \int_{4\pi} I_{f,d\leftarrow s} y_{10}(\boldsymbol{\omega})d\omega \\ \int_{4\pi} I_{f,d\leftarrow s} y_{11}(\boldsymbol{\omega})d\omega \end{pmatrix} \\
&= \frac{1}{\pi} \begin{pmatrix} \int_{2\pi} \boldsymbol{n}_f \cdot \boldsymbol{\omega} y_{00}(\boldsymbol{\omega})d\omega \\ \int_{2\pi} \boldsymbol{n}_f \cdot \boldsymbol{\omega} y_{1-1}(\boldsymbol{\omega})d\omega \\ \int_{2\pi} \boldsymbol{n}_f \cdot \boldsymbol{\omega} y_{10}(\boldsymbol{\omega})d\omega \\ \int_{2\pi} \boldsymbol{n}_f \cdot \boldsymbol{\omega} y_{11}(\boldsymbol{\omega})d\omega \end{pmatrix} \cdot \boldsymbol{v}_{f,d\leftarrow s}^T \cdot \boldsymbol{c}_s.
\end{aligned}$$

Thus, the new coefficients are effectively the result of the matrix product $\boldsymbol{c}_{f,d\leftarrow s} = \boldsymbol{M}_{f,d\leftarrow s} \cdot \boldsymbol{c}_s$, where the matrix elements are

$$\begin{aligned}
[M_{f,d\leftarrow s}]_{l^2+l+m,l'^2+l'+m'} &= \frac{1}{\pi} \int_{2\pi} \boldsymbol{n}_f \cdot \boldsymbol{\omega} y_{lm}(\boldsymbol{\omega})d\omega \\
&\quad \times \int_{\Delta\omega_{f,s}} y_{l'm'}(\boldsymbol{\omega})d\omega.
\end{aligned}$$

Furthermore, we can easily sum the contribution from all faces to obtain the total propagation from s to d as a single matrix-vector multiplication $\boldsymbol{c}_{d\leftarrow s} = \left(\sum_f' \boldsymbol{M}_{f,d\leftarrow s}\right) \cdot \boldsymbol{c}_s$. The constant propagation matrix in the parenthesis can be calculated once and for all. For example, in an axis aligned uniform LPV grid, the propagation matrices from source cells in the positive and negative x directions are

$$\sum_f{}' \boldsymbol{M}_{f,d\leftarrow d\pm x} = \begin{pmatrix} 0.167 & 0 & 0 & \mp 0.240 \\ 0 & 0.070 & 0 & 0 \\ 0 & 0 & 0.070 & 0 \\ \mp 0.037 & 0 & 0 & 0.062 \end{pmatrix}.$$

The corresponding matrices for source cells along the y and z directions are obtained from the matrix above by permutation of the fourth row and column by the second and third row and column, respectively.

In the rendering step, we can simplify the expression for outgoing radiance Eq. (7), because we only include two bands in the spherical harmonics expansion, i.e.,

$$L_o(\boldsymbol{x}_o, \boldsymbol{\omega}_o) = T(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o)\frac{2}{A}\left(\sqrt{\frac{1}{4\pi}}c_{00}^{(\boldsymbol{n})} + \frac{1}{2}\sqrt{\frac{3}{4\pi}}c_{10}^{(\boldsymbol{n})}\right),$$

where the two expansion coefficients are obtained from Eq. (6) and the explicit representation of the rotation matrix [Green 2003]

$$\begin{aligned}
c_{00}^{(\boldsymbol{n})} &= c_{00}, \\
c_{10}^{(\boldsymbol{n})} &= \sin\alpha\sin\beta c_{1-1} + \cos\beta c_{10} + \cos\alpha\sin\beta c_{11}.
\end{aligned}$$

The coefficients c_{lm} are the coefficients in the accumulation grid which refer to the world space axes.

As a final implementation detail, we should mention that during the first two propagation steps we do not accumulate the light intensities from the propagation grid in order to reduce temporal flickering resulting from moving light sources and geometry.

5 Results

We have implemented our method on an Intel Xeon E5620 2.40GHz CPU with 4GB of memory utilising an NVIDIA GeForce GTX 580 GPU with 1.5GB dedicated memory. The various steps of our algorithm are implemented in GLSL shaders, which will be made available on the companion website (http://cg.alexandra.dk/publications/sslpv/). All renderings are performed at 1280×720 with a basic Phong surface shading of the reflected direct light contribution. We have tested the various components of our method using a number of scenes which each demonstrate a single component for clarity, *i.e.* homogeneous or heterogeneous materials, the number of propagation steps, the LPV grid resolution, and the number of light sources. Our method is highly flexible in the sense that each of these components can be scaled up or down independently in order to achieve a desired performance or quality. All renderings were performed using an LPV grid of resolution 32^3 and 32 propagation steps unless otherwise stated. Likewise, most examples use one point light with an RSM of resolution 1024^2, although only 256^2 texel values are used for injection while the remaining resolution is merely used for conventional shadow mapping. Heterogeneous material tests were performed using a 128^3 grid for material parameters. We use the material parameters to shade the direct light which justifies the higher resolution; otherwise a resolution equal to the LPV grid is sufficient.

Figure 6 shows the effect of adding SSLPV to a model of a human ear lighted from the side. The flesh is modelled as a homogeneous material with a slightly red tint, while texture, normal, and specular maps provide surface details to the shading. Figure 7 demonstrates that our method handles completely dynamic geometry including topology changes. Again, a homogeneous material is used for the slime, while the monkey head is rendered using a traditional opaque Phong shading with texturing. Note that the head is not included in the RSM and therefore does not cast shadows. While the two previous examples use 32 propagation steps, Fig. 8 illustrates that visually pleasing results can be achieved using as little as 8 propagation steps depending on the scene. Figure 9 shows the Stanford Thai Statue rendered with heterogeneous materials using different LPV grid resolutions and lighting, while Fig. 10 shows two frames from a scene with an animated heterogeneous material. Please see the accompanying video for animations.

Timings averaged over several animation cycles (see the video) are presented in table 1. We have performed them using both 8 and 32 propagations steps as the number of steps affects the distance light propagates in the LPV grid. Hence, the desired number of steps to use is highly application and geometry dependent. The test scenes which demonstrate homogeneous materials have also been timed with heterogeneous materials to point out the extra overhead generated by the additional texture lookups. As the timings indicate, SSLPV is largely unaffected by the geometry complexity which only influences the RSM and shading steps. Thus, when our method is used with a traditional RSM rendering pipeline, the additional work is negligible in many cases. The complexity of the propagation step scales with the resolution of the propagation grid and is therefore independent of the scene geometry. However, it seems that increased computational complexity in one step of our method may affect the actual timings of other steps as is evident from the timings of Fig. 1. Comparing the rows involving Fig. 9 shows that the performance penalty of increasing the LPV grid resolution dominates the additional cost of adding one extra point light.

6 Discussion

Our implementation is based on a uniform cubic LPV grid. Such a grid is, however, not the optimal choice for geometries that vary over different length scales. In such cases, it would be more appropriate to use cascaded grids which should work straightforwardly with our method [Kaplanyan and Dachsbacher 2010].

The grid resolution impacts the quality of the results. A coarse grid can lead to visible artefacts, for example if objects are translated within the grid. Furthermore, a coarse grid can lead to artificial light bleeding across regions that block light. In particular, this effect becomes an issue when we consider objects that undergo topological changes such as merging objects. In this case we see light transport between the objects slightly prior to the actual merging. Additionally, small-scale features in the order of a grid cell width tend to be too dark either because the injected light is dissipated away before it can be collected in the accumulation grid or because the feature is missed entirely by the injection step. These effects can be reduced to some extent by introducing a blocking potential [Kaplanyan and Dachsbacher 2010].

Fig.	Average timings (ms) RSM	Injection	Propagation	Shading	FPS	Lights / Grid res.
1	— / 5.91	— / 0.13	— / 2.72	— / 5.94	— / 65	1 / 32^3
6	0.30 / 0.29	0.30 / 0.32	0.72 / 1.67	0.48 / 0.81	475 / 290	1 / 32^3
7	0.17 / 0.18	0.29 / 0.32	0.72 / 1.66	0.13 / 0.14	630 / 380	1 / 32^3
8(a)+10	— / 0.24	— / 0.29	— / 1.67	— / 0.36	— / 350	1 / 32^3
9(a)	— / 0.45	— / 0.12	— / 1.66	— / 0.39	— / 320	1 / 32^3
9(b)	— / 0.94	— / 0.27	— / 3.77	— / 0.87	— / 155	2 / 64^3
1	— / 5.92	— / 0.12	— / 10.88	— / 5.94	— / 43	1 / 32^3
6	0.31 / 0.30	0.37 / 0.40	2.88 / 6.75	0.47 / 0.79	220 / 115	1 / 32^3
7	0.23 / 0.21	0.36 / 0.41	2.91 / 6.68	0.14 / 0.21	250 / 125	1 / 32^3
8(c)+10	— / 0.24	— / 0.38	— / 6.78	— / 0.50	— / 120	1 / 32^3
9(a)	— / 0.47	— / 0.23	— / 6.69	— / 0.50	— / 120	1 / 32^3
9(b)	— / 0.89	— / 0.25	— / 14.95	— / 1.38	— / 55	2 / 64^3

Table 1: *Average homogeneous / heterogeneous timings for shown figures. Top section uses 8 propagation steps. Bottom section uses 32 propagation steps.*

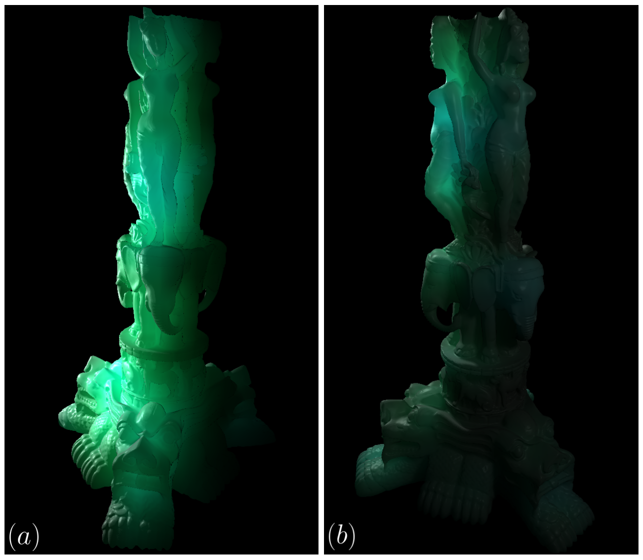

Figure 9: *Backlit Stanford Thai Statues (323K triangles) using heterogeneous materials resembling jade. (a)* 32^3 *LPV grid. 1 point light. (b)* 64^3 *LPV grid. 2 point lights.*

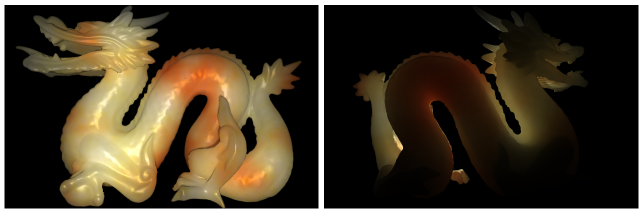

Figure 10: *Stanford Dragon (50K triangles) rendered using an animated heterogeneous material.*

One further limitation relates to the low-order spherical harmonics representation which tends to average the directional distribution of light, effectively leading to diffusion. A true directional propagation would require an infinite number of spherical harmonics. Thus, the SSLPV method combined with a low-order spherical harmonics is not suited to preserve distinct directional features of light, *e.g.* directional light transported in vacuum or in a weakly scattering material. However, the diffusion effect actually resembles the effect of light transport in a highly scattering medium. In this case, the effect of many scattering events is to smooth the directional dependence [Stam 1995]. Hence, we propose that our method should work very well for highly scattering media.

7 Conclusion and Future Work

In conclusion, we have presented SSLPV as an efficient method to account for subsurface scattering in real-time. Our approach requires no precomputation and is able to handle fully dynamic scenes with heterogeneous materials.

In the future we would like explore the possibility of using a tetrahedral mesh based on the scene geometry instead of a fixed LPV grid. We believe that a tetrahedral mesh will produce superior results in terms of quality because the mesh actively follows the geometry and, accordingly, grid artefacts are reduced. The challenge is that dynamic scenes require re-meshing which will likely impact performance.

Furthermore, we plan to replace our GLSL based propagation implementation by a CUDA or OpenCL based implementation. We propose that these programming and memory models more naturally fit the present problem. For example, it would be interesting to include more than two spherical harmonics bands in order to simulate more directional light transport. Due to our method being heavily texture-bandwidth limited, adding an extra spherical harmonics band will severely hurt the performance in the current implementation. On the contrary, a CUDA or OpenCL implementation would be able to more efficiently exploit the memory bandwidth.

References

CARR, N., HALL, J., AND HART, J. 2003. GPU Algorithms for Radiosity and Subsurface Scattering. In *Proc. of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, Eurographics Association, 51–59.

CHANG, C.-W. 2008. Real-Time Translucent Rendering Using GPU-based Texture Space Importance Sampling. *Computer Graphics Forum 27*, 517–526.

DACHSBACHER, C., AND STAMMINGER, M. 2003. Translucent Shadow Maps. In *EGRW 03 Proc. of the 14th Eurographics workshop on Rendering*, Eurographics Association, 197–201.

DACHSBACHER, C., AND STAMMINGER, M. 2005. Reflective Shadow Maps. In *Proc. of the 2005 Symposium on Interactive 3D Graphics and Games*, ACM, 203–213.

D'EON, E., LUEBKE, D., AND ENDERTON, E. 2007. Efficient Rendering of Human Skin. *Eurographics Symposium on Rendering 2007*, 147–157.

FRANÇOIS, G., PATTANAIK, S., BOUATOUCH, K., AND BRETON, G. 2008. Subsurface Texture Mapping. *IEEE computer graphics and applications 28*, 34–42.

GEIST, R., AND WESTALL, J. 2011. Lattice-Boltzmann Lighting Models. In *GPU Computing GEMS Emerald Edition*, 381–399.

GEIST, R., WESTALL, J., AND SCHALKOFF, R. 2004. Lattice-Boltzman Lighting. In *Proc. Eurographics Symposium on Rendering*, Eurographics Association, 355–362.

GREEN, R. 2003. Spherical Harmonic Lighting: The Gritty Details. *Archives of the Game Developers Conference*.

HABLE, J., BORSHUKOV, G., AND HEJL, J. 2009. Fast Skin Shading. In *Shader X7*, Charles River Media, 161–173.

HAO, X., AND VARSHNEY, A. 2004. Real-time Rendering of Translucent Meshes. *ACM Trans. Graph. 23*, 120–142.

HENYEY, L. G., AND GREENSTEIN, J. L. 1941. Diffuse Radiation in the Galaxy. *Annales dAstrophysique 93*, 70–83.

JENSEN, H. W., AND BUHLER, J. 2002. A Rapid Hierarchical Rendering Technique for Translucent Materials. *ACM Trans. Graph. 21*, 576–581.

JENSEN, H. W., MARSCHNER, S. R., LEVOY, M., AND HANRAHAN, P. 2001. A Practical Model for Subsurface Light Transport. In *Proc. of SIGGRAPH 2001*, ACM, 511–518.

JIMENEZ, J., AND GUTIERREZ, D. 2008. Faster Rendering of Human Skin. In *CEIG*, vol. 5, 21–28.

JIMENEZ, J., SUNDSTEDT, V., AND GUTIERREZ, D. 2009. Screen-space Perceptual Rendering of Human Skin. *ACM Trans. on Applied Perception 6*, 1–15.

KAJIYA, J. T., AND HERZEN, B. P. V. 1984. Ray Tracing Volume Densities. In *Computer Graphics (Proc. of SIGGRAPH 84)*, ACM, vol. 18, 165–174.

KAPLANYAN, A., AND DACHSBACHER, C. 2010. Cascaded Light Propagation Volumes for Real-Time Indirect Illumination. In *Proc. of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ACM, 99–107.

KAPLANYAN, A., ENGEL, W., AND DACHSBACHER, C. 2011. Diffuse Global Illumination with Temporally Coherent Light Propagation Volumes. In *GPU Pro 2*, A K Peters, 185–203.

LENSCH, H. P., GOESELE, M., BEKAERT, P., KAUTZ, J., MAGNOR, M. A., LANG, J., AND SEIDEL, H.-P. 2003. Interactive Rendering of Translucent Objects. *Computer Graphics Forum 22*, 195–205.

MERTENS, T., KAUTZ, J., BEKAERT, P., SEIDELZ, H.-P., AND VAN REETH, F. 2003. Interactive Rendering of Translucent Deformable Objects. In *Proc. of the 14th Eurographics workshop on Rendering*, Eurographics Association, 130–140.

MERTENS, T., KAUTZ, J., BEKAERT, P., VAN REETH, F., AND SEIDEL, H.-P. 2005. Efficient Rendering of Local Subsurface Scattering. *Computer Graphics Forum 24*, 41–49.

SLOAN, P.-P., HALL, J., HART, J., AND SNYDER, J. 2003. Clustered Principal Components for Precomputed Radiance Transfer. In *Proc. of SIGGRAPH 2003*, ACM, 382–391.

STAM, J. 1995. Multiple scattering as a diffusion process. In *Eurographics Rendering Workshop*, 41–50.

WANG, J., ZHAO, S., TONG, X., LIN, S., LIN, Z., DONG, Y., GUO, B., AND SHUM, H.-Y. 2008. Modeling and Rendering of Heterogeneous Translucent Materials using the Diffusion Equation. *ACM Trans. Graph. 27*, 1–18.

WANG, Y., WANG, J., HOLZSCHUCH, N., SUBR, K., YONG, J.-H., AND GUO, B. 2010. Real-time Rendering of Heterogeneous Translucent Objects with Arbitrary Shapes. *Computer Graphics Forum 29*, 497–506.