Exploring the Use of Glossy Light Volumes for Interactive Global Illumination

Bernardt Duvenhage*
University of Pretoria, South Africa

Kadi Bouatouch†
University of Rennes, France

Derrick Kourie‡
University of Pretoria, South Africa

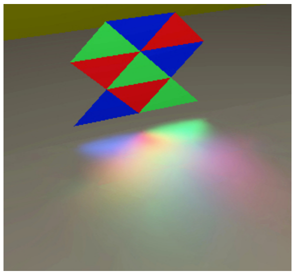


Figure 1: *A glossy transmission through roughened stained glass simulated in real-time.*

Abstract

From the literature, it is known that backward polygon beam tracing and other light volume methods are well suited to gather path coherency from specular scattering surfaces. This is of course useful for modelling and efficiently simulating caustics (LS^+DE paths). This paper generalises backward polygon beam tracing to also model glossy scattering surfaces. To this end the details of a backward polygon beam tracing model and implementation of source-to-glossy-to-diffuse light transport (LG^+DE) paths are researched.

CR Categories: I.3 [Computer Graphics]: Three Dimensional Graphics and Realism;

Keywords: glossy, caustics, backward polygon beam tracing, light volume

1 Introduction

This section introduces the importance of and the problems associated with the simulation of light transport for interactive global illumination. The scope of the paper is then presented followed by an outline of the structure.

1.1 Background

Global illumination refers to a rendering approach that attempts to consider energy contributions from all possible light transport paths between the light source and the eye. Monte Carlo ray tracing based techniques are probably the only techniques that can handle the complexity of the full rendering equation [Kajiya 1986] in large and highly detailed scenes. Such techniques are however computationally expensive. The backward polygon beam tracing technique investigated in this paper may be used to efficiently approximate (or at least lower the variance of a Monte Carlo simulation of) a selection of the possible light transport paths.

The aim with polygon beam tracing is to gather similar light rays—rays that intersect the same polygon—into a beam that may then be processed as a unit. The advantage of beam tracing is that the spatial coherence of the polygons in the image may be exploited to do a smaller number of scene traversals than required for raytracing.

This paper assumes that a material Bi-Directional Scattering Distribution Function (BSDF) can be modelled as a linear combination of a specular component and a diffuse component. In a typical scene the light transport paths may then be differentiated based on the sequence of diffuse (D) and specular (S) material interactions that connects the light source (L) to the eye (E). Using Heckbert's [1990] regular expression notation for light transport paths, an LDE path is any path where energy from the source reaches the eye via a single diffuse surface interaction. Similarly, an LSE path is any path where energy from the source reaches the eye via a single specular surface interaction. The energy transported along each of these types of "direct illumination *or* single bounce" transport paths has analytical solutions and can be simulated efficiently. Indirect illumination, on the other hand, refers to light paths which include multiple surface interactions before reaching the eye. Using Heckbert's notation such a light path may be expressed as $L(S|D)^+E$. However, the energy transported along each $L(S|D)^+E$ path type does not, in general, have an analytical solution.

The simulation of the subset of LS^+DE transport paths (also known as caustics) is one aspect of this problem which has been solved with much success. Examples of reflective and refractive caustics are shown in Figure 2 and Figure 3 respectively. Recently, implementations of high fidelity LS^+DE transport models based on variations of backward polygon beam tracing and also variations of photon tracing have achieved interactive frame rates. Finding the energy transported along the more general $L(S|D)^+E$ paths is however a hard problem to solve efficiently due to the slow convergence of the rendering equation for diffuse interactions.

1.2 Scope of Work

The first bounce material model used in this paper is one of specular micro-facets with a negligible diffuse component. The micro-facets' gradients are assumed to be random with a Gaussian distribution around the average surface normal as modelled by Torrance

*e-mail: bduvenhage@csir.co.za
†e-mail: kadi@irisa.fr
‡e-mail: dkourie@cs.up.ac.za

and Sparrow [1967]. In the rest of this paper a specular material with a large micro-facet variance such as polished car paint or sand-blasted glass is referred to as a Glossy material (G interaction). An *S interaction* is reserved to refer to an interaction from a specular material such as a mirror that has a small micro-facet variance.

With a negligible first bounce diffuse scattering component, second bounce indirect light transport paths now encompass the following four path types:

- source-to-*specular-to-specular*-to-eye,
- source-to-*specular-to-diffuse*-to-eye (caustics),
- source-to-*glossy-to-specular*-to-eye, and
- source-to-*glossy-to-diffuse*-to-eye.

The Monte Carlo simulation of second bounce transport paths with a diffuse or glossy first bounce suffers from a slow convergence of the rendering equation. Therefore, of the above transport path types, potential techniques for the interactive simulation of the last type of paths (viz. LGDE) are explored further. The inherent efficiency of the backward polygon beam tracing technique in gathering path coherency is the main motivation for investigating the technique further. Existing backward beam tracing methods however only model LSDE transport paths.

This paper therefore explores the use of backward polygon beam tracing to efficiently simulate dynamic L(S|G)DE transport paths. In particular this paper:

- Analyses the shape of a first bounce glossy beam,
- describes a backward polygon beam tracing model of L(S|G)DE transport paths, and
- presents a GPU implementation of glossy backward polygon beam tracing.

Extending backward polygon beam tracing to include more general light transport paths is a step towards efficiently implementing second bounce global illumination.

1.3 Structure of Paper

Section 2 summarises the previous work done on backward polygon beam tracing. The details and limitations of the existing LSDE backward polygon beam tracing technique are then discussed in more detail in Section 3. This is followed by the details of the proposed glossy (LGDE) backward polygon beam tracing technique in Section 4.

2 Previous work on Backward Beam Tracing

The set of works on interactive global illumination is vast. This paper will only summarise a set of the most recently published papers on the topic of backward polygon beam tracing.

Backward polygon beam tracing is presented by Watt [1990], but is related to both polygon beam tracing [Heckbert and Hanrahan 1984] and backward raytracing [Arvo 1986]. Heckbert and Hanrahan traced polygon beams from the eye through the scene instead of rays as in traditional forward raytracing. The advantage of beam tracing is that the spatial coherence of the polygons in the image may be exploited to do a smaller number of scene traversals than required for raytracing. Heckbert and Hanrahan also suggested tracing beams from the light source as Watt did with the backward polygon beam tracing technique. Arvo [1986] re-iterated the value of backwards raytracing; that is tracing rays from the light source instead of from the camera as is done in more traditional forward raytracing. Arvo used backward raytracing to render caustics which could not efficiently be simulated using the high fidelity forward raytracing and radiosity rendering techniques of the time. Similar to what Heckbert and Hanrahan proposed and did, Watt used backward polygon beam tracing to improve upon the rendering of caustics by exploiting the spatial coherency of polygons, but from the light's point of view. Watt used the backward polygon beam tracing technique to realistically render the caustics typically found below the surface of a fluid such as on the bottom of a swimming pool. He used projected caustic surface detail polygons, light volumes similar to Nishita et al. [1987] and a raytrace forward renderer to render caustics and single scattering of the light beams in the participating medium.

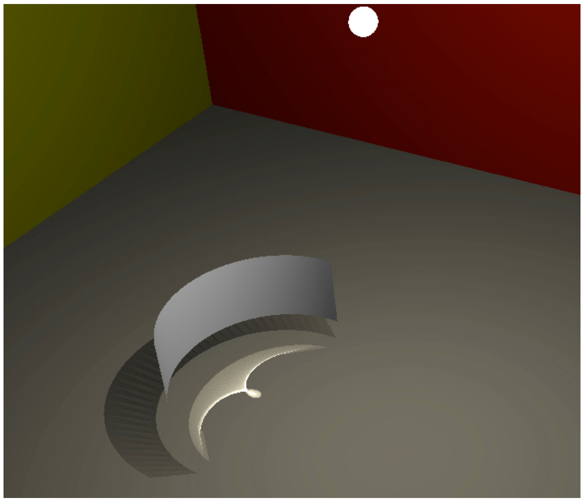


Figure 2: *A cardioid caustic (*40×2 *scattering triangles) rendered in real-time with backward polygon beam tracing.*

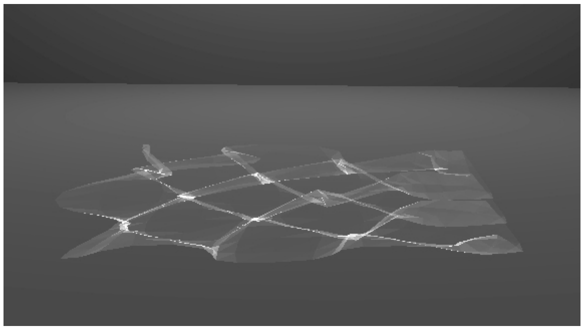


Figure 3: *Caustics from a fluid's surface (*$20 \times 20 \times 2$ *scattering triangles) rendered in real-time with backward polygon beam tracing.*

Nishita and Nakamae [1994] use a scan-line based renderer with an accumulation buffer and illumination volumes to simulate caustics without the need for an expensive raytrace pass. Their method can produce caustics on curved surfaces and includes shadows by making use of the z-buffer.

Chuang and Cheng [1995] can handle non-polygonal illuminated surfaces by searching the light beams within which any intersection point resides. Their light beams are enclosed in a hierarchy of bounding cones for more efficient point-in-beam detection.

Briere and Poulin [2000] use a light image to adaptively refine and construct light beams from the light source. They are also able

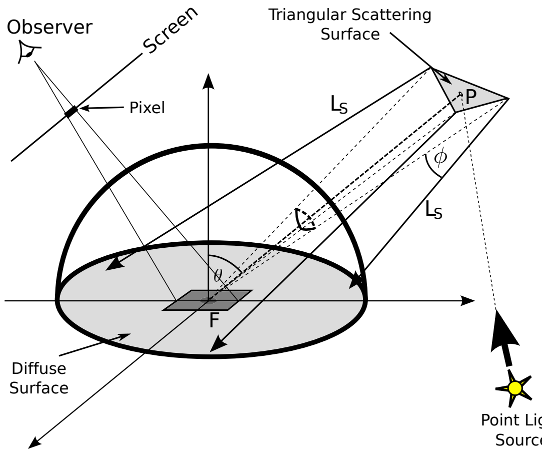


Figure 4: *A fragment lit by a polygon scatterer source.*

to handle single scatter participating media. The beam wavefront is evaluated at the intersections of the edges of the beam with a surface allowing smooth interpolation of flux density for surface points within the beam. A hierarchical structure encloses the light beams for efficient point in beam evaluation.

To further address the problem of efficiency, Iwasaki et al. [2002] make use of graphics hardware to accelerate Nishita and Nakamae's [1994] beam tracing method. They preserve the abilities to render caustics on curved surfaces and to include shadows. Iwasaki et al. [2003] propose an extension to the work by Iwasaki et al. [2002] in which an object is expressed by a set of texture mapped slices. The intensities of the caustics on an object is then calculated by using the slices. They further implement reflection and refraction mapping of the caustic slices to render objects as seen in a reflection or below a refractive fluid surface.

Backward polygon beam tracing has recently also been used by Ernst et al. [2005] to simulate LSDE transport paths. They used *warped* polygon beam volumes, interpolation of beam energies and a GPU implementation to improve the quality and the execution performance.

All of the above backward beam tracing methods model only L*S*DE transport paths to specifically render caustics. More recent methods [Ihrke et al. 2007][Shah et al. 2007][Wyman 2008][Sun et al. 2008][Umenhoffer et al. 2008][Zhou et al. 2008][Spencer and Jones 2009][McGuire and Luebke 2009] for rendering LSDE caustics all focus on GPU accelerated photon- and ray-tracing with impressive results in terms of both quality and performance. The problem of efficient simulation of L(G|D)DE transport paths is however still unsolved due to the loss of coherency when sampling the larger glossy and diffuse rendering equation domain.

3 Backward Polygon Beam Tracing

This section gives more details on the traditional backward polygon beam tracing technique that models and simulates only LSDE transport paths. An implementation approach similar to Chuang and Cheng [1995] is followed in which each diffuse receiver surface is shaded by searching for the light beams within which its fragments' world positions reside. The advantage of Chuang and Cheng's method in being able to illuminate any type of diffuse receiver, whether it be curved or flat, is preserved.

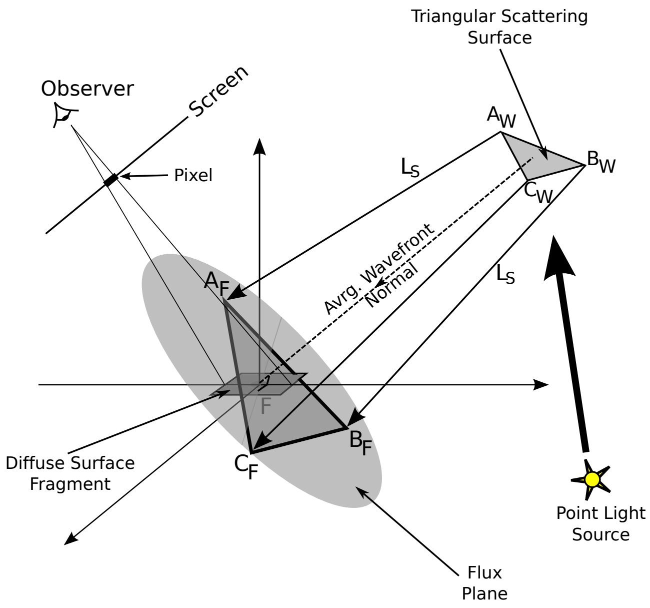


Figure 5: *The flux plane and flux triangle $A_FB_FC_F$.*

In this paper the scattered polygon beams are however not handled as a swept polygon, but rather uniquely scattered vectors with connectivity information which results in an implicit *warped volume*. Propagating the beam as a triplet of associated vectors instead of a simple swept polygon shape with planar sides allows the smooth surface represented by the mesh normals and the resulting light beams to be appropriately decoupled from the triangular surface mesh.

3.1 Modelling LSDE Transport Paths

To model light-to-specular transport paths a polygon beam is traced to each specular scattering surface. Such a surface may be specular reflective or it may be specular transmissive with a specified refraction index. The scattered light beam is modelled by individually scattering the three vertex edges of the incoming beam as shown for a reflected beam in Figure 4. A shadow beam is modelled as three shadow vectors. Each triplet of vectors is associated with a beam flux that is conserved within the beam.

During the final forward rendering pass each diffuse surface fragment is evaluated for whether or not it is in any scattered beams and therefore lit by specularly scattered light. Whether or not a fragment is in a beam is tested by first intersecting the three scattered vectors with a virtual flux plane that is perpendicular to the average wavefront normal and includes the fragment. The three intersection points are then connected into a specular flux triangle (triangle $A_FB_FC_F$ in Figure 5) and the fragment-in-beam test may be reduced to whether or not the fragment (F) is within the flux triangle. The beam's cross sectional area (area $A_FB_FC_F$) is used to calculate the in-beam flux density (the cross sectional $Watt/m^2$) and finally the resulting irradiance at the fragment's position.

This technique can be used to model reflective and refractive caustics as previously shown in Figure 2 and Figure 3 respectively. In these scenes the cylindrical reflector is composed of a mesh of 80 specular scattering triangles and the water surface is composed of a mesh of 800 specular scattering triangles.

Simple local visibility is included by only scattering from the front side of a polygon. First order shadows—shadows directly due to the light source—may be included in a number of ways such as

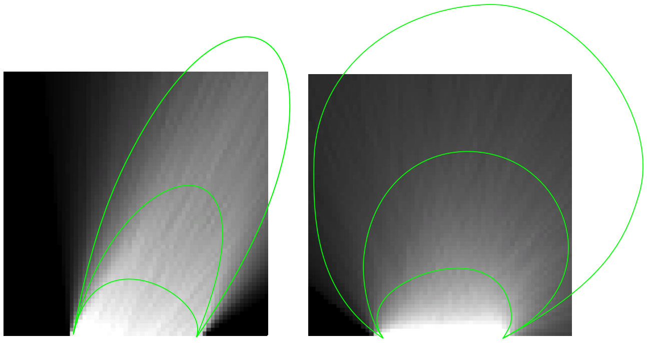


Figure 6: *A side view cross-section of the Monte-Carlo photon density results with the extracted iso-surfaces. Left: Small micro-facet variance. Right: Large micro-facet variance.*

shadow-buffering discussed by Ernst et al. [2005]. In this paper however first order shadows are efficiently generated by tracing a *negative* shadow beam in addition to the scattered polygon beams. Collins [1995] and others have used negative splatting and negative photons before, but it would seem that this technique of using negative energies has not previously been used with backward polygon beam tracing. Figure 2 shows the inclusion of such first order shadows.

A simple GPU implementation of the technique was done to render the example scenes. Each specular reflector is implemented as a triangle mesh surface with shared vertices, shared vertex normals and vertex colours as appropriate. A GLSL fragment shader is executed for all diffuse surfaces. The shader executes the fragment in light beam test for all scattered and shadow beams.

3.2 Analysis and Limitations

The traditional backward polygon beam tracing model of light transport has some limitations. The biggest limitation is that backward polygon beam tracing does not model scattering from glossy and diffuse polygons.

Another limitation is that second order shadows (scattered beams being blocked by polygons) are not included in the light transport model. Including such effects would require expensive clipping or sub-division of the scatterer geometry.

4 Formulating Glossy Backward Polygon Beam Tracing

Traditional backward polygon beam tracing models the light transport paths as radiated from a point light source, then scattered by a specular polygonal surface and finally scattered (second bounce) by a Lambertian (diffuse) surface before reaching the eye. This section investigates the shape of the light volume or beam that is required to successfully model first bounce *glossy* interactions using backward polygon beam tracing. The renderer that was built to render the caustics shown in the previous section is extended to render glossy interactions.

4.1 Analysing LGDE Transport Paths

The shape of a glossy beam may be analysed via a simple Monte-Carlo photon tracing simulation. Figure 6 shows a side view cross section of the resulting photon density plots for a relatively smooth surface (left) and a rough surface (right). Photons are emitted from a point source in the top left corner of the image and reflected off a polygon surface at the bottom. The polygon is about half the width of the image and only the reflected photon density is shown. The black to white gradient in the plot indicates an increasing photon density towards the reflecting surface. The extracted contour lines are shown to clearly visualise the shape of the photon density iso-surfaces.

Although accurate, a photon tracing simulation is computationally expensive as many photons are often required for the results to converge. It is however possible to build a glossy beam model to efficiently approximate the photon tracing results as discussed next and in Section 4.2.

Figure 4 shows a triangle scatterer illuminating a fragment F on the ground plane. P is a point on the polygon scatterer. The irradiance E on a fragment within a specular beam (from a specular scatterer) may be expressed as:

$$E \quad = \quad \frac{\Phi_s}{A_\perp}\cos\theta. \tag{1}$$

Φ_s is the specular beam flux (i.e. the flux that is contained and conserved within the specular beam). Φ_s is equal to the radiant intensity I_{light} of the point light multiplied by the solid angle Ω_{light} of the triangle scatterer as measured from the light. That is:

$$\Phi_s = I_{light}\Omega_{light}.$$

$A_\perp$ is the flux triangle's area (area $A_F B_F C_F$ in Figure 5). The flux triangle lies on a flux plane that includes the fragment position, but is orthogonal to the average wavefront normal. θ is the incoming radiance angle measured from the fragment's normal.

For LSDE transport paths the energy density at F is the result of an energy contribution from a single specular scattering direction. Generalising the fragment irradiance to fragment positions within and outside of the specular beam may therefore be expressed as:

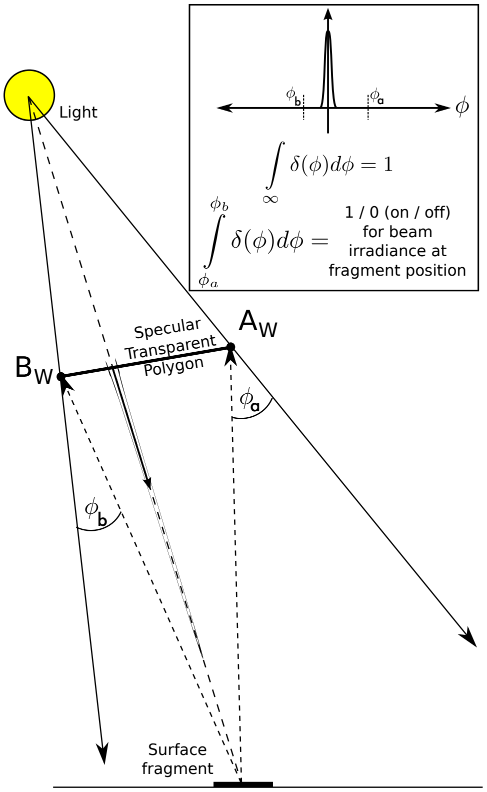
$$E = \frac{\Phi_s}{A_\perp}\cos\theta \int\limits_{\Omega} \delta(\boldsymbol{\phi})d\boldsymbol{\phi}. \tag{2}$$

The domain Ω is the domain of $\boldsymbol{\phi}$ angles over the scatterer and $\delta()$ is the Dirac delta function. Equation 2 very simply states that the fragment irradiance is given by $E = \frac{\Phi}{A_\perp}\cos\theta$ (which is Equation 1) when $\boldsymbol{\phi} = \mathbf{0}$ is included in the integration domain Ω. In other words, the fragment is illuminated when an LSDE transport path can be found that connects the fragment to the light source via the triangle scatterer. Otherwise $E = 0$ and the fragment is not illuminated. Figure 7 visually shows how, *in two dimensions*, the specular beam irradiance that is received at the fragment's position may be expressed as an integral over a Dirac delta function.

When the fragment is instead illuminated by a glossy scatterer (LGDE transport paths) the energy density can only be expressed as the result of integrating the glossy scattered contributions from *all* points on the polygon scatterer. The Dirac delta probability distribution is therefore replaced by a glossy probability distribution $\rho(\boldsymbol{\phi})$:

$$E \quad = \quad \frac{\Phi_s}{A_\perp}\cos\theta \int\limits_{\Omega} \rho(\boldsymbol{\phi})d\boldsymbol{\phi}. \tag{3}$$

Note that the specular beam flux Φ_s is still used and that the specular scatter direction is still the mean of the probability distribution.



Light
ϕ_b
ϕ_a
ϕ
$\int_{\infty} \delta(\phi) d\phi = 1$
$\int_{\phi_a}^{\phi_b} \delta(\phi) d\phi =$
1 / 0 (on / off) for beam irradiance at fragment position
Specular Transparent Polygon
A_W
B_W
ϕ_a
ϕ_b
Surface fragment

Figure 7: *A 2D side view of a fragment lit by a specular polygon source.*

However, in this case there is no longer just one specular transport path that illuminates a specific fragment. A fragment's illumination is now due to the weighted integral over *many* LGDE transport paths and the way in which the glossy paths are weighed are of importance. Figure 8 visually shows how, *in two dimensions*, the glossy beam irradiance that is received at the fragment's position may be expressed as an integral over a glossy distribution function.

One can of coarse also express E as:

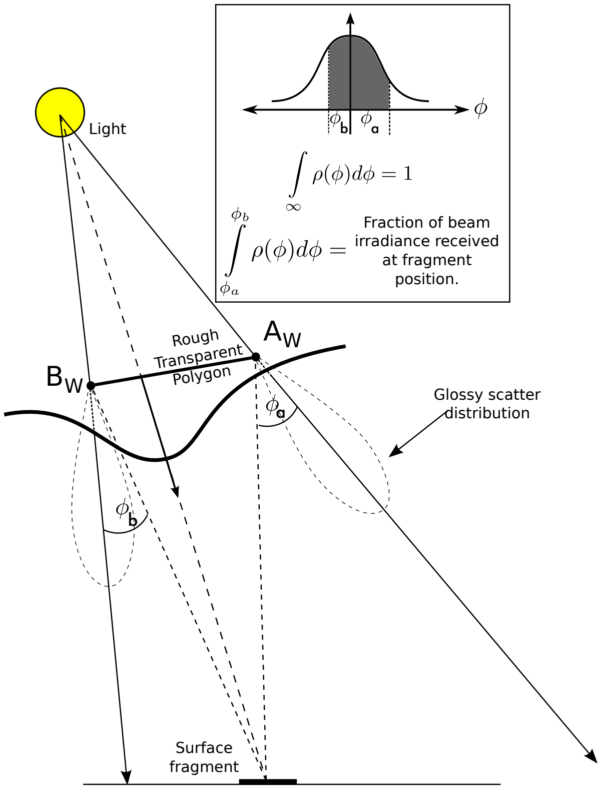
$$E \quad = \quad \frac{\Phi_s}{A_\perp} \cos\theta \int_{\overline{\Omega}} \rho(\phi(\overline{\omega}))\phi'(\overline{\omega})d\omega, \qquad (4)$$

where $\overline{\Omega}$ is the hemispherical (solid angle) integration domain.

Note however that the integral part of Equation 3 (viz. $\int_{\Omega} \rho(\boldsymbol{\phi})d\boldsymbol{\phi}$) is simply an expression for the volume over the 2D integration domain Ω and under the scatter distribution $\rho(\boldsymbol{\phi})$. The calculation of the volume may be replaced by a piecewise approximation:

$$\begin{aligned} \int_{\Omega} \rho(\boldsymbol{\phi})d\boldsymbol{\phi} \quad &= \quad \sum_{i=1}^{n} \overline{\rho_i} \int_{\Omega_i} d\boldsymbol{\phi} \\ &= \quad \sum_{i=1}^{n} \overline{\rho_i}\Omega_i, \end{aligned}$$

where $\overline{\rho_i}$ is the average ρ over Ω_i and $\Omega_1 + \Omega_2 + ... + \Omega_n = \Omega$.



Light
ϕ
ϕ_b
ϕ_a
$\int_{\infty} \rho(\phi) d\phi = 1$
$\int_{\phi_a}^{\phi_b} \rho(\phi) d\phi =$
Fraction of beam irradiance received at fragment position.
Rough Transparent Polygon
A_W
B_W
Glossy scatter distribution
ϕ_a
ϕ_b
Surface fragment

Figure 8: *A 2D side view of a fragment lit by a glossy polygon source.*

This leads to:

$$\begin{aligned} E \quad &= \quad \frac{\Phi_s}{A_\perp} \cos\theta \int_{\Omega} \rho(\boldsymbol{\phi})d\boldsymbol{\phi} \\ &\approx \quad \frac{\Phi_s}{A_\perp} \cos\theta \sum_{i=1}^{n} \overline{\rho_i}\Omega_i. \end{aligned} \qquad (5)$$

The calculation of the domain Ω is quite important and discussed further in Section 4.2. When the integration domain includes $\boldsymbol{\phi} = \mathbf{0}$ and the standard deviation is much larger than the standard deviation of ρ one finds that $\int_{\Omega} \rho(\boldsymbol{\phi})d\boldsymbol{\phi} \approx \int_{\Omega} \delta(\boldsymbol{\phi})d\boldsymbol{\phi} = 1$. Therefore, as expected, $(1) = (2) \approx (3)$ when the fragment is within a specular or relatively low variance glossy beam.

4.2 Modelling LGDE Transport Paths

The goal of this section is to describe an analytical predictive LGDE light transport model within the experimental frame of specular micro-facet materials. In other words, the model describes an approximate analytical solution to the lighting integral for all configurations of a surface fragment[1] illuminated by a specular micro-facet polygon scatterer as shown previously in Figure 4.

For the specular micro-facet material model the scatter distribution $\rho(\boldsymbol{\phi})$ is a normalised 2D Gaussian distribution. The normalised

[1] Note that this paper assumes the following definition of *a fragment*: A fragment is the part of a larger surface that is behind (and projects to) an associated pixel on the image plane. The dimensions of a fragment are usually assumed to be relatively small when compared to the dimensions of the scene.

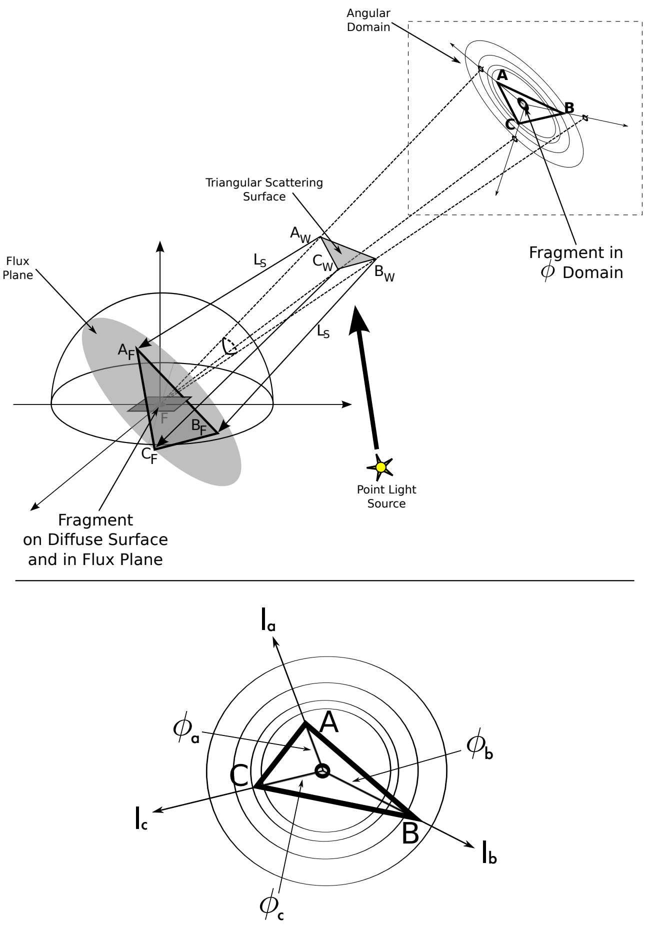


Figure 9: *The fragment in the flux plane and the triangular integration domain Ω.*

2D Gaussian function represents the probability distribution of the scattered light and the average of the distribution is the specular (smooth surface) scatter direction. It is assumed for the moment that the distribution is spherically symmetric with a specified variance.

This proposed LGDE light transport model applies Equation 5 to express the lighting integral (to gather the fragment irradiance from a glossy beam) as a function of the volume under the Gaussian probability function $\rho(\boldsymbol{\phi})$. The 2D integration domain Ω under the Gaussian function is approximated as discussed below to allow an efficient estimate of the volume under the probability distribution. The lighting integral (to gather the fragment irradiance) is then finally approximated as a function of the volume over Ω and under the 2D Gaussian probability distribution.

The following two important aspects of the light transport model are described in more detail:

- Section 4.2.1: Finding an approximate mapping from the world space domain of the polygon scatterer to a 2D triangular domain in Ω.
- Section 4.2.2: Approximating the volume over the triangular domain and under the 2D Gaussian probability distribution.

The resulting volume is in effect the probability that the fragment will receive the same irradiance that a fragment inside a specular beam would receive. The final irradiance received at the fragment's position is equal to the volume, multiplied by the specular beam irradiance on the flux plane and projected to the fragment with $\cos\theta$.

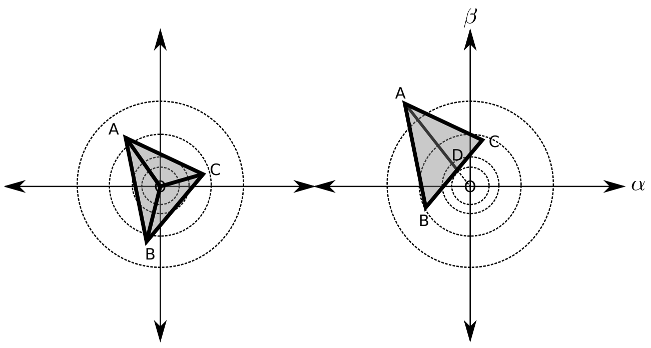


Figure 10: *The lighting integral may be approximated as a function of the volume under a normalised 2D Gaussian function over a triangular domain ABC (also shown in Figure 9). Left: The fragment, located at O, is within the specular flux triangle. Right: The fragment, located at O, is outside the specular flux triangle.*

4.2.1 Mapping from World Space to a 2D Triangular Domain

Figure 9 shows the flux triangle $A_F B_F C_F$ and the scatterer triangle vertices A_W, B_W and C_W. The scatter angles ϕ_a, ϕ_b and ϕ_c between the lines $A_W F$, $B_W F$ and $C_W F$ and their associated specular scatter directions $A_W A_F$, $B_W C_F$ and $C_W C_F$ (see angle ϕ_a and ϕ_b in Figure 8) may be directly calculated.

The goal is to now create a 2D scatter angle domain Ω as discussed in Section 4.1. Points A, B and C are chosen on the edge of the domain and placed at angular distances ϕ_a, ϕ_b and ϕ_c from the centre ($\boldsymbol{\phi} = \mathbf{0}$) to represent the scatterer vertices in the angular domain. Since the angles ϕ_a, ϕ_b and ϕ_c were measured from the fragment position, $\boldsymbol{\phi} = \mathbf{0}$ represents the fragment position in the angular domain.

Once the angular distances to the vertices A, B and C are known then only their relative directions are required to calculate their positions using direction · length. The directional position of the vertices around the centre of Ω, viz. O, are however the same as the directional positions of A_F, B_F and C_F around F as shown in Figure 9. The mean of the Gaussian distribution is only inside Ω when the fragment is within the specular flux triangle and outside otherwise.

To construct the A, B and C vertices on the edge of Ω, the normalised direction vectors ($\overline{l_a} = \frac{A_F - F}{\|A_F - F\|}$, $\overline{l_b} = \frac{B_F - F}{\|B_F - F\|}$ and $\overline{l_c} = \frac{C_F - F}{\|C_F - F\|}$) from the fragment position to each of the vertices in the flux plane are first calculated. The vertex A is finally expressed as $A = \phi_a \cdot \overline{l_a}$. Similarly vertex B may be expressed as $B = \phi_b \cdot \overline{l_b}$ and vertex C as $C = \phi_c \cdot \overline{l_c}$. Triangle ABC therefore lies on a plane parallel to the flux plane, but now in a frame of reference that has the fragment position at its centre O. The distances AO, BO and CO are angles and equal to ϕ_a, ϕ_b and ϕ_c respectively.

The assumption is made that the domain Ω may be approximated by the triangular domain ABC as shown in Figure 10

4.2.2 Approximating the Volume under a 2D Gaussian

The volume over the triangular domain under a 2D Gaussian distribution does not have an analytical solution. The Gaussian surface

1. First, find the fragment's probability value P_O at the centre of the distribution:
   $P_O = \frac{1}{2\pi\sigma^2}$
2. Then, find the volume P_E:
   $P_E = 0.5 * \|\overline{AOB} \cdot P_{AOB} + \overline{BOC} \cdot P_{BOC} + \overline{COA} \cdot P_{COA}\|$.
   P_{AOB} is the average of the probability values at A, O (centre) and B.
   $\overline{AOB} = \phi_a \cdot \overline{l_a} \times \phi_b \cdot \overline{l_b}$ as discussed in Section 4.2.2.
3. Finally, the irradiance E is calculated as:
   $E = \frac{\Phi_s}{A_\perp} \cos\theta \ \min(P_E, 1.0)$ [see Equation 3].
   θ is the incoming irradiance angle, and Φ_s and $A_\perp$ are the CPU calculated beam flux and flux triangle area as discussed above.

Figure 11: *Calculation of the volume P_E and the irradiance E when the fragment is within the flux triangle.*

may however be approximated by a polygon mesh and the volume under a polygon is simply the area of its 2D domain times the average probability (see Equation 5). The volume under the 2D Gaussian function is currently approximated by tessellating the surface over the triangular domain ABC as shown in Figure 10.

To prevent visual anomalies for a single triangle scatterer an important consideration for the volume calculation is that there should be no discontinuities between the volumes calculated for fragment positions within and outside of the flux triangle. To prevent visual anomalies for a plate of polygon scatterers a second important consideration for the volume calculation is to approximate the scatter distribution as close as possible. Note that this statement holds regardless of the distribution, but a Gaussian distribution approximates the specular micro-facet reality.

When the fragment is within the specular flux triangle then the triangular domain includes the peak of the Gaussian distribution as shown on the left in Figure 10. In this case three polygons are used to approximate the Gaussian surface.

When the fragment is outside of the specular flux triangle then the triangular domain does not include the peak of the Gaussian distribution. The edge closest to the peak may however contain a local maximum as shown on the right in Figure 10. In this case two polygons are used to approximate the Gaussian surface. The edge that is intersected at the point D by one of the $\phi_a \cdot \overline{l_a}$, $\phi_b \cdot \overline{l_b}$ or $\phi_c \cdot \overline{l_c}$ vectors is split there to more closely match the Gaussian surface profile.

The area of each triangular domain is calculated by using the cross product rule. For example, when the fragment is within the flux triangle then

$$Area\ ABO = 0.5 * \|(\phi_a \cdot \overline{l_a}) \times (\phi_b \cdot \overline{l_b})\|.$$

When the fragment is outside the flux triangle then

$$Area\ ABD = 0.5 * \|(a \cdot \overline{l_a}) \times (b \cdot \overline{l_b}) + (b \cdot \overline{l_b}) \times (d \cdot \overline{l_d})\|.$$

The volume under all the polygons are finally calculated and the result clamped to unity. The tessellated surface however only approximates the 2D Gaussian function. A better approximation could lead to a more accurate volume calculation and results that approximate reality more closely.

4.3 The GPU Implementation

The GPU implementation of the LGDE backward polygon beam tracer is an extension of the LSDE GPU implementation mentioned in Section 3. The specular polygon beams are generated on the CPU. The CPU code traces rays from the light source through the vertices of each scatterer to generate the reflected, refracted and shadow rays. The scattered and shadow rays are then grouped into polygon beams. OpenMP is used to parallelise the beam construction over the available CPU cores.

1. Firstly, given the flux triangle vertices A_F, B_F and C_F in the flux plane and the flux triangle normal N_F, the flux triangle edge normals $\overline{N_0}, \overline{N_1}, \overline{N_2}$ are calculated as follows:
   $$N_0 = (B_F - A_F) \times N_F,$$
   $$N_1 = (C_F - B_F) \times N_F,$$
   $$N_2 = (A_F - C_F) \times N_F.$$
2. Given the fragment position F, if $((F - C_F) \cdot (N_1) < 0.0)$ and $((F - C_F) \cdot (N_2) < 0.0)$ then the split point D (see Figure 10) is on edge AB. Similar tests are done to see if D doesn't instead fall on edges BC or CA.
3. If the split point is on an edge: Then, the intersection vertex D on the flux plane, viz. D_F, is used to parameterise the triangle edge with a scalar, s. The parameter is then used to calculate the vertex D_W on the scatterer triangle and the specular scatter direction S_D from D_W. S_D, D_W and D_T are used to find P_D and $\phi_d \cdot \overline{l_d}$. Once P_D and $\phi_d \cdot \overline{l_d}$ are known then if D splits edge AB:
   $$\begin{aligned} P_E = \quad & 0.5 * \|\overline{COA} + \overline{AOD}\| \cdot P_{CAD} \quad + \\ & 0.5 * \|\overline{BOC} + \overline{DOB}\| \cdot P_{CAD} \end{aligned}$$
   A similar P_E calculation is performed when D splits edge BC or CA instead.
4. If there is not a split point (when A, B or C is closer to the fragment than a split point D would be):
   $P_E = 0.5 * \|\overline{AOB} + \overline{BOC} + \overline{COA}\| \cdot P_{ABC}$ is simply given by the volume over the triangular domain ABC.
5. Finally, the irradiance E is calculated as:
   $E = \frac{\Phi_s}{A_\perp} \cos\theta \ \min(P_E, 1.0)$ [see Equation 3].
   θ is the incoming irradiance angle, and Φ_s and $A_\perp$ are the CPU calculated beam flux and flux triangle area as discussed above.

Figure 12: *Calculation of the volume P_E and the irradiance E when the fragment is outside of the flux triangle.*

Once constructed the polygon beams are packed into an OpenGL texture for upload to the GPU. Each beam currently consumes 8 RGBA texels to store all of its attributes. The beam attributes required by the shader are:

- The specular red, green and blue beam flux ($\Phi_s = I_{light} \cdot \Omega_{light}$ calculated by the CPU) and the generator's micro-facet variance,
- the generator plane's normal and offset,
- the three polygon vertices, and
- the three scattered or shadow vectors.

For each fragment, the GPU shader then linearly iterates over all the polygon beams. The linear traversal should however in future be replaced with a hierarchical traversal similar to the cone hierarchy traversal used by Chuang and Cheng [1995]. This would allow the early culling of groups of beams which do not contribute to the current fragment.

For each beam the flux plane and flux triangle is calculated and used to test whether the fragment is inside or outside the specular beam. Once a beam's flux triangle is found and the in-beam test has happened the irradiance that the beam contributes to the fragment's surface may be calculated. When the fragment is within the flux triangle then the shader calculates the volume P_E and the irradiance E as shown in Figure 11. However, when the fragment is outside of the flux triangle then the shader calculates the volume P_E and the irradiance E as shown in Figure 12.

4.4 Results

Figure 13 contain results of modelling transmissive glossy interactions. It shows a stained glass window lit from behind with varying roughness of the glass. The variance of the micro-facet distribution is increased to the right as if sandblasting the window with a courser and courser grain. Note that the energy of the glossy beams have been elevated for clarity.

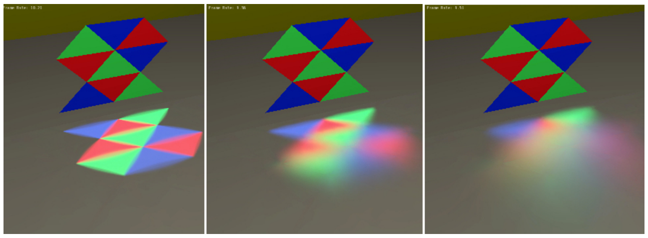


Figure 13: *Transmitted glossy beams from glass with small to large micro-facet variance.*

Figure 14 shows how glossy reflecting surfaces may also be approximated by the micro-facet material model. The left figure shows a very large micro-facet variance which results in an extremely fuzzy (glossy) reflection from the pattern. The right figure shows an even larger micro-facet variance which results in a surface with reflection properties that approach that of a diffuse surface.

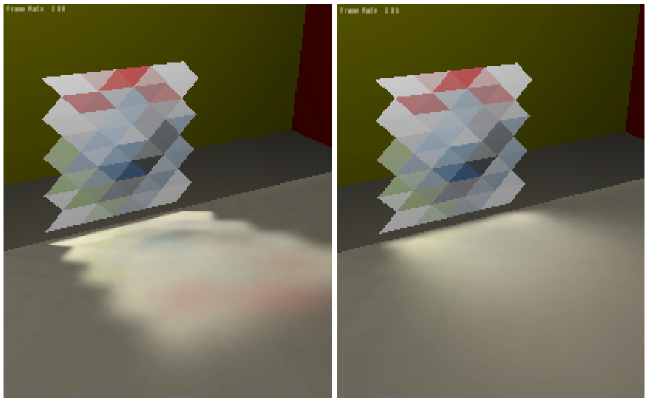


Figure 14: *Reflected glossy beams from a stained glass window with very large micro-facet variance.*

The simulation performance on a Macbook Pro 13" notebook with a 2.26 GHz Intel Core 2 Duo CPU and an NVIDIA Geforce 9400M GPU is around 6 frames per second for the scene shown in Figure 13 at a resolution of 640x480 pixels. The scene shown in Figure 14 achieves a performance of around 1 frame per second at the same resolution. The current performance bottleneck appears to be the shader execution time which is due to the linear scene traversal. Table 1 gives more results for the second scene which shows the near linear dependence of performance on the number of GPU cores.

Table 1: *Performance Results for the scene shown in Figure 14*

GPU	# Cores	Performance
9400M	16	1.8 FPS
9600GT	64	10 FPS
GTS 250	128	20 FPS
GTX285	240	39 FPS

4.5 Analysis and Limitations

The L(S|G)DE model is physically based and uses an established specular micro-facets material model, but includes the following approximations to allow a simpler model and an efficient implementation:

- The scatter distribution is currently assumed to be radially symmetric.
- The beam flux density is assumed to be constant over any beam cross section which is orthogonal to the average wavefront normal.
- The integration domain Ω is approximated by a triangular domain ABC.
- The 2D Gaussian function is approximated by a low number of polygons that are adaptively tessellated over the triangular domain ABC.

Removing or improving any of the above approximations would improve the fidelity of the glossy backward polygon beam model to better approximate reality.

The current implementation uses a linear beam traversal. The implementation's performance is therefore linearly related to the number of scattering polygons and not really scalable to large scenes.

Currently the light field is only implicitly specified by the glossy light volume. The light volumes do therefore not encode enough information to deduce the light field's angular distribution. This information is however required to model second order shadows from glossy scatterers and many-bounce interactions.

A further limitation is that surface details are only accurately represented by the light volumes if the surface geometry matches the scale of the surface details. Including high frequency surface detail therefore requires a micro-polygon surface tessellation step to set up the light volumes, although the micro-polygons need not be rendered.

5 Conclusion

5.1 Discussion

A very large number of photons or rays are typically required to render LGDE transport paths with techniques such as ray tracing and photon mapping. It is therefore traditionally very difficult to render these transport paths at interactive frame rates.

Backward polygon beam tracing has successfully been extended to model L(S|G)DE transport paths. The efficiency of backward beam tracing in gathering path coherency was also exploited to render these transport paths at interactive frame rates on entry-level hardware. Extending backward polygon beam tracing to include glossy light transport paths is a step towards efficiently implementing second bounce global illumination.

The remaining limitations of L(S|G)DE backward polygon beam tracing are:

- The listed approximations that limit the model fidelity.
- The O(n) performance of the current linear beam traversal implementation.
- Many bounce glossy interaction cannot yet be modelled.
- Second order shadows are not yet included in the light transport model.
- Including high frequency surface detail requires a micropolygon surface tessellation step to set up the light volumes.

5.2 Future Work

A rigorous account of energy, to qualify the LGDE transport model as physically based and approximating reality, is still outstanding. Once this is done the implementation should be verified against the qualified model and finally operationally validated against reality. The listed approximations in building the model may then be improved as required.

Adding a non-zero diffuse component to the material model and using a more accurate micro-facet distribution as discussed by Cook and Torrance [1982] should also be investigated in future to improve the fidelity of the glossy beam tracing model. The inclusion of many bounce glossy transport paths and second order shadows (polygons obstructing the light volumes) should also be investigated.

A beam hierarchy similar to the cone hierarchy used by Chuang and Cheng [1995] is required for efficient traversal of the beam set when many hundreds of beams are used in larger scenes. OpenMP is currently used to generate the beams concurrently on the multiple processor cores of the host CPU. The use of CUDA and OpenCL is however being considered to accelerate the beam construction and potentially the forward rendering pass.

References

ARVO, J. 1986. Backward ray tracing. In *Developments in Ray Tracing, SIGGRAPH Course Notes, Vol. 12.*

BRIÈRE, N., AND POULIN, P. 2000. Adaptive representation of specular light flux. In *Graphics Interface*, 127–136.

CHUANG, J.-H., AND CHENG, S.-A. 1995. Computing caustic effects by backward beam tracing. *The Visual Computer 11*, 3, 156–166.

COLLINS, S. 1995. Reconstructing of illumination from area luminaires. In *Rendering Techniques*, 274–283.

COOK, R. L., AND TORRANCE, K. E. 1982. A reflectance model for computer graphics. *ACM Trans. Graph. 1*, 1, 7–24.

ERNST, M., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Interactive rendering of caustics using interpolated warped volumes. In *GI '05: Proceedings of Graphics Interface 2005*, Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 87–96.

HECKBERT, P. S., AND HANRAHAN, P. 1984. Beam tracing polygonal objects. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 119–127.

HECKBERT, P. 1990. Adaptive radiosity textures for bidirectional ray tracing. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 145–154.

IHRKE, I., ZIEGLER, G., TEVS, A., THEOBALT, C., MAGNOR, M., AND SEIDEL, H.-P. 2007. Eikonal rendering: efficient light transport in refractive objects. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, ACM, New York, NY, USA, 59.

IWASAKI, K., DOBASHI, Y., AND NISHITA, T. 2002. An efficient method for rendering underwater optical effects using graphics hardware. *Computer Graphics Forum 21*, 4, 701–711.

IWASAKI, K., DOBASHI, Y., AND NISHITA, T. 2003. A fast rendering method for refractive and reflective caustics due to water surfaces. *Computer Graphics Forum 23*, 3, 601–609.

KAJIYA, J. T. 1986. The rendering equation. *SIGGRAPH Comput. Graph. 20*, 4, 143–150.

MCGUIRE, M., AND LUEBKE, D. 2009. Hardware-accelerated global illumination by image space photon mapping. In *HPG '09: Proceedings of the Conference on High Performance Graphics 2009*, ACM, New York, NY, USA, 77–89.

NISHITA, T., AND NAKAMAE, E. 1994. Method of displaying optical effects within water using accumulation buffer. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 373–379.

NISHITA, T., MIYAWAKI, Y., AND NAKAMAE, E. 1987. A shading model for atmospheric scattering considering luminous intensity distribution of light sources. *SIGGRAPH Comput. Graph. 21*, 4, 303–310.

SHAH, M. A., KONTTINEN, J., AND PATTANAIK, S. 2007. Caustics mapping: An image-space technique for real-time caustics. *IEEE Transactions on Visualization and Computer Graphics 13*, 2, 272–280.

SPENCER, B., AND JONES, M. W. 2009. Into the blue: Better caustics through photon relaxation. In *Eurographics 2009: Proceedings of the 30th Annual Conference of the European Association for Computer Graphics*.

SUN, X., ZHOU, K., STOLLNITZ, E., SHI, J., AND GUO, B. 2008. Interactive relighting of dynamic refractive objects. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, ACM, New York, NY, USA, 1–9.

TORRANCE, K. E., AND SPARROW, E. M. 1967. Theory for off-specular reflection from roughened surfaces. *Journal of the Optical Society of America 57*, 9, 1105–1114.

UMENHOFFER, T., PATOW, G., AND SZIRMAY-KALOS, L. 2008. Caustic triangles on the gpu. In *CGI 2008: Proceedings of the 26th Computer Graphics International Conference*.

WATT, M. 1990. Light-water interaction using backward beam tracing. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 377–385.

WYMAN, C. 2008. Hierarchical caustic maps. In *I3D '08: Proceedings of the 2008 symposium on Interactive 3D graphics and games*, ACM, New York, NY, USA, 163–171.

ZHOU, K., HOU, Q., WANG, R., AND GUO, B. 2008. Real-time kd-tree construction on graphics hardware. *ACM Trans. Graph. 27*, 5, 1–11.