DOI: 10.1111/j.1467-8659.2011.01903.x

COMPUTER GRAPHICS *forum*
Volume 0 (2011), number 0 pp. 1–12

Extending Backward Polygon Beam Tracing to Glossy Scattering Surfaces

B. Duvenhage,[1] K. Bouatouch[2] and D. G. Kourie[1]

[1]University of Pretoria, Pretoria, South Africa
dkourie@cs.up.ac.za, bduvenhage@csir.co.za
[2]University of Rennes 1, Rennes, France
kadi.bouatouch@irisa.fr

Abstract

Backward polygon beam tracing methods, that is beam tracing from the light source (L), are well suited to gather path coherency from specular (S) scattering surfaces. These methods are useful for modelling and efficiently simulating caustics on diffuse (D) surfaces; an effect due to LS^+D transport paths. This paper generalizes backward polygon beam tracing to include a glossy (G) scattering surface. To this end the details of a beam tracing lumped model and implementation of $L(S \mid G)D$ transport paths are presented. Although we limit the discussion to short transport paths, we show that backward beam tracing is faster than photon mapping by an order of magnitude for rendering caustics from glossy and specular surfaces.

Keywords: backward beam tracing, caustics, glossy scattering, lumped transport model

ACM CCS: I.3.7 [Computer Graphics]: Three Dimensional Graphics and Realism—Color, shading, shadowing, and texture.

1. Introduction

Watt [Wat90] and others [EAMJ05] [BP00] have used backward polygon beam tracing to efficiently simulate caustics such as shown in Figure 1. Caustics are due to transport paths from the light source (L) that contain a sequence of specular (S) surface interactions and a final diffuse (D) interaction. Using Heckbert's [Hec90] regular expression notation, the modelled transport paths are described with the expression LS^+D. Note that we use the description of *backward* tracing to refer to ray or beam tracing from the light source as opposed to *forward* tracing from the eye.

The reason that specular transport paths can be lumped into beams and efficiently simulated is that specular paths are coherent. Efficiently simulating general light transport paths that include glossy (G) and D surface interactions is however difficult to do because of

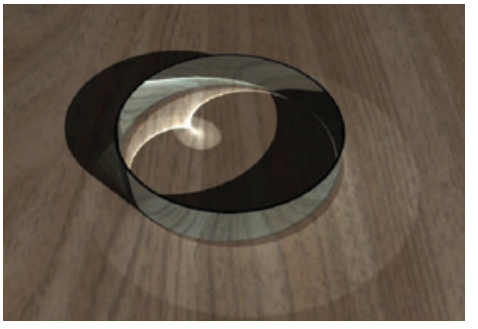

Figure 1: *A cardioid caustic (75 × 5 scattering surfaces) rendered with our implementation of backward polygon beam tracing.*

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd. Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

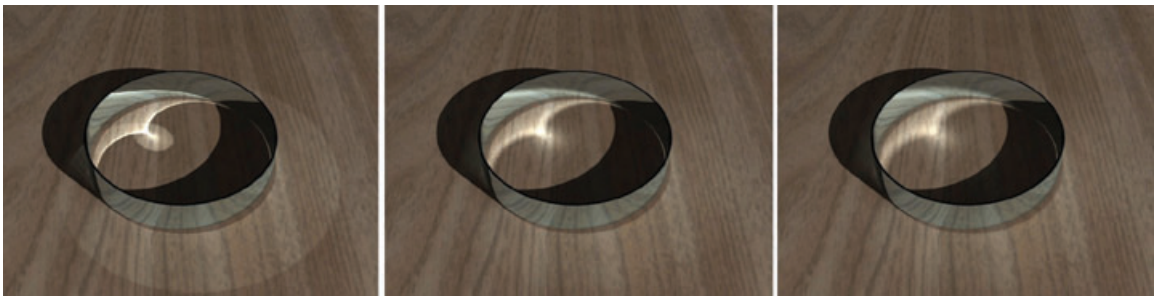

Figure 2: *A cardioid caustic (75 × 5 scattering surfaces) rendered with our glossy backward polygon beam tracing technique. Ring roughness (for caustic transport paths) is increased from left to right.*

- The large transport path domain (i.e. solution space) of the rendering equation, and
- the loss of coherency for diffuse and glossy transport paths.

The general problem experienced in simulating global illumination is consequently one of inefficiency because of our inability to lump glossy and diffuse transport paths into groups such as beams.

In this paper, we explore the use of backward polygon beam tracing to model and efficiently simulate dynamic glossy transport paths, that is LGD paths. The inherent efficiency of the backward polygon beam tracing technique in gathering path coherency is the main motivation for investigating the technique. A forward renderer is required to connect the LGD paths to the eye (E) and Figure 2 shows a cardioid caustic due to the LGDE transport paths under consideration.

In particular in this paper we

- analyse the shape of single scatter glossy beams,
- describe a backward polygon beam tracing model of L(S | G)D transport paths (the main contribution) and
- present a two pass beam trace and forward render implementation of this new lumped light transport model.

We demonstrate a ray trace forward renderer with what we refer to as a hierarchical cone bounding volume (HCBV) acceleration structure. We do nonetheless also present a reference to a rasteriser forward renderer that uses a brute force beam processing scheme that is more suited to an OpenGL or a DirectX GPU implementation. The glossy backward polygon beam tracing research is a follow-up of the work we presented in [DBK10]. We believe that extending backward polygon beam tracing to include more general light transport paths is a step towards efficiently implementing global illumination simulations using such lumped transport models.

Section 2 summarizes the related work done on backward polygon beam tracing. The details and limitations of a typical LSDE backward polygon beam tracing implementation are then discussed in Section 3. This section is followed by the details of our glossy (LGDE) backward polygon beam tracing technique in Section 4. Section 5 then presents the results before the paper is concluded in Section 6.

2. Related Work

Watt's [Wat90] backward polygon beam tracing method is related to Heckbert and Hanrahan's polygon beam tracing [HH84] and to Arvo's backward raytracing [Arv86]. Heckbert and Hanrahan [HH84] 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. Arvo [Arv86] used backward raytracing to render caustics that could not otherwise be simulated efficiently using the high fidelity forward raytracing and radiosity rendering techniques of the time.

Similar to what Heckbert and Hanrahan proposed, Watt [Wat90] used backward polygon beam tracing to improve upon the rendering of caustics by exploiting the spatial coherency of polygons. He used projected caustic surface detail polygons, light beams (similar to Nishita *et al.* [NMN87]) and a raytrace forward renderer to simulate single scatter caustics and single scattering of the light beams in a participating medium.

Nishita and Nakamae [NN94] used a scan-line based renderer with an accumulation buffer and light beams to simulate caustics without the need for a forward raytrace pass. Their method can produce caustics on curved surfaces and includes shadows by making use of the z-buffer.

Chuang and Cheng [CC95] can also handle non-polygonal illuminated surfaces by finding the light beams within which any surface fragment resides. A fragment may be defined as a part of a larger surface that projects to an associated pixel on the image plane and the dimensions of a fragment is relatively small when compared with the dimensions of

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

the scene. Chuang and Cheng's light beams are enclosed in a hierarchy of bounding cones for more efficient point-in-beam detection.

Briere and Poulin [BP00] used a light image to adaptively refine and construct light beams from the light source. 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.* [IDN02] made use of GPUs to accelerate Nishita and Nakamae's [NN94] beam tracing method. They preserved the abilities to render caustics on curved surfaces and to include shadows. Iwasaki *et al.* [IDN03] proposed an extension to the work 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 implemented reflection and refraction mapping of the caustic slices to render objects as seen in a reflection or below a refractive fluid surface.

Ernst *et al.* [EAMJ05] also made use of backward polygon beam tracing. They used warped polygon beam volumes, interpolation of beam energies and a GPU implementation to improve the quality and the execution performance. Ernst *et al.* did not use a beam hierarchy for accelerating their GPU implementation. They instead exploited the GPU's ability to render polygons and drew a bounding volume around each beam. Fragments are then evaluated against a beam only if it is within the screen-space of the bounding volume.

The listed backward beam tracing methods model only LS^+D transport paths. More recent methods [IZT*07] [SKP07] [Wym08] [SZS*08] [UPSK08] [ZHWG08] [SJ09] [ML09] for rendering LS^+D transport paths all focus on accelerated photon-tracing and photon-mapping with impressive results in terms of both quality and performance. In particular, progressive photon mapping [HOJ08] refines the light transport simulation results with each rendered frame. Progressive photon mapping is able to provide an efficient estimate of even $L(S \mid G \mid D)^+DE$ paths, but the solution is still created with individual photons and can take quite a number of iterations to converge.

3. Backward Polygon Beam Tracing

Our implementation of the backward polygon beam tracing technique models and simulates LSD transport paths, but we follow an approach similar to Chuang and Cheng [CC95] in which each diffuse receiver surface fragment is lit by finding the light beams within which it resides. We adapt a Whitted [Whi80] forward ray tracer to connect to the eye (E) the LSD paths of each scattered beam. Therefore, much the same as Chuang and Cheng, we make use of what we refer to as an HCBV acceleration structure to optimize beam processing. If an OpenGL or DirectX rasterizer forward renderer is used instead then a screen-space bounding volume approach with a highly optimized inner render loop such as proposed by Ernst *et al.* [EAMJ05] might be more appropriate.

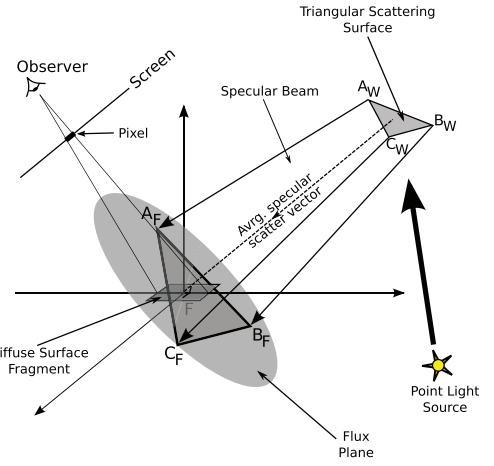

Figure 3: *The flux plane and flux triangle $A_F B_F C_F$.*

3.1. Modelling LSDE transport paths

We model light-to-specular transport paths with polygon beams between the light source and each specular scattering surface. Such a surface may be specular reflective or it may be specular transmissive with a specified index of refraction. A scattered light beam is modelled by individually scattered light vectors from each of the polygon vertices (A_W, B_W and C_W) as shown for a reflected beam in Figure 3. Each set of scattered vectors is associated with a beam flux that is conserved within the beam. Propagating the beam as a set of associated vectors instead of a swept polygon shape with planar sides implicitly results in a warped caustic volume [EAMJ05]. An advantage of doing it this way is that the smooth surface (as represented by the mesh normals) and the scattered light beams are appropriately decoupled from the polygon surface mesh.

During the second (forward rendering) pass each diffuse surface fragment is evaluated for whether or not it is in any scattered beams and as a result lit via an LSD transport path. Whether or not a fragment is in a specific beam is tested by first intersecting the scattered vectors with a virtual flux plane. The flux plane is perpendicular to the average specular scatter direction and includes the fragment's world position. The intersection points are connected into a specular flux polygon (triangle $A_F B_F C_F$ in Figure 3) at which point the fragment-in-beam test is reduced to whether or not the fragment position (F) is on the flux polygon. The beam's cross-sectional area (area $A_F B_F C_F$) is used to calculate the in-beam flux density (the cross–sectional $W\,m^{-2}$) and the cosine rule is used to calculate the resulting irradiance at the fragment's position.

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

We assume that the average incoming light direction is representative of the incoming directions over the virtual flux polygon. This is however valid only if the specular scattering polygon's solid angle from the light source is small. Under this same condition we also assume that the reflectivity (due to surface colour, Fresnel effects, shadowing, and masking, etc.) is constant over the scattering polygon and that the flux density is constant over the virtual flux polygon. The use of a virtual flux polygon instead of a caustic triangle does however allow simple tracking of the wavefront flux density. This results in a smooth interpolation of irradiance over the caustic receiver. Unlike Ernst *et al.* [EAMJ05] we do not explicitly average the flux density over neighbouring beams and it might still happen that a virtual flux triangle has an area of zero (and consequently a flux density of infinity) for certain fragments. In practice this is not a problem if the beams are generated with four or more vertices.

All objects in the scene are modelled with brush geometry (convex polytopes) defined from binary space partition planes. The scattered beams are generated by the CPU directly from the brush faces and simple local visibility is included by scattering only from the front side of a brush face. A bounding cone is also set up for each scattered beam that includes all the brush face's scattered vertex vectors. The cone bounding volumes are stored in an HCBV acceleration using a two-dimensional (2D) kd-tree. The kd-tree spatially partitions the cones according to beam origin under the assumption that co-located beams would have a similar direction and field of view (FOV). As with Chuang and Cheng's approach, one HCBV structure is built per point light source.

The bounding cones around each leaf in the tree (i.e. around each scattered beam) are recursively merged along the branches of the tree towards the root. This cone hierarchy then provides the space partition required for efficient tree-traversal. Therefore, the smaller the overlap between neighbouring bounding cones can be made, the more efficient the tree-traversal becomes. In our implementation, the performance of the hierarchical processing is much faster than linear beam processing, provided that the scattered beam set is more than approximately 1000 beams. For smaller beam sets, the hierarchical execution performance is comparable to that of linear beam processing.

To accelerate the forward ray tracer's brush/scene traversal, the spherical bounding volumes of the brushes are stored in a hierarchical sphere bounding volume acceleration structure using a 2D kd-tree. The bounding spheres at each level in the tree provide the hierarchical space partition required for efficient scene traversal.

3.2. Summary of limitations

The advantage of previous methods of being able to illuminate any type of diffuse receiver, whether it be curved or flat, is preserved. This backward polygon beam tracing model of light transport does however have the limitation that the specular scattering polygon's solid angle from the light source should be small. This would require that scattering surfaces close to the light source be appropriately subdivided.

The biggest limitation is of course that backward polygon beam tracing models scattering only from specular surfaces, whereas most real surfaces such as car paint and porcelain are not perfectly specular but at least slightly glossy.

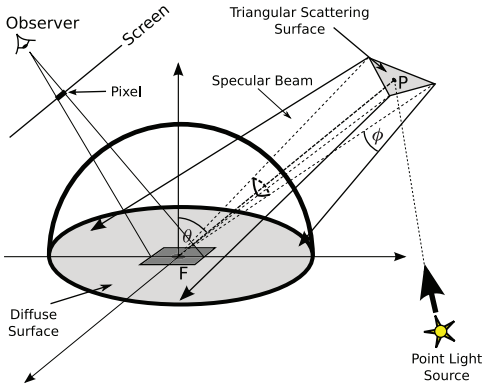

Figure 4: *A fragment lit by a polygon scatterer source.*

4. 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. We generalize this model by investigating the shape of the light beam that is required to model single scatter glossy interactions.

The glossy material model used in this paper is one of specular microfacets with a negligible diffuse component. Torrance and Sparrow [TS67] assumed such microfacets' gradients are random with a Gaussian distribution around the average surface normal. Although this assumption has since been improved upon by Cook and Torrance [CT82], we still make the Gaussian assumption for the sake of simplicity. We refer to a specular material with a significant microfacet variance as a Glossy material (G interaction), whereas an S interaction is reserved to refer to an interaction from a specular material such as a mirror with an insignificant microfacet variance.

4.1. Analysing LGDE transport paths

Figure 4 shows a polygon scatterer illuminating a fragment F on the ground plane. P is a point on the scatterer. The

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

average flux density (or irradiance $\overline{E_T}$) over the virtual flux polygon ($A_F B_F C_F$ in Figure 3) is simply equal to the total flux divided by the surface area of the polygon, that is

$$\overline{E_T} = \frac{\Phi_s}{A_\perp},$$

where $A_\perp$ is the flux polygon's area and Φ_s is the specular beam flux. The flux polygon lies on a flux plane that includes the fragment position, but is orthogonal to the average specular scatter direction. The flux reflected by $A_F B_F C_F$ from the point light is conserved within the scattered beam because of the coherency of the specular transport paths. Φ_s is consequently equal to the radiant intensity I_{light} of the point light multiplied by the solid angle Ω_T of the polygon scatterer as measured from the light. That is

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

For a narrow specular beam (i.e. a small Ω_T), the incoming radiance angle θ over the fragments within the beam is approximately constant. For simplicity we therefore assume that the average specular scatter direction may be used to approximate the fragment irradiance E_F (flux per unit surface area) for all fragments illuminated by specular beams, that is

$$\begin{aligned} E_F &= \overline{E_T}\cos\theta, \\ E_F &= \frac{\Phi_s}{A_\perp}\cos\theta, \end{aligned} \tag{1}$$

where θ is the incoming radiance angle between the fragment's surface normal and the average specular scatter direction. For LSDE transport paths, the energy density at F is the result of an energy contribution from a single specular scattering direction. The fragment irradiance within and outside of the specular beam may therefore be expressed as

$$E_F = \frac{\Phi_s}{A_\perp}\cos\theta \int_\Omega \delta(\vec{\phi})\,\mathrm{d}\vec{\phi}. \tag{2}$$

As shown in Figure 4, ϕ is the angle between PF (the transport path under consideration) and the specular scatter direction. The domain Ω is the domain of $\vec{\phi}$ angles over the scatterer and $\delta()$ is the Dirac delta function. $\int_\Omega \delta(\vec{\phi})\,\mathrm{d}\vec{\phi} = 1$ when $\vec{0} \in \Omega$ and $\int_\Omega \delta(\vec{\phi})\,\mathrm{d}\vec{\phi} = 0$ when $\vec{0} \notin \Omega$. The fragment irradiance given by Equation (2) therefore reduces to $E_F = \frac{\Phi}{A_\perp}\cos\theta$ [Equation (1)] when $\vec{0} \in \Omega$ and to $E_F = 0$ when $\vec{0} \notin \Omega$. In other words, the fragment is illuminated only when an LSDE transport path can be found that connects the fragment to the light source via the scatterer. Figure 5 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

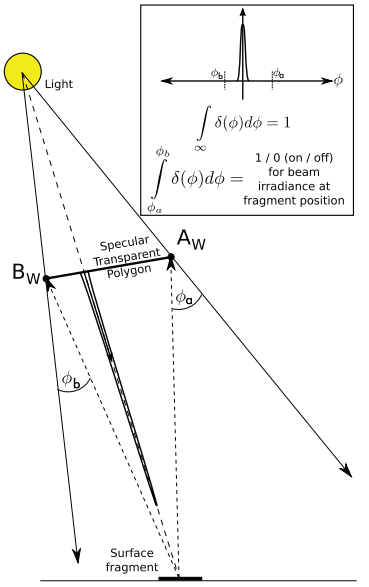

Figure 5: *A 2D side view of a fragment lit by a specular polygon source. Note that a specular transmissive polygon is shown to simplify the drawing.*

Dirac delta probability distribution is therefore replaced by a glossy probability distribution $\rho(\vec{\phi})$

$$E_F = \frac{\Phi_s}{A_\perp}\cos\theta \int_\Omega \rho(\vec{\phi})\,\mathrm{d}\vec{\phi}. \tag{3}$$

A fragment's illumination is now due to the weighted sum over many LGDE transport paths. Figure 6 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. Note Φ_s still represents the flux carried in the beam and that the specular scatter direction is the mean of the probability distribution. Also, because of the widening of the beam a fragment can be within the glossy beam even though it would be outside of the specular flux beam.

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

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

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

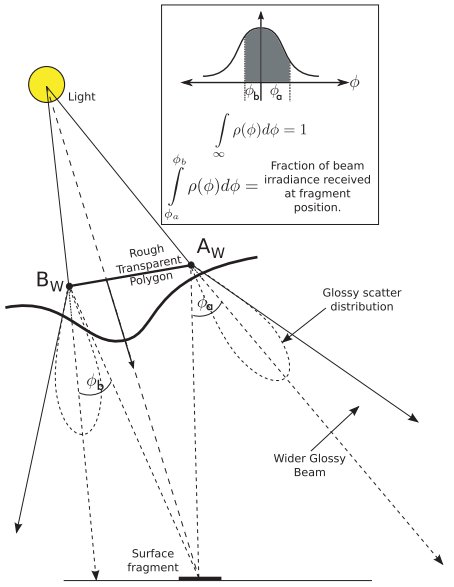

Figure 6: *A 2D side view of a fragment lit by a glossy polygon source. Note that a glossy transmissive polygon is shown to simplify the drawing.*

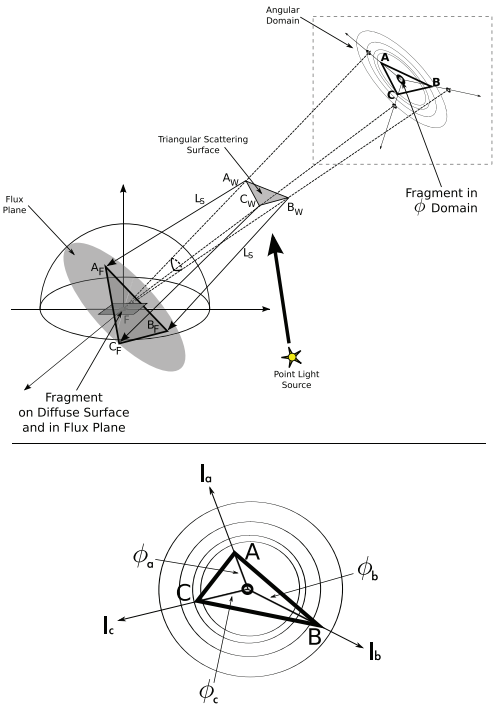

Figure 7: *The fragment in the flux plane and a triangular integration domain* Ω.

where ρ_i is the probability over Ω_i, $\overline{\rho_i}$ is the average of the probability function over Ω_i and $\Omega_1 + \Omega_2 + \cdots + \Omega_n = \Omega$.

For our purpose this leads to

$$E_F = \frac{\Phi_s}{A_\perp} \cos\theta \int_\Omega \rho(\vec{\phi})\, d\vec{\phi} = \frac{\Phi_s}{A_\perp} \cos\theta \sum_{i=1}^{n} \rho_i . \quad (4)$$

4.2. Modelling LGDE transport paths

Equation (4) is applied to express the per fragment irradiance from a glossy beam as a function of $P_E = \sum_{i=1}^{n} \rho_i$. The final irradiance received at the fragment's position is equal to P_E, multiplied by the specular beam irradiance on the flux plane and projected onto the fragment with $\cos\theta$.

For the glossy material model, the scatter distribution $\rho(\vec{\phi})$ is a normalized 2D Gaussian distribution. The average of the distribution is the specular (smooth surface) scatter direction and it is assumed that the scatter distribution is spherically symmetric with a constant variance over the scatterer.

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

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

4.2.1. *Mapping from world space to a 2D triangular domain*

Figure 7 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 6) may be directly calculated.

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 ($\vec{\phi} = \vec{0}$) to represent the scatterer vertices in the angular domain. Because the angles ϕ_a, ϕ_b and ϕ_c were measured from the fragment position, $\vec{\phi} = \vec{0}$ 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. The relative directions of the vertices around the centre of Ω, viz. O, are taken to be the same as the relative directions of A_F, B_F and C_F around F (shown in Figure 7). The mean of the Gaussian distribution

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

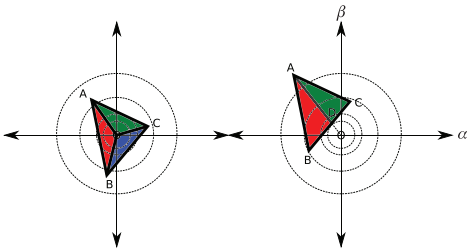

Figure 8: *The lighting integral may be approximated as a function of the volume over a triangular domain ABC. Left: The fragment, located at O, is within the specular flux triangle. Right: The fragment, located at O, is outside the specular flux triangle.*

is inside Ω only when the fragment is within the specular flux triangle.

To construct the A, B and C vertices on the edge of Ω, the normalized direction vectors ($\vec{l_a} = \frac{A_F - F}{\|A_F - F\|}$, $\vec{l_b} = \frac{B_F - F}{\|B_F - F\|}$ and $\vec{l_c} = \frac{C_F - F}{\|C_F - F\|}$) from the fragment position to each of the vertices in the flux plane are first calculated. The vertex A is finally expressed as $A = \phi_a \cdot \vec{l_a}$. Similarly vertex B may be expressed as $B = \phi_b \cdot \vec{l_b}$ and vertex C as $C = \phi_c \cdot \vec{l_c}$. Triangle ABC accordingly 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.

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 and must be approximated. There are two conditions on the volume calculation to prevent visual anomalies, which are as follows:

- for a single triangle scatterer there should be no discontinuities between the volumes calculated for fragment positions within and outside of the flux triangle, and
- for a plate/mesh of polygon scatterers the probability distribution must be approximated as close as possible.

Note that the last condition holds regardless of the distribution, but a Gaussian distribution approximates the specular microfacet reality.

We approximate the 2D Gaussian probability distribution with a 3D lookup table that contains the pre-calculated probability ρ_i [from Equation (4)] over a triangle AOB such as shown in Figure 8. Therefore, to calculate P_E, the probability (volume under the distribution) of each of the domain triangles AOC, BOC and COA is looked up and summed. This summation is done regardless of whether or not the fragment is inside or outside of the specular flux triangle ABC. The only point to take note of is that the lookup table result is multiplied by the sign of the triangle area (e.g. sign of $\phi_a \vec{l_a} \times \phi_b \vec{l_b}$). A typical table of $128 \times 128 \times 128$ float entries has a size of 8 MB and may be used as shown in Figure 9 to calculate the fragment irradiance. The normal distribution is scene independent and currently computed offline using rejection sampling.

1. The volume P_E is calculated as the sum of volumes over $\overline{AOB}$, $\overline{BOC}$ and $\overline{COA}$ i.e.
   $P_E = GaussVolume(\phi_a \cdot \vec{l_a}, \phi_b \cdot \vec{l_b}, \sigma) + GaussVolume(\phi_b \cdot \vec{l_b}, \phi_c \cdot \vec{l_c}, \sigma) + GaussVolume(\phi_c \cdot \vec{l_c}, \phi_a \cdot \vec{l_a}, \sigma)$
2. The $GaussVolume(\vec{A}, \vec{B}, \sigma)$ function is defined as:
   $GaussVolume(\vec{A}, \vec{B}, \sigma) = \frac{\vec{A} \times \vec{B}}{\|\vec{A} \times \vec{B}\|}\ gaussTable(iA, iB, i\theta)$,
   and the normalised table indices as:
   $iA = min(\frac{\|\vec{A}\|}{\sigma^*}, 1.0)$
   $iB = min(\frac{\|\vec{B}\|}{\sigma^*}, 1.0)$, $\sigma^* = \sigma_{TableMax} \cdot \sigma$
   $i\theta = min(\frac{\theta}{\Pi}, 1.0)$, $\cos\theta = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \|\vec{B}\|}$.
   The `min` function ensures that the indices are clamped to [0,1]. The gaussTable stores the probability integral result over a triangle AOB such as shown in Figure 9.

Figure 9: *Calculation of the volume P_E from a table lookup.*

We found that quantization errors in the table indices for small triangular domains propagated to the image as noise. We reduced this noise by forcing probability values smaller than 0.01 (i.e. 1%) to zero.

5. Results and Analysis

We compare the performance of glossy backward beam tracing to that of a similar implementation of specular backward beam tracing. The quality and performance of glossy backward beam tracing is also compared to that of photon mapping as traditional specular beam tracing cannot render LGDE transport paths.

The point light source and Gaussian microfacet material models that we implemented in the backward beam tracer are used without modification in the photon mapping implementation. For comparison to photon mapping the standard deviation of the scatter distribution of the glossy beam must be related to the standard deviation of the microfacets of the material model. Using the law of reflection, the standard deviation of the microfacets is taken to be approximately half that of the scatter distribution.

We implemented a Whitted [Whi80] forward ray tracer that uses either the set of scattered glossy beams or the caustic photon map when shading a fragment. All performance

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

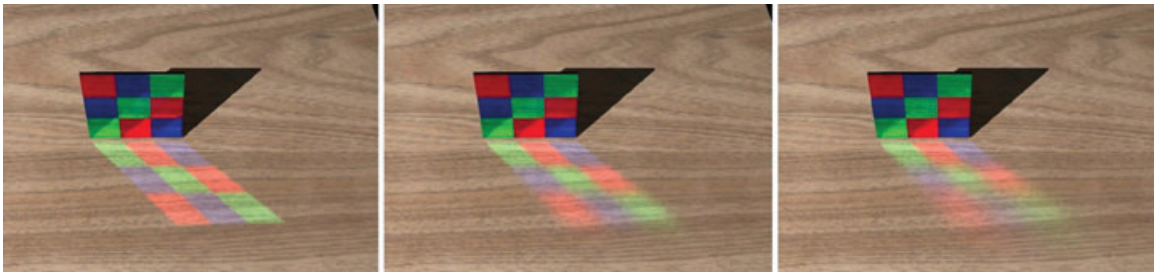

Figure 10: *Glossy backward beam tracing with a 0%, 10% and 30% 4 × microfacet scatter distribution on the surface of the tricolour reflector.*

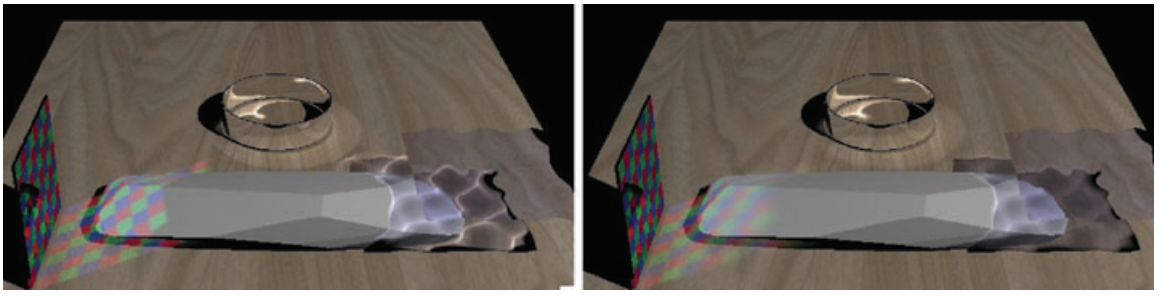

Figure 11: *Glossy backward beam tracing of a more complex scene with a 0% and 20% 4 × microfacet scatter distribution in the left and right images, respectively.*

measurements are done on a Macbook Pro with a 2.26 GHz Intel Core 2 Duo CPU running OSX 10.6.6. OpenMP is used to accelerate the ray tracing over the available two CPU cores and all frames are rendered at a resolution of 640 × 480 with one sample per pixel. The scatter distribution's standard deviation σ is specified as a function of a percentage P viz. $4\sigma = \frac{P}{100}\frac{\Pi}{2}$ and we refer to a 4 × microfacet scatter distribution of P%.

5.1. Results

Figure 10 shows glossy backward polygon beam tracing running at 6.5 frames/s. Note that the microfacet variance does not impact the execution performance, but the larger cone bounding volumes reduce the beam tree-traversal performance as will be shown for the scenes in Figures 11 and 12. The added cost of the glossy backward beam tracer over that of the specular beam tracer (not shown, but executes at 6.8 frames per second) is the cost of the probability distribution table lookups. Three lookups are required for a beam scattered from a triangle primitive, four lookups if scattered from a quad primitive, etc.

Figure 11 shows a more complex scene including a diffuse random polytope, a caustic ring, a coloured wall and a refractive water surface with specular (left) and glossy (right) materials. These images were rendered in 4.5 and 7.4 s for the 0% and 10% scatter distributions, respectively. Note the change in the execution performance of the beam processing due to the widening of the glossy beams.

Figure 12 shows a scene with two diffuse Stanford Bunnies, a refractive water surface, a coloured wall and a partial caustic ring. These images were rendered in 5.3, 7.9 and 12 s for the 0%, 10% and 20% scatter distributions, respectively. Note the change in the execution performance of the beam processing due to the widening of the glossy beams. The bottom right pane in the figure shows a wireframe view of the scene. The Stanford Bunny is the most complex with 65k polygons, the caustic ring has 210 polygons on the inside and the water surface has 4096 polygons. The rendering time for Figure 12 is similar to that of Figure 11 due to the hierarchical acceleration structures used for beam tree-traversal and scene tree-traversal.

In Figure 13, backward beam tracing (left) is compared to a reference photon map implementation (right) with a microfacet standard deviation of 0%. The difference image (bottom, centre) is shown at the same brightness scale. Note that only the $L(S\,|\,G)DE$ transport paths are rendered. The ring object (Figure 1) is constructed out of 75 × 5 scattering surfaces on the inside and also on the outside and renders in 1.5 s using backward beam tracing and approximately 60 s

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

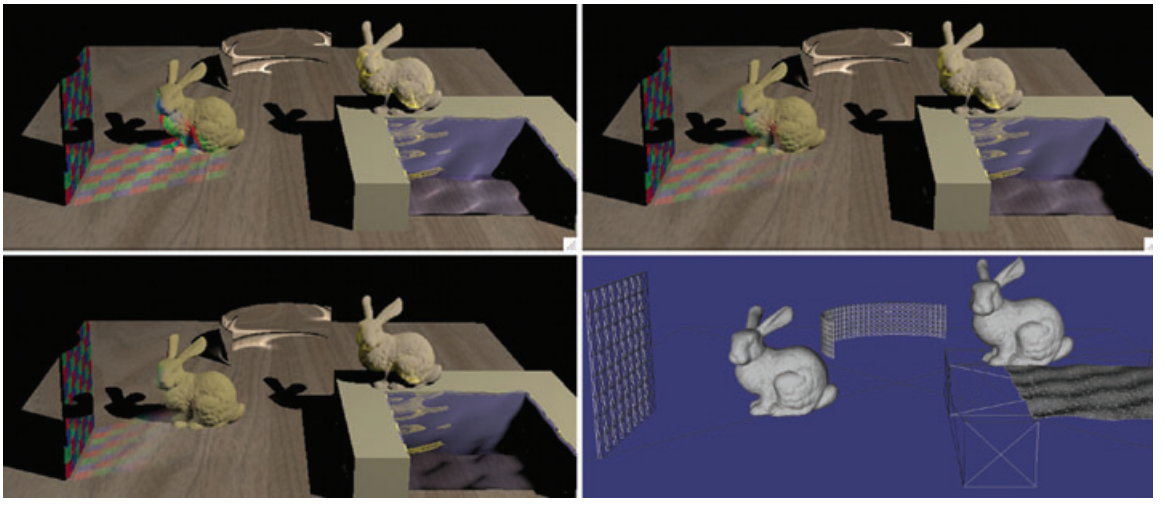

Figure 12: *Glossy backward beam tracing of a more complex scene with a 0%, 10% and 20% 4 × microfacet scatter distribution in the left, right and bottom-right images, respectively. The bottom-right pane shows a wireframe view of the scene.*

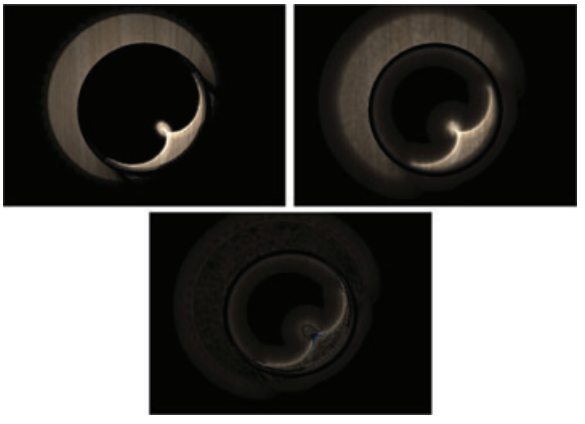

Figure 13: *Backward beam tracing (left) compared to photon mapping (right) of a ring object (Figure 1) with a scatter distribution variance of 0; only the L(S | G)DE transport paths are rendered.*

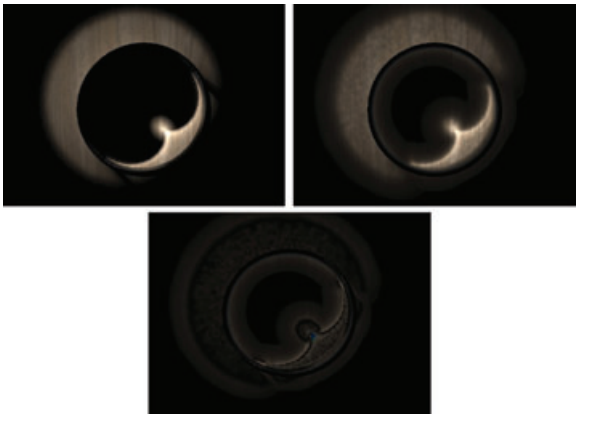

Figure 14: *Backward beam tracing (left) compared to photon mapping (right) of a ring object (Figure 1) with a 4 × scatter distribution standard deviation of 10%; only the L(S | G)DE transport paths are rendered.*

using photon mapping. The forward ray tracer is capable of tracing 600k rays per second on this scene and rendering the ring object without the $L(S \mid G)DE$ paths takes only 0.5 s.

The reference images are rendered with 420k photons radiated in the direction of the ring and 100k scattered photons absorbed into the caustic photon map. From the view point shown most of the render time is spent doing the k-Nearest-Neighbour (kNN) queries because even though the photon tracer is capable of tracing 600k rays per second our implementation of the kNN query can do only 8k queries per second for $k = 100$. The caustic photon map is rendered directly and a cone filter is applied in the radiance estimate during forward rendering.

Figure 14 shows the comparative results for a 4 × scatter distribution standard deviation of 10.0% rendered in 2.5 s and 60 s for the beam tracing and photon mapping respectively. Again there is a fairly good match between the glossy beam tracing and photon mapping results. Figure 15 shows the comparative results for a 4 × scatter distribution standard deviation of 30.0% rendered in 5.0 s and 60 s for the beam tracing and photon mapping respectively. Again,

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

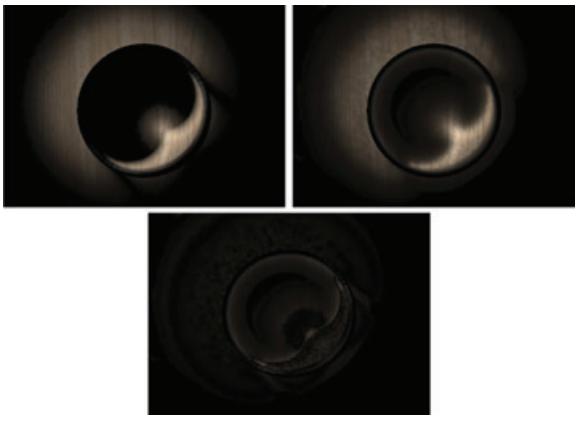

Figure 15: *Backward beam tracing (left) compared to photon mapping (right) of a ring object (Figure 1) with a 4 × scatter distribution standard deviation of 30%; only the L(S | G)DE transport paths are rendered.*

note the change in the execution performance of the beam tree-traversal due to the widening of the glossy beams.

5.2. Analysis and limitations

The difference images show that there is a good match between the overall shape and radiance of backward beam tracing and photon mapping. Using the photon map takes approximately 60 s to render the ring scene whereas the beam tracer renders it in 1.5–5 s depending on beam width. This is of course because of the forward renderer having to process a hierarchy of only 375 beams to light a fragment instead of doing a radiance estimate (kNN query) from a kd-tree with 100k photons.

The added cost of the glossy backward beam tracer over that of the specular beam tracer is only the overhead of the probability distribution table lookups. In this implementation, the overhead amounts to about a 5% performance difference between glossy and specular backward beam tracing.

The following approximations are included to allow a simpler model and an efficient implementation:

- The irradiance is assumed constant over the flux polygon and the average specular scatter direction is used for lighting the fragment.
- The scatter distribution is currently assumed radially symmetric and to have a constant variance over the polygon.
- The integration domain Ω is approximated by a triangular domain ABC.

Removing or improving any of the approximations would improve the fidelity of the glossy backward polygon beam model to better approximate reality. Glossy backward beam tracing is nonetheless able to render single scatter caustics of a quality approaching that of a photon map while using a relatively small number of scattered glossy beams.

6. Conclusion

A large number of photons or rays are typically required to render LGDE transport paths. It is consequently difficult to render these transport paths efficiently.

To address this problem of efficiency, we have expressed the lighting integral as the volume under a well-defined 2D probability distribution. The generalized backward polygon beam tracing technique now provides a lumped model of L(S | G)DE transport paths, which results in a simulation that is an order of a magnitude faster than photon mapping for rendering specular and glossy caustics. We have therefore shown that the efficiency of backward beam tracing in gathering path coherency can be exploited for non-specular surfaces. The added fragment shading overhead of this generality is a probability distribution table lookup per beam vertex.

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

- As with other finite element methods, including high-frequency surface detail requires a surface subdivision step to set up the beams.
- The approximations listed at the end of Section 5.2 that limit the model fidelity.

As far as future work is concerned, Cook and Torrance [CT82] have already considered the benefits of using other microfacet distributions for computer graphics. A more accurate microfacet distribution could also be applied to the glossy beam transport model and one might even be able to generalize the distribution and model to include the diffuse component of the surface BRDF. The light field is also only implicitly specified by the glossy beams. The glossy beams do therefore not encode enough information to deduce the light field's angular distribution. This information is required to model multiscatter glossy caustics and area light sources.

OpenMP is currently used to generate the scattered beams concurrently on the multiple processor cores of the host CPU. One could consider the use of CUDA and OpenCL to accelerate the beam construction on GPUs when large scenes are rendered, but for the scenes shown the beam construction takes only about 1% of the total time it takes to render a frame. An approach similar to the light image by Briere and Poulin [BP00] could also be investigated further to efficiently set up and manage the backward beams.

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

We would certainly like to port the raytrace forward renderer to NVIDIA's OptiX ray engine to access GPU acceleration while maintaining the accuracy of ray tracing. Similar to the CPU implementation, when a caustic receiver is intersected the OptiX ray shader would process the glossy beam set and execute the pseudo code shown in Figure 9.

References

[Arv86] Arvo J.: Backward ray tracing. In *Developments in Ray Tracing, SIGGRAPH Course Notes* (vol. 12) (New York, NY, USA, 1986), ACM Press.

[BP00] Brière N., Poulin P.: Adaptive representation of specular light flux. In *Graphics Interface*. Sidney Fels and Pierre Poulin (Eds.). Montreal, Canada (May 2000), Canadian Human Computer Communications Society, pp. 127–136.

[CC95] Chuang J.-H., Cheng S.-A.: Computing caustic effects by backward beam tracing. *The Visual Computer 11*, *3* (1995), 156–166.

[CT82] Cook R. L., Torrance K. E.: A reflectance model for computer graphics. *ACM Transactions on Graphics 1*, *1* (1982), 7–24.

[DBK10] Duvenhage B., Bouatouch K., Kourie D.: Exploring the use of glossy light volumes for interactive global illumination. In *AFRIGRAPH '10: Proceedings of the 7th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa* (New York, NY, USA, 2010), ACM Press, New York, pp. 139–148.

[EAMJ05] Ernst M., Akenine-Möller T., Jensen H. W.: Interactive rendering of caustics using interpolated warped volumes. In *GI '05: Proceedings of Graphics Interface 2005* (School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2005), Canadian Human-Computer Communications Society, pp. 87–96.

[Hec90] Heckbert P.: Adaptive radiosity textures for bidirectional ray tracing. In *SIGGRAPH '90: Proceedings of the 17th Annual Conference on Computer graphics and Interactive Techniques* (New York, NY, USA, 1990), ACM Press, New York, pp. 145–154.

[HH84] Heckbert P. S., Hanrahan P.: Beam tracing polygonal objects. In *SIGGRAPH '84: Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1984), ACM Press, New York, pp. 119–127.

[HOJ08] Hachisuka T., Ogaki S., Jensen H. W.: Progressive photon mapping. In *SIGGRAPH Asia '08: ACM SIGGRAPH Asia 2008 Papers* (New York, NY, USA, 2008), ACM Press, New York, pp. 1–8.

[IDN02] Iwasaki K., Dobashi Y., Nishita T.: An efficient method for rendering underwater optical effects using graphics hardware. *Computer Graphics Forum 21*, *4* (2002), 701–711.

[IDN03] Iwasaki K., Dobashi Y., Nishita T.: A fast rendering method for refractive and reflective caustics due to water surfaces. *Computer Graphics Forum 23*, *3* (2003), 601–609.

[IZT*07] Ihrke I., Ziegler G., Tevs A., Theobalt C., Magnor M., Seidel H.-P.: Eikonal rendering: efficient light transport in refractive objects. In *SIGGRAPH '07: ACM SIGGRAPH 2007 Papers* (New York, NY, USA, 2007), ACM Press, New York, p. 59.

[ML09] McGuire M., Luebke D.: Hardware-accelerated global illumination by image space photon mapping. In *HPG '09: Proceedings of the Conference on High Performance Graphics 2009* (New York, NY, USA, 2009), ACM Press, New York, pp. 77–89.

[NMN87] Nishita T., Miyawaki Y., Nakamae E.: A shading model for atmospheric scattering considering luminous intensity distribution of light sources. *SIGGRAPH Computer Graphics 21*, *4* (1987), 303–310.

[NN94] Nishita T., Nakamae E.: 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* (New York, NY, USA, 1994), ACM Press, New York, pp. 373–379.

[SJ09] Spencer B., Jones M. W.: Into the blue: better caustics through photon relaxation. In *Eurographics 2009: Proceedings of the 30th Annual Conference of the European Association for Computer Graphics* (Munich, Germany, 2009).

[SKP07] Shah M. A., Konttinen J., Pattanaik S.: Caustics mapping: an image-space technique for real-time caustics. *IEEE Transactions on Visualization and Computer Graphics 13*, *2* (2007), 272–280.

[SZS*08] Sun X., Zhou K., Stollnitz E., Shi J., Guo B.: Interactive relighting of dynamic refractive objects. In *SIGGRAPH '08: ACM SIGGRAPH 2008 Papers* (New York, NY, USA, 2008), ACM Press, New York, pp. 1–9.

[TS67] Torrance K. E., Sparrow E. M.: Theory for off-specular reflection from roughened surfaces. *Journal of the Optical Society of America 57*, *9* (1967), 1105–1114.

[UPSK08] Umenhoffer T., Patow G., Szirmay-Kalos L.: Caustic triangles on the gpu. In *CGI 2008: Proceedings*

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.

of the 26th Computer Graphics International Conference (Istanbul, Turkey, 2008).

[Wat90] WATT M.: Light-water interaction using backward beam tracing. In *SIGGRAPH '90: Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1990), ACM Press, New York, pp. 377–385.

[Whi80] WHITTED T.: An improved illumination model for shaded display. *Communications of the ACM 23* (June 1980), 343–349.

[Wym08] WYMAN C.: Hierarchical caustic maps. In *I3D '08: Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2008), ACM Press, New York, pp. 163–171.

[ZHWG08] ZHOU K., HOU Q., WANG R., GUO B.: Real-time kd-tree construction on graphics hardware. *ACM Transactions on Graphics 27*, *5* (2008), 1–11.

© 2011 The Authors
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd.