*From the ACM SIGGRAPH Asia 2011 conference proceedings.*

# Progressive Photon Beams

Wojciech Jarosz[1] Derek Nowrouzezahrai[1] Robert Thomas[1] Peter-Pike Sloan[2] Matthias Zwicker[3]

[1]Disney Research Zürich [2]Disney Interactive Studios [3]University of Bern

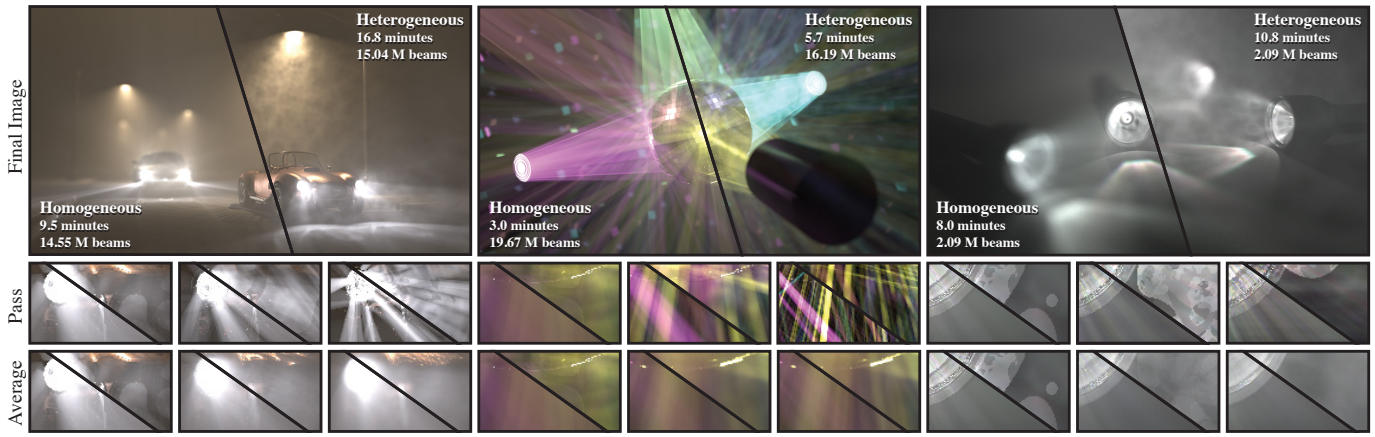


**Figure 1:** *The* CARS, DISCO, *and* FLASHLIGHTS *scenes rendered using progressive photon beams for both homogeneous and heterogeneous media. We generate a sequence of independent render passes (middle) where we progressively reduce the photon beam radii. This slightly increases variance in each pass, but we prove that the average of all passes (bottom) converges to the ground truth.*

## Abstract

We present progressive photon beams, a new algorithm for rendering complex lighting in participating media. Our technique is efficient, robust to complex light paths, and handles heterogeneous media and anisotropic scattering while provably converging to the correct solution using a bounded memory footprint. We achieve this by extending the recent photon beams variant of volumetric photon mapping. We show how to formulate a progressive radiance estimate using photon beams, providing the convergence guarantees and bounded memory usage of progressive photon mapping. Progressive photon beams can robustly handle situations that are difficult for most other algorithms, such as scenes containing participating media and specular interfaces, with realistic light sources completely enclosed by refractive and reflective materials. Our technique handles heterogeneous media and also trivially supports stochastic effects such as depth-of-field and glossy materials. Finally, we show how progressive photon beams can be implemented efficiently on the GPU as a splatting operation, making it applicable to interactive and real-time applications. These features make our technique scalable, providing the same physically-based algorithm for interactive feedback and reference-quality, unbiased solutions.

**CR Categories:** I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Raytracing

**Keywords:** participating media, global illumination, photon mapping, photon beams, density estimation

**Links:** DL PDF WEB VIDEO



## 1 Introduction

The scattering of light creates stunning visual complexity, in particular with interactions in participating media such as clouds, fog, and even air. Rendering this complex light transport requires solving the radiative transport equation [Chandrasekar 1960] combined with the rendering equation [Kajiya 1986] as a boundary condition.

The gold standard for rendering is arguably computing unbiased, noise-free images. Unfortunately, the only options to achieve this are variants of brute force path tracing [Kajiya 1986; Lafortune and Willems 1993; Veach and Guibas 1994; Lafortune and Willems 1996] and Metropolis light transport [Veach and Guibas 1997; Pauly et al. 2000], which are slow to converge to noise-free images despite recent advances [Raab et al. 2008; Yue et al. 2010]. This becomes particularly problematic when the scene contains so-called SDS (specular-diffuse-specular) or SMS (specular-media-specular) subpaths, which are actually quite common in physical scenes (e.g. illumination due to a light source inside a glass fixture). Unfortunately, path tracing methods cannot robustly handle these situations, especially in the presence of small light sources. Methods based on volumetric photon mapping [Jensen and Christensen 1998; Jarosz et al. 2008] do not suffer from these problems. They can robustly handle S(D|M)S subpaths, and generally produce less noise. However, these methods suffer from bias, which can be eliminated *in theory* by using infinitely many photons, but *in practice* this is not feasible since it requires unlimited memory.

We combine the benefits of these two classes of algorithms. Our approach converges to the gold standard much faster than path tracing, is robust to S(D|M)S subpaths, and has a bounded memory footprint. In addition, we show how the algorithm can be accelerated on the GPU. This allows for interactive lighting design in the presence of complex light sources and participating media. We also obtain reference-quality results at real-time rates for simple scenes containing complex volumetric light interactions. Our algorithm gracefully handles a wide spectrum of fidelity settings, ranging from real-time frame rates to reference quality solutions. This makes it possible to produce interactive previews with the same technique used for a high-quality final render — providing visual consistency, an essential property for interactive lighting design tools.

Our approach draws upon two recent advances in rendering: pho-

ton beams and progressive photon mapping. The photon beams method [Jarosz et al. 2011] is a generalization of volumetric photon mapping which accelerates participating media rendering by performing density estimation on the full paths of photons (beams), instead of just photon scattering locations. Photon beams are blurred with a finite width, leading to bias. Reducing this width reduces bias, but unfortunately increases noise. Progressive photon mapping (PPM) [Hachisuka et al. 2008; Knaus and Zwicker 2011] provides a way to eliminate bias and noise simultaneously in photon mapping.

Unfortunately, naïvely applying PPM to photon beams is not possible due to the fundamental differences between density estimation using points and beams, so convergence guarantees need to be rederived for this more complicated case. Moreover, previous PPM derivations only apply to fixed-radius or k-nearest neighbor density estimation, which are commonly used for surface illumination. Photon beams, on the other hand, are formulated using variable kernel density estimation [Silverman 1986] where each beam has an associated kernel.

We show how to overcome the challenges of combining PPM with photon beams. The resulting progressive photon beams (PPB) algorithm is efficient, robust to S(D|M)S subpaths, and converges to ground truth with bounded memory usage. Additionally, we demonstrate how photons beams can be applied to efficiently handle heterogeneous media.

At a high-level our approach proceeds similarly as previous PPM techniques. The main idea, as illustrated in Figure 1, is to average over a sequence of rendering passes, where each pass uses an independent photon (beam) map. In our approach, we render each pass using a collection of stochastically generated photon beams. As a key step, we reduce the radii of the photon beams using a global scaling factor after each pass. Therefore each subsequent image has less bias but slightly more noise. As we add more passes, however, the average of these images converges to the correct solution. A main contribution of this paper is to derive an appropriate sequence of radius reduction factors for photon beams, and to prove convergence of the progressive approach. In summary:

- We perform a theoretical error analysis of density estimation using photon beams to derive the necessary conditions for convergence, and we provide a numerical validation of our theory. Our analysis generalizes previous approaches by allowing photon beams as the data representation, and by considering varying kernels centered at photons and not at query locations.
- We use our analysis to derive a progressive form of the photon beams algorithm that converges to an unbiased solution using finite memory.
- We introduce a progressive and unbiased generalization of deep shadow maps to handle heterogeneous media efficiently.
- We reformulate the photon beam radiance estimate as a splatting operation to exploit GPU rasterization: increasing performance for common light paths, and allowing us to render simple scenes with multiple specular reflections in real-time.

## 2 Related Work

**Photon Mapping.** Volumetric photon mapping was first introduced by Jensen and Christensen [1998] and subsequently improved by Jarosz et al. [2008] to avoid costly and redundant density queries due to ray marching. They formulated a "beam radiance estimate" that considered all photons along the length of a ray in one query. Jarosz et al. [2011] showed how to apply the beam concept not just to the query operation but also to the photon data representation. They utilized the entire photon path instead of just photon points to obtain a significant quality and performance improvement, and also suggested (but did not demonstrate) a biased way to handle heterogeneous media by storing an approximation of transmittance along each path, in the spirit of deep shadow maps [Lokovic and Veach 2000]. The photon beams method is conceptually similar to ray maps for surface illumination [Lastra et al. 2002; Havran et al. 2005; Herzog et al. 2007] as well as the recent line-space gathering technique [Sun et al. 2010]. All of these methods are biased, which allows for more efficient simulation; however, when the majority of the illumination is due to caustics (which is often the case with realistic lighting fixtures or when there are specular surfaces) the photons are visualized directly and many are required to obtain high-quality results. Though these methods converge to an exact solution as the number of photons increases, obtaining a converged solution requires *storing* an infinite collection of photons, which is not feasible.

**Progressive Photon Mapping.** Progressive photon mapping [Hachisuka et al. 2008] alleviates this memory constraint. Instead of storing all photons needed to obtain a converged result, it updates progressive estimates of radiance at measurement points in the scene [Hachisuka et al. 2008; Hachisuka and Jensen 2009] or on the image plane [Knaus and Zwicker 2011]. Photons are traced and discarded progressively, and the radiance estimates are updated after each photon tracing pass in such a way that the approximation converges to the correct solution in the limit. Previous progressive techniques have primarily focused on surface illumination, with the exception of Knaus and Zwicker [2011], who also demonstrated results for traditional volumetric photon mapping [Jensen and Christensen 1998]. Unfortunately, volumetric photon mapping with photon points produces inferior results to photon beams [Jarosz et al. 2011], especially in the presence of complex specular interactions that benefit most from progressive estimation. We specifically tackle this problem and extend progressive photon mapping theory to photon beams.

**Real-time Techniques.** To improve the efficiency of our approach, we progressively splat photon beams whenever possible using GPU rasterization. Though this significantly improves performance for the general case, restricted settings like homogeneous single-scattering with shadows can be handled more efficiently by specialized techniques [Engelhardt and Dachsbacher 2010; Chen et al. 2011]. There are also a number of techniques that have considered using line-like primitives for real-time rendering of volume caustics, but these techniques generally have different goals than ours. The method of Krüger et al. [2006] uses splatting, but does not produce physically-based results. Hu et al. [2010] proposed a physically-based approach, but it requires ray marching even for homogeneous media. Liktor and Dachsbacher [2011] improved upon this technique by using adaptive and approximate beam-tracing and splatting. These methods could handle only one or two specular surface interactions. For simple scenes with similar lighting restrictions we are also able to obtain real-time performance, but additionally provide a progressive and convergent solution. The recent technique by Engelhardt et al. [2010] targets approximate multiple scattering in heterogeneous media and can obtain real-time performance. Unlike our approach, all of these methods are either approximate, do not handle S(D|M)S subpaths, or will not converge to ground truth in bounded memory.

## 3 Preliminaries

We briefly describe the technical details of light transport in participating media and summarize relevant equations related to photon beams and progressive photon mapping which we build upon.

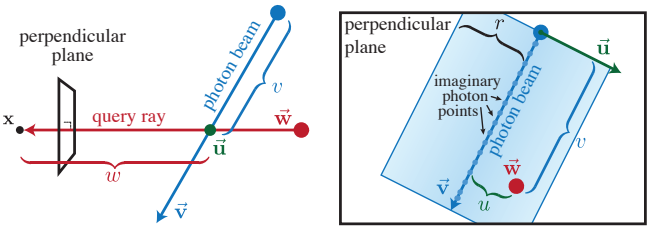


**Figure 2:** *Radiance estimation with one photon beam as viewed from the side (left) and in the plane perpendicular to the query ray where the direction* $\vec{\mathbf{u}} = \vec{\mathbf{v}} \times \vec{\mathbf{w}}$ *extends out of the page (right).*

### 3.1 Light Transport in Participating Media

In the presence of participating media, the light incident at any point in the scene $\mathbf{x}$ (e.g. the camera) from a direction $\vec{\mathbf{w}}$ (e.g. the direction through a pixel) can be expressed using the radiative transport equation [Chandrasekar 1960] as the sum of two terms:

$$L(\mathbf{x}, \vec{\mathbf{w}}) = T_r(s) L_s(\mathbf{x}_s, \vec{\mathbf{w}}) + L_m(\mathbf{x}, \vec{\mathbf{w}}). \quad (1)$$

The first term is outgoing reflected radiance from a surface, $L_s$, at the endpoint of the ray, $\mathbf{x}_s = \mathbf{x} - s\vec{\mathbf{w}}$, after attenuation due to the transmittance $T_r$. In homogeneous media, $T_r(s) = e^{-s\sigma_t}$, where $\sigma_t$ is the extinction coefficient. In heterogeneous media, transmittance accounts for the extinction coefficient along the entire segment between the two points, but we use this simple one-parameter notation here for brevity. The second term is medium radiance,

$$L_m(\mathbf{x}, \vec{\mathbf{w}}) = \int_0^s \sigma_s(\mathbf{x}_w)\, T_r(w) \int_{\Omega_{4\pi}} f(\vec{\mathbf{w}} \cdot \vec{\mathbf{v}})\, L(\mathbf{x}_w, \vec{\mathbf{v}})\, d\vec{\mathbf{v}}\, dw, \quad (2)$$

where $f$ is the normalized phase function, $\sigma_s$ is the scattering coefficient, and $w$ is a scalar distance along the camera direction $\vec{\mathbf{w}}$. Equation (2) integrates scattered light at all points along $\mathbf{x}_w = \mathbf{x} - w\vec{\mathbf{w}}$ until the nearest surface at distance $s$. The inner integral computes in-scattered radiance which recursively depends on radiance arriving at $\mathbf{x}_w$ from directions $\vec{\mathbf{v}}$ on the sphere $\Omega_{4\pi}$.

### 3.2 Photon Beams

Photon mapping methods approximate the medium radiance (2) using a collection of photons, each with a power, position, and direction. Instead of performing density estimation on just the positions of the photons, the recent photon beams approach [Jarosz et al. 2011] treats each photon as a beam of light starting at the photon position and shooting in the photon's outgoing direction. Jarosz et al. derived a "Beam × Beam 1D" estimator which directly computes medium radiance (2) due to photon beams along a query ray.

We summarize the main ideas here and use a coordinate system $(\vec{\mathbf{u}}, \vec{\mathbf{v}}, \vec{\mathbf{w}})$ where $\vec{\mathbf{w}}$ is the query ray direction, $\vec{\mathbf{v}}$ is the direction of the photon beam, and $\vec{\mathbf{u}} = \vec{\mathbf{v}} \times \vec{\mathbf{w}}$ is the direction perpendicular to both the query ray and the photon beam (see Figure 2).

To estimate radiance due to a photon beam, we treat the beam as an infinite number of imaginary photon **points** along its length (Figure 2, right). The power of the photons is blurred in 1D, along $\vec{\mathbf{u}}$.

An estimate of the incident radiance along the direction $\vec{\mathbf{w}}$ using one photon beam can be expressed as [Jarosz et al. 2011]:

$$L_m(\mathbf{x}, \vec{\mathbf{w}}, r) \approx k_r(u)\, \sigma_s(\mathbf{x}_w)\, \Phi\, T_r(w)\, T_r(v)\, \frac{f(\vec{\mathbf{w}} \cdot \vec{\mathbf{v}})}{\sin(\vec{\mathbf{w}}, \vec{\mathbf{v}})}, \quad (3)$$

where $\Phi$ is the power of the photon, and the scalars $(u, v, w)$ are signed distances along the three axes to the imaginary photon point closest to the query ray (the point on the beam closest to the ray $\vec{\mathbf{w}}$). The first transmittance term accounts for attenuation through a distance $w$ to $\mathbf{x}$, and the second computes the transmittance through a distance $v$ to the start of the beam. The photon beam is blurred using a 1D kernel $k_r$ centered on the beam with a support width of $r$ along direction $\vec{\mathbf{u}}$. This is illustrated in Figure 2.

In practice, Equation (3) is evaluated for many beams to obtain a high quality image and is a consistent estimator like standard photon mapping. In other words, it produces an unbiased solution when using an infinite number of beams with an infinitesimally-small blur kernel. This is an important property which we will use later on.

However, obtaining an unbiased result requires *storing* an infinite number of beams, and using an infinitesimal blurring kernel; neither of which are feasible in practice. In Section 4.1 we analyze the variance and bias of Equation (3) in order to develop a progressive, memory-bounded consistent estimator in the spirit of PPM.

### 3.3 Progressive Photon Mapping

Though a full derivation is outside the scope of this paper, we provide an overview of PPM for completeness, summarizing the approach used by Knaus and Zwicker [2011]. Their method builds on a probabilistic analysis of the error in radiance estimation. They model the error as consisting of two components: its variance (noise) and its expected value (bias). The main idea in PPM is to average a sequence of images generated using independent photon maps. The key insight is that radiance estimation can be performed such that both the variance and the expected value of the *average error* are reduced continuously as more images are added. Therefore, PPM achieves noise- and bias-free results in the limit.

Denote the error of radiance estimation in pass $i$ at some point in the scene by $\epsilon_i$. The average error after $N$ passes is $\bar{\epsilon}_N = \frac{1}{N}\sum_{i=1}^{N} \epsilon_i$. Since each image $i$ uses a different photon map, the errors $\epsilon_i$ can be interpreted as samples of independent random variables. Hence, the variance (noise) and expected value (bias) of average error are

$$\mathrm{Var}[\bar{\epsilon}_N] = \frac{1}{N^2}\sum_{i=1}^{N} \mathrm{Var}[\epsilon_i] \quad \text{and} \quad \mathrm{E}[\bar{\epsilon}_N] = \frac{1}{N}\sum_{i=1}^{N} \mathrm{E}[\epsilon_i]. \quad (4)$$

Convergence in progressive photon mapping is obtained if the average noise and bias go to zero simultaneously as the number of passes goes to infinity, i.e., as $N \to \infty$,

$$\left.\begin{array}{ll} A) & \mathrm{Var}[\bar{\epsilon}_N] \to 0 \\ B) & \mathrm{E}[\bar{\epsilon}_N] \to 0 \end{array}\right\} \Longrightarrow \text{convergence.} \quad (5)$$

Observe from Equation (4) that if the same radii were used in the radiance estimate in each pass, the variance of the average error would be reduced, but the bias would remain the same. The main trick of PPM is to decrease the radiance estimation radii in each pass by a small factor. This increases variance in each pass, but only so much that the variance of the average still vanishes. At the same time, reducing the radii decreases the bias, hence achieving the desired convergence. We extend this theory in the next section, applying it to radiance estimation using photon beams.

## 4 Progressive Photon Beams

The goal of our rendering technique is to compute the contribution of media radiance $L_m$ to pixel values $c$. Figure 3 illustrates the problem schematically. Mathematically, we formulate this as:

$$c = \iint W(\mathbf{x}, \vec{\mathbf{w}})\, L_m(\mathbf{x}, \vec{\mathbf{w}})\, d\mathbf{x}\, d\vec{\mathbf{w}}, \quad (6)$$

where $W$ is a function that weights the contribution of $L_m$ to the pixel value (accounting for antialiasing, glossy reflection, depth-of-field, etc.). We compute $c$ by tracing a number of paths $N$ from the eye, evaluating $W$, and evaluating the media radiance $L_m$. This

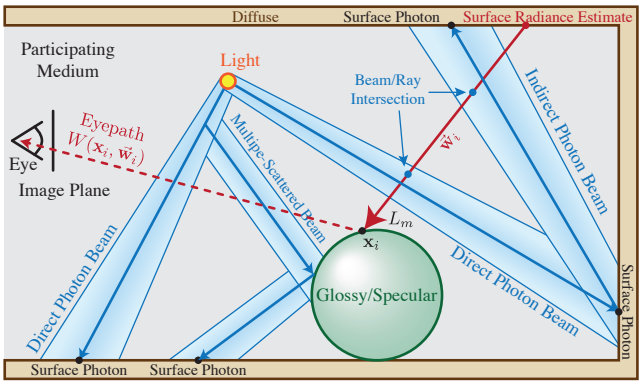


**Figure 3:** *We first shoot photon beams (blue) from the light sources (yellow). We then trace paths from the eye (red) until we hit a diffuse surface, estimating volumetric scattering by finding the beam/ray intersections and weighting by the contribution $W$ to the camera. In each pass we reduce a global radius scaling factor, and repeat.*

forms a Monte Carlo estimate $\bar{c}_N$ for $c$:

$$\bar{c}_N = \frac{1}{N}\sum_{i=1}^{N}\frac{W(\mathbf{x}_i,\vec{\mathbf{w}}_i)L_m(\mathbf{x}_i,\vec{\mathbf{w}}_i)}{p(\mathbf{x}_i,\vec{\mathbf{w}}_i)}, \tag{7}$$

where $p(\mathbf{x}_i,\vec{\mathbf{w}}_i)$ denotes the probability density of generating a particular position and direction when tracing paths from the eye and $L_m$ is approximated using photon beams.

The photon beam approximation of the media radiance introduces an error, which we can make explicit by redefining Equation (3) as:

$$L_m(\mathbf{x},\vec{\mathbf{w}},r) = k_r(u)\,\gamma + \epsilon(\mathbf{x},\vec{\mathbf{w}},r), \tag{8}$$

where all terms in Equation (3) except for the kernel have been folded into $\gamma$. The error term $\epsilon$, is the difference between the true radiance $L_m(\mathbf{x},\vec{\mathbf{w}})$ and the radiance estimated using a photon beam with a kernel of radius $r$. To prove convergence of this algorithm we need to perform an in-depth analysis of this error.

### 4.1 Error Analysis of Photon Beam Density Estimation

We first analyze the variance $\mathrm{Var}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)]$ of the error (noise), and expected value $\mathrm{E}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)]$ of the error (bias) for the case of one beam. We then generalize the result for the case of many beams. This is a key step that is necessary to derive a progressive estimate for photon beams, and allows us to assign different kernel radii to each beam. Our analysis builds on the technique of Knaus and Zwicker and extends it to work with photon beams. We provide in-depth derivations of certain steps in the appendices.

Since photon beams are generated using stochastic ray tracing, we can interpret the 1D distance $u$ between a photon beam and a query ray $\vec{\mathbf{w}}$ as independent and identically distributed samples of a random variable $U$ with probability density $p_U^{\vec{\mathbf{w}}}$. The photon contributions $\gamma$ can take on different values, which we again model as samples of a random variable. We assume that $u$ and $\gamma$ are independent. $\gamma$ incorporates several terms: $\sigma_s$, $\Phi$, and the transmittance along $w$ are all independent of $u$, and the remaining terms depend on $v$. Graphically, we assume that if we fix our query location and generate a random beam, the distances $u$ and $v$ (Figure 2, right) are mutually independent (note that these measure distance along orthogonal directions). This assumption need only hold locally since only beams close to a query ray contribute. Additionally, as the beam radii decrease, the accuracy of this assumption increases.

**Variance using One Photon Beam.** To derive the variance of the error we also assume that locally (within the 1D kernel's support at $\vec{\mathbf{w}}$), the distance $u$ between the beam and view ray is a uniformly distributed random variable. This is similar to the uniform local density assumption used in previous PPM methods [Hachisuka et al. 2008; Knaus and Zwicker 2011]. We show in Appendix A that under these assumptions, the variance can be expressed as:

$$\mathrm{Var}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)] = \frac{(\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2)\,p_U^{\vec{\mathbf{w}}}(0)\,C_1}{r} \tag{9}$$

where $C_1$ is a constant derived from the kernel, and $p_U^{\vec{\mathbf{w}}}(0)$ is the probability density of a photon beam intersecting the view ray $\vec{\mathbf{w}}$ exactly. This result states that the variance of beam radiance estimation *increases linearly* if we reduce the kernel radius $r$.

**Expected Error using One Photon Beam.** On the other hand, in Appendix B we show that, for some constant $C_2$, the expected error *decreases linearly* as we reduce the kernel support $r$:

$$\mathrm{E}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)] = r\,\mathrm{E}[\gamma]\,C_2. \tag{10}$$

**Using Many Beams.** In practice, the photon beam method generates images using more than one photon beam at a time. Moreover, the photon beam widths need not be equal, but could be determined adaptively per beam using e.g. photon differentials [Schjøth et al. 2007]. We can express this by generalizing Equation (8) as:

$$L_m(\mathbf{x},\vec{\mathbf{w}},r_1\ldots r_M) = \frac{1}{M}\sum_{j=1}^{M} k_{r_j}(u_j)\,\gamma_j - \epsilon(\mathbf{x},\vec{\mathbf{w}},r_1\ldots r_M).$$

In Appendix C we show that if we use $M$ beams, each with their own radius $r_j$ at $\vec{\mathbf{w}}$, the variance of the error is:

$$\mathrm{Var}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r_1\ldots r_M)] = \frac{1}{M}\frac{(\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2)\,p_U^{\vec{\mathbf{w}}}(0)\,C_1}{r_H}, \tag{11}$$

where $r_H$ is the harmonic mean of the radii, $1/r_H = \frac{1}{M}\sum\frac{1}{r_j}$. This includes the expected behavior that variance decreases linearly with the number of emitted photon beams $M$.

Furthermore, we show that the expected error is now (Appendix D):

$$\mathrm{E}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r_1\ldots r_M)] = r_A\,\mathrm{E}[\gamma]\,C_2, \tag{12}$$

where $r_A$ is the arithmetic mean of of the photon radii. Note that this does not depend on $M$, so the expected error (bias) will not decrease as we use more photon beams.

To be absolutely precise, this analysis requires that the minimum radius in any pass be non-zero (to avoid infinite variance) and the maximum radius be finite (to avoid infinite bias). In practice enforcing some bounds is not a problem and the high-level intuition remains the same: the variance *increases linearly* and expected error *decreases linearly* if we globally decrease all the beam radii as well as these radii bounds. It is also worth highlighting that for photon beams this relationship is *linear* due to the 1D blurring kernel. This is in contrast to the analysis performed by Knaus and Zwicker, which resulted in a *quadratic* relationship for the standard radiance estimate on surfaces and a *cubic* relationship for the volumetric radiance estimate using photon points.

**Summary.** Our analysis generalizes previous work in two important ways. Firstly, we consider the more complicated case of photon beams, and secondly, we allow the kernel radius to be associated with data elements (e.g. photons or photon beams) instead of the query location (as in k-nearest neighbor estimation). This second feature allows variable kernel density estimation in any PPM context, such as density estimation on surfaces using photons points. This analysis could also be applied in modified form to any of the other volumetric radiance estimates summarized by Jarosz et al. [2011] to form convergent algorithms, such as the beam radiance estimate [Jarosz et al. 2008] using photon points. The only effective change is that blur dimensionality (1D, 2D, or 3D) will dictate whether the variance and bias relationship is linear, quadratic or cubic.

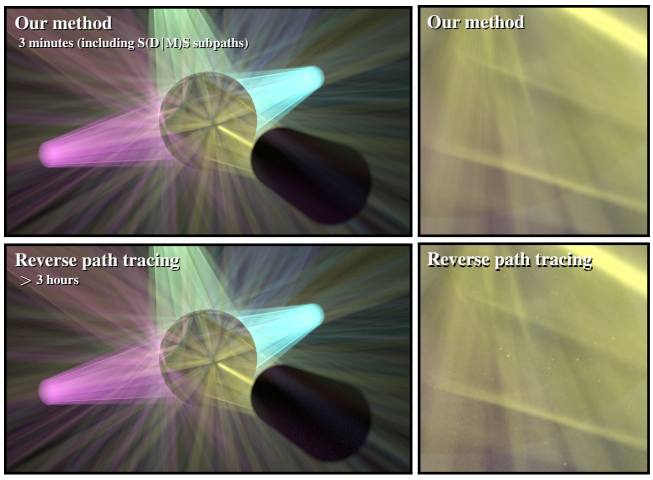


**Figure 4:** *Comparing the directly visible media radiance in the* DISCO *scene: our method (top) and reverse path tracing (bottom).*

### 4.2 Achieving Convergence

To obtain convergence using the approach outlined in Section 3.3 we show how to enforce conditions $A$ and $B$ from Equation (5). The key idea of PPM is to let variance increase slightly in each pass, but in such a way that the variance of the average error still vanishes. Increasing variance allows us to reduce the kernel scale (11), which in turn reduces the expected error of the radiance estimate (12).

**Variance of Average Error.** Knaus and Zwicker [2011] showed that convergence in PPM can be achieved by enforcing the following ratio of variance between passes:

$$\frac{\mathrm{Var}[\epsilon_{i+1}]}{\mathrm{Var}[\epsilon_i]} = \frac{i+1}{i+\alpha} \tag{13}$$

where $\alpha$ is a user specified constant between 0 and 1. Given the variance of the first pass, this ratio induces a variance sequence, where the variance of the $i^{\text{th}}$ pass is predicted as:

$$\mathrm{Var}[\epsilon_i] = \mathrm{Var}[\epsilon_1] \left( \prod_{k=1}^{i-1} \frac{k}{k+\alpha} \right) i. \tag{14}$$

Using this ratio, the variance of the average error after $N$ passes can be expressed in terms of the variance of the first pass $\mathrm{Var}[\epsilon_1]$:

$$\mathrm{Var}[\bar{\epsilon}_N] = \frac{\mathrm{Var}[\epsilon_1]}{N^2} \left( 1 + \sum_{i=2}^{N} \left( \prod_{k=1}^{i-1} \frac{k}{k+\alpha} \right) i \right), \tag{15}$$

which vanishes as desired when $N \to \infty$. Hence, if we could enforce such a variance sequence we would satisfy condition $A$.

**Radius Reduction Sequence.** In our approach, we use a global scaling factor $R_i$ to scale the radius of each beam, as well as the minimum and maximum radii bounds, in pass $i$. Note that scaling all radii by $R_i$ scales their harmonic and arithmetic means by $R_i$ as well. Our goal is to modify the radii such that the increase in variance between passes corresponds to Equation (13). We can enforce this by plugging in our expression for variance (11) into this ratio. Since variance is *inversely* proportional to beam radius, we obtain:

$$\frac{R_{i+1}}{R_i} = \frac{\mathrm{Var}[\epsilon_i]}{\mathrm{Var}[\epsilon_{i+1}]} = \frac{i+\alpha}{i+1} \tag{16}$$

Given an initial scaling factor of $R_1 = 1$, this ratio induces a sequence of scaling factors

$$R_i = \left( \prod_{k=1}^{i-1} \frac{k+\alpha}{k} \right) \frac{1}{i}, \tag{17}$$

**Expected Value of Average Error.** Since the expected error is proportional to the average radius (12), we can obtain a relation regarding the expected error of each pass from Equation (17):

$$\mathrm{E}[\epsilon_i] = \mathrm{E}[\epsilon_1] R_i, \tag{18}$$

where $\epsilon_1$ is the error of the first pass. We can solve for the expected value of the average error in a similar way to Equation (15):

$$\mathrm{E}[\bar{\epsilon}_i] = \frac{\mathrm{E}[\epsilon_1]}{N} \left( 1 + \sum_{i=2}^{N} \left( \prod_{k=1}^{i-1} \frac{k+\alpha}{k} \right) \frac{1}{i} \right). \tag{19}$$

Knaus and Zwicker [2011] showed that such an expression vanishes as $N \to \infty$. Hence, by using the radius sequence in Equation (17), we furthermore satisfy $B$.

### 4.3 Empirical Validation

We validate our algorithm against a reverse path tracing reference solution of the DISCO scene. This is an incredibly difficult scene for unbiased path sampling techniques. We compute many connections between each light path and the camera to improve convergence. Unfortunately, since the light sources in this scene are point lights, mirror reflections of the media constitute SMS subpaths which cannot be simulated. We therefore visualize only media radiance directly visible by the camera and compare to the media radiance using PPB. The reference solution in Figure 4 (bottom) took over 3 hours to render while our technique requires only 3 minutes (**including** S(D|M)S subpaths, as in Figure 1, which are not visualized here). We use $\alpha = 0.7$ and 19.67M beams in total.

We also numerically validate our error analysis by examining the noise and bias behavior (for three values of $\alpha$) of the highlighted point in the SPHERECAUSTIC scene in Figure 5. We use progressive photon beams to simulate multiple scattering and multiple specular bounces of light in the glass. We compute 10,000 independent runs of our simulation and compare these empirical statistics to the theoretical behavior predicted by our models.

Figure 6 (left) shows the sample variance of the radiance estimate as a function of the iterations. Since our theoretical error model depends on some scene-dependent constants which are not possible to estimate in the general case, we fit these constants to the empirical data. We gather statistics for $\alpha = 0.1$, $0.5$, and $0.9$, showing the effect of this parameter on the convergence. The top curves in Figure 6 (left) show the variance of each pass, $\mathrm{Var}[\epsilon_i]$, increases, as predicted by Equation (14). The lower curves in Figure 6 (left) show that variance of the average error, $\mathrm{Var}[\bar{\epsilon}_i]$, decreases after each pass as Equation (15) predicts. We also examine the expected average error (bias) in Figure 6 (right). This experiment show that both the bias and noise decay with the number of passes.

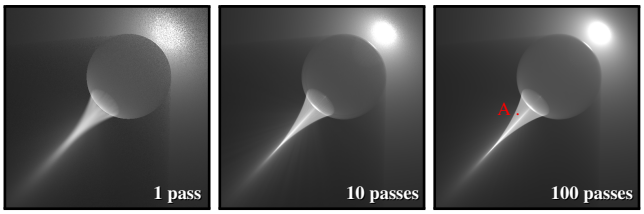


**Figure 5:** *The* SPHERECAUSTIC *scene contains a glass sphere, an anisotropic medium, and a point light. We shoot 1K beams per pass and obtain a high-quality result in 100 passes (right, ∼10 seconds). In Figure 6 we analyze bias and variance of the highlighted point.*

## 5 Algorithm

The inner loop of our algorithm uses the standard two-pass photon beams algorithm [Jarosz et al. 2011].

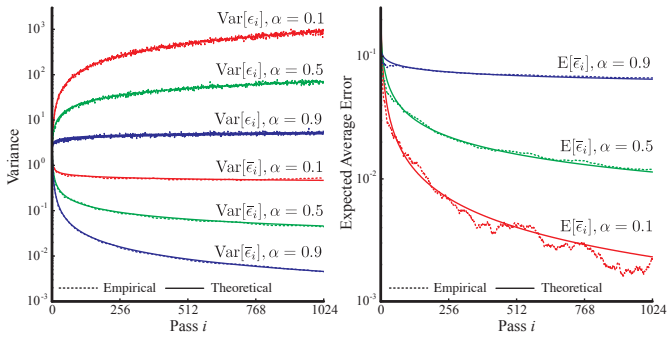


**Figure 6:** *Plots of per-pass variance $Var[\epsilon_i]$ and average variance $Var[\bar{\epsilon}_i]$ (left), and bias $E[\bar{\epsilon}_i]$ (right) with three $\alpha$ settings for the highlighted point in Figure 5. Empirical results closely match the theoretical models we derive in Section 4.1. The noise in the empirical curves is due to a limited number (10K) of measurement runs.*

In the first pass, photon beams are emitted from lights and scatter at surfaces and media in the scene. Other than computing appropriate beam widths, this pass is effectively identical to the photon tracing in volumetric and surface-based photon mapping [Jensen 2001]. We determine the kernel width of each beam by tracing photon differentials during the photon tracing process. We also automatically compute and enforce radii bounds to avoid infinite variance or bias.

In the second pass, we compute radiance along each ray using Equation (3). For homogeneous media, this involves a couple exponentials and scaling by the scattering coefficient and foreshortened phase function. We discuss heterogeneous media in Section 5.2.

In our progressive estimation framework, we repeat these two steps in each progressive pass, scaling the widths of all beams (and the radii bounds) by the global scaling factor.

### 5.1 User Parameters

Our goal is to provide the user with a single intuitive parameter to control convergence. In standard PPM a number of parameters influence the algorithm's performance: the bias-noise tradeoff $\alpha \in (0, 1)$, the number of photons per pass $M$, and either a number of nearest neighbors $k$ or an initial global radius.

Unfortunately, the parameters $\alpha$ and $M$ both influence the bias-noise tradeoff in an interconnected way. Since the radius is updated each pass, increasing $M$ means that the radius is scaled more slowly with respect to the total number of photons after $N$ passes. This is illustrated in differences between red and green curves in Figure 7.

Given two progressive runs, we would hope to obtain the same output image if the same total number of beams have been shot, regardless of the number $M$ per pass. We provide a simple improvement that achieves this intuitive behavior by reducing the effect of $M$ on the bias-noise tradeoff. Our solution is to always apply $M$ progressive radius updates after each pass. This induces a piecewise constant approximation of $M = 1$ for the radius reduction, at any arbitrary setting of $M$ (see the blue curves in Figure 7 and compare to the green curves). This modifies Equations (16) and (17) to:

$$\frac{R_{i+1}}{R_i} = \sum_{j=1}^{M} \frac{(i-1)M + j + \alpha}{(i-1)M + j + 1}, \quad R_i = \left( \prod_{k=1}^{Mi-1} \frac{k+\alpha}{k} \right) \frac{1}{Mi}.$$

It is clear that variance still vanishes in this modified scheme, since the beams in each pass have larger (or for the first pass equal) radii to the "single beam per pass" ($M = 1$) approach. The expected error vanishes because eventually the scaling factor is still zero.

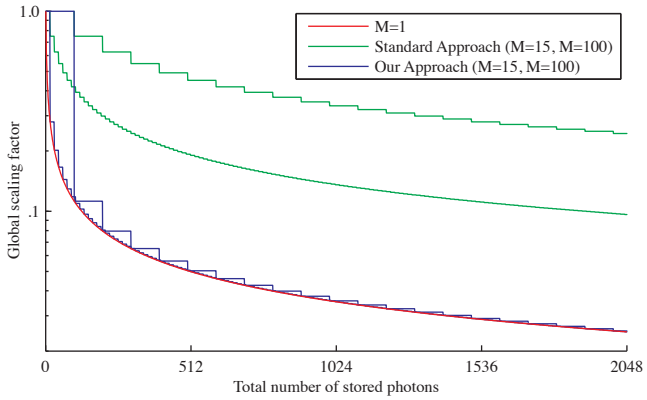


**Figure 7:** *Plotting the global radius scaling factor, with varying $M$, the standard approach produces vastly different scaling sequences for a progressive simulation using the same total number of stored photons. We reduce the scale factor $M$ times after each pass, which approximates the scaling sequence of $M = 1$ regardless of $M$.*

### 5.2 Evaluating $\gamma$ in Heterogeneous Media

In theory, our error analysis and convergence derivation in Section 4 apply to both homogeneous and heterogeneous media – the properties of the medium are encapsulated in the random variable $\gamma$. To realize a convergent algorithm for heterogeneous media, however, we need a way to evaluate the scattering properties and compute the transmittances contained in $\gamma$. To handle heterogeneous media, Jarosz et al. [2011] proposed storing a sampled representation of the transmittance function along each beam and numerically evaluating transmittance to the camera. Unfortunately, the standard approach for computing transmittance, ray marching, is biased, and would compromise our algorithm's convergence. To prevent this, our transmittance estimator must be unbiased.

We can form such an unbiased estimator by using mean-free path sampling as a black-box. Given a function, $\mathrm{d}(\mathbf{x}, \vec{\omega})$, which returns a random propagation distance from a point $\mathbf{x}$ in direction $\vec{\omega}$, the transmittance between $\mathbf{x}$ and a point $s$ units away in direction $\vec{\omega}$ is

$$T_r(\mathbf{x}, \vec{\omega}, s) = \mathrm{E}\left[ \frac{1}{n} \sum_{j=0}^{n} \mathrm{H}(\mathrm{d}(\mathbf{x}, \vec{\omega}) - s) \right], \qquad (20)$$

where H is the Heaviside step function. This estimates transmittance by counting samples that successfully propagate a distance $\geq s$.

This has immediate utility for rendering photon beams in heterogeneous media. A naïve solution could numerically compute the two transmittance terms in Equation (8) using Equation (20). This approach can be applied to both homogeneous media, where $\mathrm{d}(\mathbf{x}, \vec{\omega}) = -\log(1 - \xi)/\sigma_t$, and to heterogeneous media, where $\mathrm{d}(\mathbf{x}, \vec{\omega})$ can be implemented using Woodcock tracking [Woodcock et al. 1965]. Woodcock tracking is a technique developed in the particle physics field for unbiased mean-free path sampling. Several recent graphics techniques [Raab et al. 2008; Szirmay-Kalos et al. 2011] have employed this, in combination with Equation (20), for estimating transmittance in heterogeneous media.

Though this approach converges to the correct result, it is inefficient, since there are many ray/beam intersections to evaluate and each evaluation requires many samples to obtain low variance. Our solution builds upon this technique to more efficiently evaluate transmittance for all ray/beam intersections. We first consider transmittance towards the eye, and then transmittance along each photon beam.

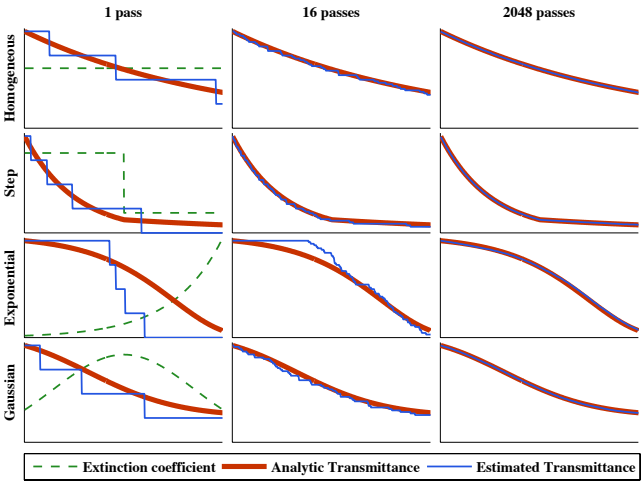


**Figure 8:** *Validation of progressive deep shadow maps (blue) for extinction functions (green) with analytically-computable transmittances (red). Here we use four random propagation distances, resulting in a four-step approximation of transmittance in each pass.*

#### 5.2.1 Progressive Deep Shadow Maps

The key to a more efficient approach is that each execution of Equation (20) actually provides enough information for an unbiased evaluation of the *transmittance function* for all distances along the ray, and not just the *transmittance value* at a single distance $s$. We achieve this by computing all $n$ propagation distances and re-evaluating Equation (20) for arbitrary values of $s$. This results in an *unbiased*, piecewise-constant representation of the transmittance function, as illustrated in Figure 8 (left). The collection of transmittance functions across all primary rays could be viewed as a deep shadow map [Lokovic and Veach 2000] from the camera. Deep shadow maps also store multiple distances to approximate transmittance; however, the key difference here is that our transmittance estimator remains unbiased, and will converge to the correct solution when averaged across passes. In Appendix E we prove this convergence and in Figure 8 we validate it empirically for heterogeneous media with closed-form solutions to transmittance.

We similarly accelerate transmittance along each beam. Instead of repeatedly evaluating Equation (20) at each ray/beam intersection, we compute and store several unbiased random propagation distances along each beam. Given these distances, we can re-evaluate transmittance using Equation (20) at any distance along the beam. The collection of transmittance functions across all photon beams forms an unstructured deep shadow map which converges to the correct result with many passes.

#### 5.2.2 Effect on Error Convergence

When using Equation (20) to estimate transmittance, the only effect on the error analysis is that $\mathrm{Var}[\gamma]$ in Equations (9) and (11) increases compared to using analytic transmittance (note that bias is not affected since $\mathrm{E}[\gamma]$ does not change with an unbiased estimator). Homogeneous media could be rendered using the analytic formula or using Equation (20). Both approaches converge to the same result (top row of Figure 8), but the Monte Carlo estimator for transmittance adds additional variance. We therefore use analytic transmittance in the case of homogeneous media.

## 6 Implementation and Results

We demonstrate the generality and flexibility of our approach with several efficient implementations of our theory. First, we introduce our most general implementation, which is a CPU-GPU hybrid capable of rendering arbitrary surface and volumetric shading effects, including complex paths with multiple reflective, refractive and volumetric interactions in homogeneous or heterogeneous media. We also present two GPU-only implementations: a GPU ray-tracer capable of supporting general lighting effects, and a rasterization-only implementation that uses a custom extension of shadow-mapping to accelerate beam tracing. Both the hybrid and rasterization demos exploit a reformulation of the beam radiance estimate as a splatting operation, described in Section 6.1.

### 6.1 Hybrid Beam Splatting and Ray-Tracing Renderer

Our most general renderer combines a CPU ray tracer with GPU rasterization. The CPU ray tracer handles the photon shooting process. For radiance estimation, we decompose the light paths into ones that can be easily handled using GPU-accelerated rasterization, and handle all other light paths with the CPU ray tracer. We rasterize all photon beams which are *directly* visible by the camera. The CPU ray tracer handles the remaining light paths, such as those visible only via reflections/refractions off of objects.

We observe that Equation (3) has a simple geometric interpretation, (Figure 2): each beam is an axial-billboard [Akenine-Möller et al. 2008] facing the camera. As in the standard photon beams approach, our CPU ray tracer computes ray-billboard intersections with this representation. However, we exploit the fact that for directly-visible beams, Equation (3) can be reformulated as a splatting operation amenable to GPU rasterization.

We implemented our hybrid approach in C++ using OpenGL. In our implementation, after CPU beam shooting, we generate the photon beam billboard quad geometry for every stored beam on the CPU. This geometry is rasterized with GPU blending enabled and a simple pixel shader evaluates Equation (3) for every pixel under the support of the beam kernel on the GPU. We additionally cull the beam quads against the remaining scene geometry to avoid computing radiance from occluded beams. To handle stochastic effects such as anti-aliasing and depth-of-field, we Gaussian jitter the camera matrix in each pass. The CPU component handles all other light paths using Monte Carlo ray tracing with a single path per pixel per pass. We obtain distribution effects like glossy reflections by using different stochastic Monte Carlo paths in each pass.

For homogeneous media, the fragment shader evaluates Equation (3) using two exponentials for the transmittance. We use several layers of simplex noise [Perlin 2001] for heterogeneous media, and follow the approach derived in Section 5.2.1. For transmittance along a beam, we compute and store a fixed number $n_b$ of random propagation distances along each beam using Woodcock tracking (in practice, $n_b = 4$ to 16). Since transmittance is constant between these distances, we split each beam into $n_b$ quads before rasterizing and assign the appropriate transmittance to each segment using Equation (20). For transmittance towards the camera, we construct a progressive deep shadow map on the GPU using Woodcock tracking. This is updated before each pass, and accessed by the beam fragment shader to evaluate transmittance to the camera. We implement this by rendering to an off-screen render target and pack 4 propagation distances per pixel into a single RGBA output. We use the same random sequence to compute Woodcock tracking for each pixel within a single pass. This replaces high-frequency noise with coherent banding while remaining unbiased across many passes. This also improves performance slightly due to more coherent branching behavior in the fragment shader. We support up to $n_c = 12$ distances per pixel in our current implementation (using 3 render targets); however, we found that using a single RGBA texture is a good performance-quality compromise.

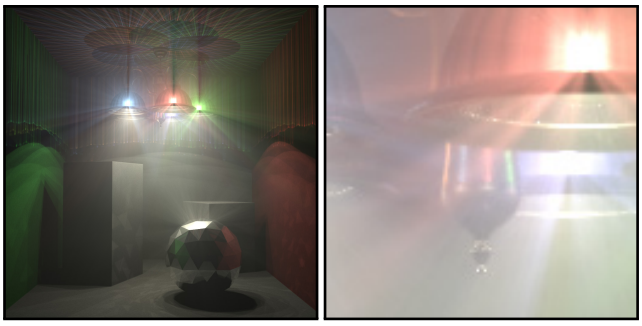

**Figure 9:** *We render media radiance in the* CHANDELIER *scene with 51.2M beams in about 3 minutes. Diffuse shading (which is not our focus) using PPM took an additional 100 minutes.*

#### 6.1.1 Results

We demonstrate our hybrid implementation on several scenes with complex lighting and measure performance on a 12-core 2.66 GHz Intel Xeon 12GB with an ATI Radeon HD 5770. In Figure 1 we show the CARS, DISCO, and FLASHLIGHTS scenes in both homogeneous and heterogeneous media, including zoomed insets of the media illumination showing the progressive refinement of our algorithm. The scenes are rendered at $1280 \times 720$, and they include depth-of-field and antialiasing. In addition, the CHANDELIER scene in Figure 9 is $1024^2$. The lights in these scenes are all modeled realistically with light sources inside reflective/refractive fixtures. All illumination encounters several specular bounces before arriving at surfaces or media, making these scenes impractical for path tracing. We use PPM for surface shading, but focus on performance and quality of the media scattering using PPB.

The CARS scene contains a few million triangles and 16 lights, all encased in glass. We obtain the results in Figure 1 in 9.5 minutes for homogeneous and in 16.8 minutes for heterogeneous media (with an additional 34.2 minutes for shading on diffuse surfaces using PPM). The DISCO scene contains a mirror disco ball illuminated by 6 lights inside realistic Fresnel-lens casings. Each Fresnel light has a complex emission distribution due to refraction, and the reflections off of the faceted sphere produce intricate volume caustics. We render the media radiance in 3 minutes in homogeneous and 5.7 minutes in heterogeneous media. The surface caustics on the wall of the scene require another 7.5 minutes. The FLASHLIGHTS scene renders in 8.0 minutes and 10.8 minutes respectively using 2.1M beams (diffuse shading takes an additional 124 minutes). These results highlight that heterogeneous media incurs only a marginal performance penalty over homogeneous media using photon beams.

Beam storage includes: start and end points ($2 \times 3$ floats), differential rays ($2 \times 2 \times 3$ floats), and power (3 floats). A scene-dependent acceleration structure is also necessary, and even for a single bounding box per beam this is $2 \times 3$ floats (we use a BVH and implicitly split beams as described in Jarosz et al. [2011]). We did not optimize our implementation for memory usage as our approach is progressive, but even with careful tuning this would likely be above 100 bytes per beam. Thus, even in simple scenes, beam storage can quickly exceed available memory: e.g., the intricate refractions in the CHANDELIER scene (Figure 9) require over 51M beams for high-quality results. This would exceed 5.2 GB of memory even with the conservative 100 bytes/beam estimate. We render this in 3 minutes using our progressive approach.

We also provide a real-time preview using our GPU rasterization: as the user modifies scene parameters, the directly visible beams are visualized at high frame rates with GPU splatting; once the user finishes editing, the CPU ray-tracer results are computed and combined with the GPU results, generating a progressively refined solution. This interaction allows immediate feedback. We demonstrate this in the accompanying video for the DISCO and SPHERECAUSTIC scenes in homogeneous and heterogeneous media.

In Figure 10 we render the SOCCER scene from Sun et al. [2010] in about a minute using our CPU/GPU hybrid. For high-quality results similar to ours, Sun et al.'s algorithm requires 73 minutes on the CPU or 6.5 minutes on the GPU. Their method only considers single scattering, whereas we also include several bounces of multiple scattering. The lines in line-space gathering are similar to photon beams; however, these lines are all blurred with a fixed width kernel (producing cylinders). Our beams use adaptive kernels with ray differentials, which explains how we obtain higher quality results using fewer beams. Also, we use rasterization for large portions of the illumination which improves performance.

Our method (16 passes) 7.5 seconds CPU+GPU | Our method (512 passes) 61 seconds CPU+GPU | [Sun et al. 2010] 73 min (CPU); 6.5 min (GPU)

**Figure 10:** *The* SOCCER *scene (Sun et al. [2010]) takes about 1 minute to render using 512K beams. We simulate complex caustics due to the glass figure but also include several bounces of* ***multiple scattering****, which Sun et al. cannot handle.*

### 6.2 Raytracing on the GPU

We also implemented our approach using the OptiX [Parker et al. 2010] GPU ray tracing API. Our OptiX renderer implements two kernels: one for photon beam shooting, and one for eye ray tracing and progressive accumulation. We shoot and store photon beams, in parallel, on the GPU. The shading kernel traces against all scene geometry and photon beams, each stored in their own BVH, with volumetric shading computed using Equation (3) at each beam intersection.

#### 6.2.1 Results

We demonstrate this implementation on the BUMPYSPHERE scene from Walter et al. [2009] shown in Figure 11 and measure performance on a 6-Core Intel Core i7 X980 3.33 GHz 12GB with an NVIDIA GTX 480 graphics card. This scene contains a deformed refractive sphere filled with a homogeneous medium. The refractions at the surface interface create intricate internal volume caustics. We render this scene at a resolution of $512^2$ at interactive rates with 1K beams per pass, and produce a crisp reference-quality solution in about 13 seconds (significantly faster than previous work [Walter et al. 2009; Jarosz et al. 2011]). The accompanying video shows this scene rendered interactively with dynamic light and camera.

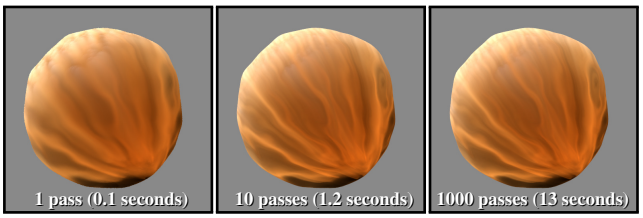


**Figure 11:** *We render the* BUMPYSPHERE *scene (Walter et al. [2009]) with the OptiX GPU ray tracing API using 1K beams per pass at interactive rates, converging (right) in 13.3 seconds.*

### 6.3 Augmented Shadow Mapping for Beam Shooting

We also implemented a real-time GPU renderer that only uses OpenGL rasterization in scenes with a limited number of specular bounces. We extend shadow mapping [Williams 1978] to trace and generate beam quads that are visualized with GPU rasterization as in Section 6.1. McGuire et al. [2009] also used shadow maps, but for *photon* splatting on surfaces. We generate and splat *beams*, exploiting our progressive framework to obtain convergent results.

We prepare a light-space projection transform (as in standard shadow mapping) used to rasterize the scene from the light's viewpoint. Instead of recording depth for each shadow map texel, each texel instead produces a photon beam. At the hit-point, we compute the origin and direction of the central beam as well as the auxiliary differential rays. The differential ray intersection points are computed using differential properties stored at the scene geometry as vertex attributes, interpolated during rasterization. Depending on whether the light is inside or outside a participating medium, beam directions are reflected and/or refracted at the scene geometry's interface. This entire process is implemented in a simple pixel shader that outputs to multiple render-targets. Depending on the number of available render targets, several (reflected, refracted, or light) beams can be generated per render pass.

We map the render target outputs to vertex buffers and render points at the beam origins. The remainder of the beam data is passed as vertex attributes to the point geometry, and a geometry shader converts each point into an axial billboard. These quads are then rendered using the same homogeneous shader as our hybrid demo in Section 6.1. To ensure convergence, the shadow map grid is jittered to produce a different set of beams each pass. This end-to-end rendering procedure is carried out entirely with GPU rasterization, and can render photon beams that emanate from the light source as well as those due to a single surface reflection/refraction. Though we currently do not support this in our implementation, two specular bounces could be handled using approximate ray tracing through geometry images as in Liktor and Dachsbacher [2011].

#### 6.3.1 Results

For interactive and accurate lighting design, we use a $64^2$ shadow map, generating 4K beams, and render progressive passes in less than 2 ms per frame for the ocean geometry with 1.3M triangles. By jittering the perspective projection, we can also incorporate antialiasing and depth-of-field effects. Since every progressive pass reduces the beam width, higher passes render significantly faster. Figure 12 illustrates the OCEAN scene, where the viewer sees light beams refracted through the ocean surface and scattering in the ocean's media. The progressive rasterization converges in less than a second. We show real-time results in the supplemental video.

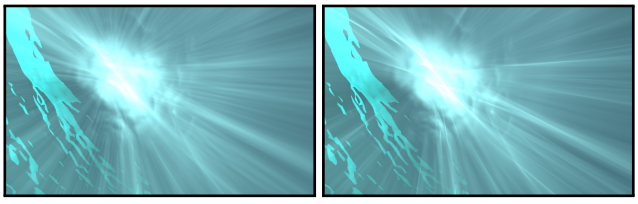

**Figure 12:** *We render the* OCEAN *scene in real-time using GPU rasterization with $4K$ beams at around 600 FPS ($<$ 2 ms). The image after 20 passes (left) renders at around 30 FPS (33 ms). The high-quality result renders in less than a second (right, 450 passes).*

## 7 Limitations and Future Work

Even with progressive deep shadow maps, the heterogeneous version of our algorithm is slower than the homogeneous form. We found that performance loss of each pass is primarily due to multiple evaluations of Perlin noise, which is quite expensive both on the GPU and CPU. Accelerated variants of Woodcock tracking [Yue et al. 2010; Szirmay-Kalos et al. 2011] could help to reduce the number of density evaluations needed for free-path sampling.

In addition to increased cost per pass, the Monte Carlo transmittance estimator increases variance per pass. Though we found that our approach worked reasonably well for our test scenes, the variance of the transmittance estimate increases with distance (where fewer random samples propagate). More precisely, at a distance where transmittance is 1%, only 1% of the beams contribute, which results in higher variance. The worse-case scenario is if both the camera and light source are very far away from the subject (or, conversely, if the medium is optically thick) since most of the beams terminate before reaching the subject, and most of the deep shadow map distances to the camera result in zero contribution. An unbiased transmittance estimator which falls off to zero in a piecewise-continuous, and not piecewise-constant, fashion could alleviate this. Another possibility is to incorporate ideas from Markov Chain Monte Carlo or adaptive sampling techniques to reduce variance.

In our implementation, $\alpha$ controls the tradeoff between reducing variance and bias and is set to a constant. The initial beam radii also significantly influence the tradeoff between bias and variance. Though our error analysis predicts this tradeoff, it is scene-independent. A scene-dependent, and spatially varying error analysis like the one by Hachisuka et al. [2010] could lead to adaptively choosing $\alpha$, or the initial beam radii, to optimize convergence.

## 8 Conclusion

We presented progressive photon beams, a new algorithm to render complex illumination in participating media. The main advantage of our algorithm is that it converges to the gold standard of rendering, i.e., unbiased, noise free solutions of the radiative transfer and the rendering equation, while being robust to complex light paths including S(D|M)S subpaths. We showed how PPM can be combined with photon beams in a simple and elegant way: in each iteration, we only need to reduce a global scaling factor that is applied to the beam radii. We presented a theoretical analysis to derive suitable scaling factors that guarantee convergence, and we empirically validated our approach. We demonstrated the flexibility and generality of our algorithm using several practical implementations: a CPU-GPU hybrid with an interactive preview option, a GPU ray tracer, and a real-time GPU renderer based on a shadow mapping approach. We also exploited a splatting formulation for beams directly visible to the camera and demonstrated how beams can be used efficiently in heterogeneous media. Our implementations underline the practical usefulness of our progressive theory and its applicability to different scenarios.

**Acknowledgements.** We are indebted to the anonymous reviewers for their valuable suggestions and the participants of the ETH/DRZ internal review for their feedback in improving early drafts of this paper. We thank Xin Sun for providing the SOCCER scene in Figure 10 and Bruce Walter for the BUMPYSPHERE scene used in Figure 11.

## References

AKENINE-MÖLLER, T., HAINES, E., AND HOFFMAN, N. 2008. *Real-Time Rendering 3rd Edition*. A. K. Peters, Ltd., Natick, MA, USA.

CHANDRASEKAR, S. 1960. *Radiative Transfer*. Dover Publications.

CHEN, J., BARAN, I., DURAND, F., AND JAROSZ, W. 2011. Real-

time volumetric shadows using 1D min-max mipmaps. In *Symposium on Interactive 3D Graphics and Games*, 39–46.

ENGELHARDT, T., AND DACHSBACHER, C. 2010. Epipolar sampling for shadows and crepuscular rays in participating media with single scattering. In *Symposium on Interactive 3D Graphics and Games*, 119–125.

ENGELHARDT, T., NOVÁK, J., AND DACHSBACHER, C. 2010. Instant multiple scattering for interactive rendering of heterogeneous participating media. Tech. rep., Karlsruhe Institut of Technology, Dec.

HACHISUKA, T., AND JENSEN, H. W. 2009. Stochastic progressive photon mapping. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2009) 28*, 5 (Dec.), 141:1–141:8.

HACHISUKA, T., OGAKI, S., AND JENSEN, H. W. 2008. Progressive photon mapping. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2008) 27*, 5 (Dec.), 130:1–130:8.

HACHISUKA, T., JAROSZ, W., AND JENSEN, H. W. 2010. A progressive error estimation framework for photon density estimation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2010) 29*, 6 (Dec.), 144:1–144:12.

HAVRAN, V., BITTNER, J., HERZOG, R., AND SEIDEL, H.-P. 2005. Ray maps for global illumination. In *Rendering Techniques 2005: (Proceedings of the Eurographics Symposium on Rendering)*, 43–54.

HERZOG, R., HAVRAN, V., KINUWAKI, S., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2007. Global illumination using photon ray splatting. *Computer Graphics Forum (Proceedings of Eurographics 2007) 26*, 3 (Sept.), 503–513.

HU, W., DONG, Z., IHRKE, I., GROSCH, T., YUAN, G., AND SEIDEL, H.-P. 2010. Interactive volume caustics in single-scattering media. In *Symposium on Interactive 3D Graphics and Games*, 109–117.

JAROSZ, W., ZWICKER, M., AND JENSEN, H. W. 2008. The beam radiance estimate for volumetric photon mapping. *Computer Graphics Forum (Proceedings of Eurographics 2008) 27*, 2 (Apr.), 557–566.

JAROSZ, W., NOWROUZEZAHRAI, D., SADEGHI, I., AND JENSEN, H. W. 2011. A comprehensive theory of volumetric radiance estimation using photon points and beams. *ACM Transactions on Graphics (Presented at SIGGRAPH 2011) 30*, 1 (Jan.), 5:1–5:19.

JENSEN, H. W., AND CHRISTENSEN, P. H. 1998. Efficient simulation of light transport in scenes with participating media using photon maps. In *SIGGRAPH '98*, 311–320.

JENSEN, H. W. 2001. *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters, Ltd., Natick, MA, USA.

KAJIYA, J. T. 1986. The rendering equation. In *Computer Graphics (Proceedings of SIGGRAPH 86)*, 143–150.

KNAUS, C., AND ZWICKER, M. 2011. Progressive photon mapping: A probabilistic approach. *ACM Transactions on Graphics (Presented at SIGGRAPH 2011) 30*, 3 (May), 25:1–25:13.

KRÜGER, J., BÜRGER, K., AND WESTERMANN, R. 2006. Interactive screen-space accurate photon tracing on gpus. In *Rendering Techniques 2006 (Proceedings of the Eurographics Workshop on Rendering)*, 319–330.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1993. Bi-directional path tracing. In *Compugraphics*, 145–153.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1996. Rendering participating media with bidirectional path tracing. In *Eurographics Rendering Workshop 1996*, 91–100.

LASTRA, M., UREÑA, C., REVELLES, J., AND MONTES, R. 2002. A particle-path based method for monte carlo density estimation. In *Rendering Techniques (Proceeeings of the Eurographics Workshop on Rendering)*.

LIKTOR, G., AND DACHSBACHER, C. 2011. Real-time volume caustics with adaptive beam tracing. In *Symposium on Interactive 3D Graphics and Games*, 47–54.

LOKOVIC, T., AND VEACH, E. 2000. Deep shadow maps. In *SIGGRAPH*, 385–392.

MCGUIRE, M., AND LUEBKE, D. 2009. Hardware-accelerated global illumination by image space photon mapping. In *Proceedings of High Performance Graphics*.

PARKER, S. G., BIGLER, J., DIETRICH, A., FRIEDRICH, H., HOBEROCK, J., LUEBKE, D., MCALLISTER, D., MCGUIRE, M., MORLEY, K., ROBISON, A., AND STICH, M. 2010. Optix: A general purpose ray tracing engine. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2010) 29*, 4 (July), 66:1–66:13.

PAULY, M., KOLLIG, T., AND KELLER, A. 2000. Metropolis light transport for participating media. In *Rendering Techniques 2000 (Proceedings of the Eurographics Workshop on Rendering)*, 11–22.

PERLIN, K. 2001. Noise hardware. In *Realtime Shading, ACM SIGGRAPH Course Notes*.

RAAB, M., SEIBERT, D., AND KELLER, A. 2008. Unbiased global illumination with participating media. In *Monte Carlo and Quasi-Monte Carlo Methods 2006*. Springer, 591–606.

SCHJØTH, L., FRISVAD, J. R., ERLEBEN, K., AND SPORRING, J. 2007. Photon differentials. In *GRAPHITE*, ACM, New York.

SILVERMAN, B. 1986. *Density Estimation for Statistics and Data Analysis*. Monographs on Statistics and Applied Probability. Chapman and Hall, New York.

SUN, X., ZHOU, K., LIN, S., AND GUO, B. 2010. Line space gathering for single scattering in large scenes. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2010) 29*, 4 (July), 54:1–54:8.

SZIRMAY-KALOS, L., TÓTH, B., AND MAGDICS, M. 2011. Free path sampling in high resolution inhomogeneous participating media. *Computer Graphics Forum 30*, 1, 85–97.

VEACH, E., AND GUIBAS, L. 1994. Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques (Proceedings of the Eurographics Workshop on Rendering)*, 147–162.

VEACH, E., AND GUIBAS, L. J. 1997. Metropolis light transport. In *Proceedings of SIGGRAPH 97*, 65–76.

WALTER, B., ZHAO, S., HOLZSCHUCH, N., AND BALA, K. 2009. Single scattering in refractive media with triangle mesh boundaries. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2009) 28*, 3 (July), 92:1–92:8.

WILLIAMS, L. 1978. Casting curved shadows on curved surfaces. *Computer Graphics (Proceedings of SIGGRAPH 78) 12* (Aug.), 270–274.

WOODCOCK, E., MURPHY, T., HEMMINGS, P., AND T.C., L. 1965. Techniques used in the GEM code for Monte Carlo neutronics calculations in reactors and other systems of complex geometry. In *Applications of Computing Methods to Reactor Problems*, Argonne National Laboratory.

YUE, Y., IWASAKI, K., CHEN, B.-Y., DOBASHI, Y., AND NISHITA, T. 2010. Unbiased, adaptive stochastic sampling for rendering inhomogeneous participating media. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2010) 29* (Dec.), 177:1–177:8.

# A Variance of Beam Radiance Estimation

The variance of the error in Equation (8) is:

$$\mathrm{Var}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)] = \mathrm{Var}\left[k_r(U)\,\gamma - L(\vec{\mathbf{w}})\right] \tag{21}$$

$$= \mathrm{Var}[k_r(U)](\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2) + \mathrm{Var}[\gamma]\mathrm{E}[k_r(U)]^2. \tag{22}$$

Using the definition of variance, we have:

$$\mathrm{Var}[k_r(U)] = \int_\Omega k_r(\xi)^2 p_U^{\vec{\mathbf{w}}}(\xi)\,d\xi - \left[\int_\Omega k_r(\xi) p_U^{\vec{\mathbf{w}}}(\xi)\,d\xi\right]^2, \tag{23}$$

where $\Omega$ is the kernel's support. We assume that locally (within $\Omega$), the distance between the beam and ray is a uniformly distributed random variable. Hence, the pdf within the kernel support is constant and equal to the probability density of an imaginary photon landing a distance 0 from our view ray $\vec{\mathbf{w}}$: $p_U^{\vec{\mathbf{w}}}(\xi) = p_U^{\vec{\mathbf{w}}}(0)$. We have:

$$\mathrm{Var}[k_r(U)] = p_U^{\vec{\mathbf{w}}}(0)\int_\Omega k_r(\xi)^2 d\xi - \left[p_U^{\vec{\mathbf{w}}}(0)\underbrace{\int_\Omega k_r(\xi)\,d\xi}_{=1}\right]^2 \tag{24}$$

$$= p_U^{\vec{\mathbf{w}}}(0)\int_\Omega k_r(\xi)^2 d\xi - p_U^{\vec{\mathbf{w}}}(0)^2$$

$$= p_U^{\vec{\mathbf{w}}}(0)\left[\int_\Omega k_r(\xi)^2 d\xi - p_U^{\vec{\mathbf{w}}}(0)\right]$$

$$= p_U^{\vec{\mathbf{w}}}(0)\left[\frac{1}{r}\int_{\mathbb{R}} k\left(\frac{\xi}{r}\right)^2 d\xi - p_U^{\vec{\mathbf{w}}}(0)\right]. \tag{25}$$

The last step replaces $k_r$ with an equivalently-shaped unit kernel $k$. Inserting into Equation (22), and noting that $\mathrm{E}[k_r(U)] = p_U^{\vec{\mathbf{w}}}(0)$ under the uniform density assumption, we have

$$\mathrm{Var}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)] = p_U^{\vec{\mathbf{w}}}(0)\left[\frac{1}{r}\int_{\mathbb{R}} k\left(\frac{\xi}{r}\right)^2 d\xi - p_U^{\vec{\mathbf{w}}}(0)\right](\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2)$$
$$+ \mathrm{Var}[\gamma]\, p_U^{\vec{\mathbf{w}}}(0)^2 \tag{26}$$

$$\approx \frac{(\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2)\, p_U^{\vec{\mathbf{w}}}(0)}{r}\left[\int_{\mathbb{R}} k\left(\frac{\xi}{r}\right)^2 d\xi\right]$$

$$= \frac{(\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2)\, p_U^{\vec{\mathbf{w}}}(0)}{r} C_1, \tag{27}$$

where the second line assumes the kernel overlaps only a small portion of the scene and hence $p_U^{\vec{\mathbf{w}}}(0)^2$ is negligible. The term remaining in square brackets is just a constant associated with the kernel, which we denote $C_1$.

# B Bias of Beam Radiance Estimation

The expected error of our single-photon beam radiance estimate is

$$\mathrm{E}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)] = \mathrm{E}[k_r(U)\gamma - L(\vec{\mathbf{w}})] = \mathrm{E}[\gamma]\mathrm{E}[k_r(U)] - L(\vec{\mathbf{w}}). \tag{28}$$

In the variance analysis in Appendix A, we assumed locally uniform densities. This is too restrictive here since it leads to zero expected error (bias). To analyze the expected error we instead use a Taylor expansion of the density around $\xi$: $p_U^{\vec{\mathbf{w}}}(\xi) = p_U^{\vec{\mathbf{w}}}(0) + (\xi)\nabla p_U^{\vec{\mathbf{w}}}(0) + \mathcal{O}(\xi^2)$, and insert into the integral for the expected value of the kernel:

$$\mathrm{E}[k_r(U)] = \frac{1}{r}\int_{\mathbb{R}} k\left(\frac{\xi}{r}\right) p_U^{\vec{\mathbf{w}}}(\xi)\,d\xi \tag{29}$$

$$= \frac{1}{r}\int_{\mathbb{R}} k\left(\frac{\xi}{r}\right)\left(p_U^{\vec{\mathbf{w}}}(0) + (\xi)\nabla p_U^{\vec{\mathbf{w}}}(0) + \mathcal{O}(\xi^2)\right) d\xi, \tag{30}$$

where we have used the same change of variable to a canonical kernel $k$. If we assume a kernel with a vanishing first moment, then the middle term drops out, resulting in

$$\mathrm{E}[k_r(U)] = p_U^{\vec{\mathbf{w}}}(0) + \frac{1}{r}\int_{\mathbb{R}} k\left(\frac{\xi}{r}\right)\mathcal{O}(\xi^2)\,d\xi \tag{31}$$

$$= p_U^{\vec{\mathbf{w}}}(0) + \frac{1}{r}\int_{\mathbb{R}} k\left(\psi\right) r\,\mathcal{O}(\psi^2)\, r\, d\psi \tag{32}$$

$$= p_U^{\vec{\mathbf{w}}}(0) + r\underbrace{\int_{\mathbb{R}} k\left(\psi\right)\mathcal{O}(\psi^2)\,d\psi}_{C_2}$$

$$= p_U^{\vec{\mathbf{w}}}(0) + r\,C_2, \tag{33}$$

for some constant $C_2$. Combining with (28), and noting an infinitesimal kernel results in the exact radiance, $L(\vec{\mathbf{w}}) = \mathrm{E}[\gamma]\mathrm{E}[\delta(U)] = \mathrm{E}[\gamma]p_U^{\vec{\mathbf{w}}}(0)$, we obtain

$$\mathrm{E}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r)] = \mathrm{E}[\gamma](p_U^{\vec{\mathbf{w}}}(0) + r\,C_2) - \mathrm{E}[\gamma]p_U^{\vec{\mathbf{w}}}(0) = r\,\mathrm{E}[\gamma]\,C_2. \tag{34}$$

# C Variance Using Many Photons

For $M$ photons in the photon beams estimate, each with their own kernel radius $r_j$, we have:

$$\mathrm{Var}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r_1\ldots r_M)] = \mathrm{Var}\left[\frac{1}{M}\sum_{j=1}^{M}\epsilon(\mathbf{x},\vec{\mathbf{w}},r_j)\right] \tag{35}$$

$$= \frac{1}{M^2}\sum_{j=1}^{M}\mathrm{Var}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r_j)]$$

$$= \frac{1}{M^2}\sum_{j=1}^{M}\left[\frac{(\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2)\, p_U^{\vec{\mathbf{w}}}(0)\, C_1}{r_j}\right] \tag{36}$$

$$= \frac{(\mathrm{Var}[\gamma] + \mathrm{E}[\gamma]^2)\, p_U^{\vec{\mathbf{w}}}(0)\, C_1}{r_H\, M}, \tag{37}$$

where we use the harmonic mean of the radii $1/r_H = \frac{1}{M}\sum\frac{1}{r_j}$ in the last step.

# D Expected Error Using Many Photons

We apply a similar procedure for expected error using many photon beams. We have:

$$\mathrm{E}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r_1\ldots r_M)] = \mathrm{E}\left[\frac{1}{M}\sum_{j=1}^{M}\epsilon(\mathbf{x},\vec{\mathbf{w}},r_j)\right] \tag{38}$$

$$= \frac{1}{M}\sum_{j=1}^{M}\mathrm{E}[\epsilon(\mathbf{x},\vec{\mathbf{w}},r_j)]$$

$$= \frac{\mathrm{E}[\gamma]\,C_2}{M}\sum_{p=1}^{M} r_j = r_A\,\mathrm{E}[\gamma]\,C_2, \tag{39}$$

where $r_A$ denotes the arithmetic mean of the beam radii.

# E Unbiased Progressive Deep Shadow Maps

Progressive deep shadow maps simply count the number of stored propagation distances, $\mathrm{d}_j$, that travel further than $s$ in each pass using Equation (20). If $p(\mathrm{d}_j)$ denotes the probability density (PDF) of the propagation distance $\mathrm{d}_j$, we have:

$$\mathrm{E}\left[\frac{1}{n}\sum_{j=0}^{n}\mathrm{H}(\mathrm{d}_j - s)\right] = \int_s^\infty p(\mathrm{d})\,d\mathrm{d} = \int_s^\infty \sigma_t(\mathrm{d})\,e^{-\tau(\mathrm{d})}d\mathrm{d} = e^{-\tau(s)}, \tag{40}$$

where $\tau(\mathrm{d}) = \int_0^{\mathrm{d}}\sigma_t(\mathbf{x}+t\vec{\omega})\,dt$ denotes the optical depth. The final result, $e^{-\tau(s)}$ is simply the definition of transmittance, confirming that averaging progressive deep shadow maps produces an unbiased estimator for transmittance.