Hierarchical Photon Mapping

Ben Spencer and Mark W. Jones

Abstract—Photon mapping is an efficient method for producing high-quality photorealistic images with full global illumination. In this paper, we present a more accurate and efficient approach to final gathering using the photon map based upon the hierarchical evaluation of the photons over each surface. We use the footprint of each gather ray to calculate the irradiance estimate area rather than deriving it from the local photon density. We then describe an efficient method for computing the irradiance from the photon map given an arbitrary estimate area. Finally, we demonstrate how the technique may be used to reduce variance and increase efficiency when sampling diffuse and glossy-specular BRDFs.

Index Terms—Photon mapping, hierarchical, final gathering, density estimation, ray tracing.

1 Introduction

The photon mapping algorithm [1] represents one of the most important advances in the computation of fast and accurate global illumination solutions. More than a decade on from its inception, the technique has become one of the most widely adopted methods in computer rendering owing to its speed, low complexity, and relative ease of implementation. Since then, a number of refinements and adaptations have been proposed, which help to reduce variance, increase speed, and decrease memory demand. Advances in computing power have also given rise to adaptations of the technique, which run at interactive frame rates on programmable graphics hardware [2].

One of the key problems encountered when computing a radiance estimate from the photon map is that of selecting an optimal kernel estimate area. Large kernels are known to produce smoother estimates at the cost of boundary bias and low accuracy on intricate geometry. Conversely, small kernels suffer from high-frequency noise. Ideally, the estimate area should correspond to the footprint of the ray at the point of intersection, however, this approach intrinsically poses problems. When the ray footprint is very small (for example, for specular rays), the classic solution is to derive an estimate area as a function of the local photon density. This keeps the estimation kernel as compact as possible while still including enough photons to prevent overly noisy estimates.

When computing diffuse and glossy-specular final gathering however, ray footprints are typically much larger. In this instance, a gather ray could potentially subtend many thousands of photons, making it prohibitively expensive to include them all in the radiance estimate (Fig. 1b). The traditional technique of using a relatively small estimate area for final gathering works well when irradiance over a surface changes gradually. However, this approach becomes less accurate when the irradiance differential is large. An example of such a scenario might be an intricate texture pattern (for example, Fig. 14) or high frequency changes in illumination from direct and caustic lighting (Figs. 12 and 13). In these cases, inadequate sampling from a small kernel size leads to higher variance, which manifests itself as visible high-frequency noise in the final image.

Our solution to this problem is to dynamically control the density of the photon map at query time using a hierarchical evaluation of the irradiance stored in the kd-tree (Fig. 1c). The main advantages of this approach are

- More accurate integration of the photon map over the ray footprint. This results in decreased noise in diffuse and glossy-specular final gathering for scenes containing complex lighting and texture patterns.
- Improved efficiency when calculating the irradiance from the photon map.
- Ease of integration with existing photon map implementations and alternative acceleration techniques. Our method only requires minor modifications to the existing kd-tree traversal algorithm. This is in addition to a preprocessing step of linear complexity to the number of photons in order to propagate irradiance and check coherence.

2 An Overview of the Photon Mapping Algorithm

The original photon mapping method is an elegant approach to solving the rendering equation [3] by caching incident radiant flux via a propagation pass, which is later referenced during the rendering pass. Photons are seeded at light-emitting surfaces and are propagated throughout the scene using standard ray tracing techniques. At each bounce, the energy carried by a photon may be stored in the photon map according to the properties of the BRDF at the surface it intersects. The resulting distribution of

- *The authors are with the Deparment of Computer Science, Swansea University, Swansea, SA2 8PP, United Kingdom. E-mail: csbenjamin, M.W.Jones}@swansea.ac.uk.*

Manuscript received 13 Sept. 2007; revised 4 Mar. 2008; accepted 17 Apr. 2008; published online 25 Apr. 2008.
Recommended for acceptance by P. Slusallek.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2007-09-0138.
Digital Object Identifier no. 10.1109/TVCG.2008.67.

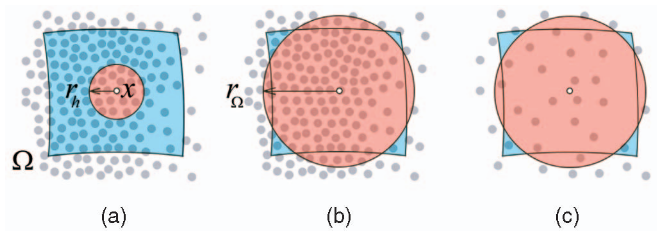

Fig. 1. (a) A fixed maximum estimate radius r_h (shaded in red) underestimates the area subtended by the ray footprint Ω (shaded in blue). (b) A new estimate radius r_Ω determined by Ω is more accurate but leads to large numbers of photon lookups. (c) Constraining the tree traverse depth reduces the number of photons in the estimate.

photons can be rapidly traversed when sorted into a hierarchical data structure such as a kd-tree.

During the rendering pass, the radiant flux stored in the photon map is used to compute the reflected radiance L_r at a given point x in direction $\overrightarrow{\omega}$ at a surface. The N nearest photons that lie approximately tangent to the surface at x are gathered, and their contribution $\Delta\Phi_p$ multiplied by the local BRDF f_r and the reciprocal of the area occupied by the photons πr^2:

$$L_r(x, \overrightarrow{\omega}) \approx \frac{1}{\pi r^2} \sum_{p=1}^{N} f_r\left(x, \overrightarrow{\omega_p'}, \overrightarrow{\omega}\right) \Delta\Phi_p\left(x, \overrightarrow{\omega_p'}\right). \qquad (1)$$

A typical photon map implementation separates flux data into two sets. The first represents high-frequency illumination from light that has been reflected or transmitted specularly. Due to a focusing phenomena, these data may exhibit regions of high density and definition, making a direct visualization of the photon map practical. When applied with adaptive density estimation and a good filter kernel, this approach yields high-quality caustics, which are relatively artifact free.

The second global photon map contains low-frequency diffuse interreflection at a relatively low density together with direct and caustic illumination. In such cases, direct visualization is often impractical since the kernel size would need to be sufficiently large in order to capture enough photons to smooth out the noise. Besides being computationally expensive, large kernels suffer from vignetting artifacts near surface edges and walls and are insensitive to discontinuities. Although algorithms have been developed to reduce these artifacts [4], [5], [6], many photon map implementations employ a final gather step to trade low frequency for high-frequency noise.

Final gathering integrates the contribution from the global photon map over the unit hemisphere by shooting gather rays and performing local lookups at ray intersections. Since errors in the radiance estimate returned by the photon map are typically obfuscated by the noise from the Monte Carlo integration, acceleration techniques such as precomputed irradiance [7] may be applied without noticeable artifacts.

3 Prior Work

Hierarchical data structures are widely used throughout computer graphics to store data at progressive levels of detail. When accessed according to an error metric, these structures may be used to reduce image artifacts and improve algorithm efficiency.

MIP mapping [8] utilizes a series of prefiltered copies of a texture at varying resolutions in order to alleviate high-frequency aliasing and sampling problems. This concept was extended independently by Benson and Davis with octree textures [9] and by DeBry et al. with octexes [10], both of which compensate for the poor performance of 2D MIP maps on complex 3D surfaces.

Jensen and Buhler [11] use a hierarchical data structure to rapidly evaluate the BSSRDF of translucent materials. Irradiance is sampled across the surface of the translucent object and is progressively stored throughout the nodes of an octree. Using approximated samples where it is appropriate, greatly accelerates evaluation of the diffusion approximation when compared to evaluating each sample individually.

The lightcuts algorithm [12] uses a binary tree to store a hierarchical evaluation of scene irradiance represented as point light sources. Clustering is used to progressively approximate groups of lights, which are stored at nonleaf nodes in the tree. Computing the incoming radiance at a given point requires making a "cut" through the tree in order to select a small subset of lights with an error below a given threshold. This approach is effective because it unifies the computation of both direct and indirect illumination, permitting highly efficient irradiance interpolation using reconstruction cuts.

The irradiance atlas [13] uses a sparse adaptive octree to represent photon maps that are too large to be held in memory. The irradiance stored at each photon is compiled into a hierarchical data structure called a brick map. This approach allows irradiance data to be cached and swapped in and out of memory efficiently and makes rendering scenes containing extremely detailed photon maps practical, even with limited memory. As a result of the progressive approximation, sampling from the brick map also benefits from filtering, resulting in a reduction in noise. Yue et al. employ a similar volumetric data structure to evaluate irradiance from surfaces [14], allowing relighting of scenes at interactive frame rates.

Although photon mapping provides a solution to many difficult problems in global illumination, the algorithm by itself is often inadequate or inefficient under complex lighting conditions. There has been a great quantity of research done into improving efficiency in all areas of the algorithm to deal with these situations, including new methods of density estimation, photon propagation, and sampling.

Poor-quality results caused by inadequate underrepresentation of illumination by the photon map is a well-studied problem. A number of solutions have been proposed, which focus on optimizing the distribution of photons prior to rendering. Visual importance sampling [15], [16] and, more recently, a technique based on the Metropolis-Hastings algorithm [17] have all shown to be effective at storing photons in a much more optimal distribution pattern. Conversely, unnecessary overrepresentation in certain areas has been addressed using density control [18] to restrict photon storage in areas of strong incident illumination.

The problem of how to efficiently sample the photon map during the rendering pass is also of special interest. Jensen [19] proposed storing the compressed direction of each photon and using it to importance sample indirect illumination. This method performs well, especially when a majority of the photons in the irradiance estimate originate from a similar direction. Further research by Hey and Purgathofer [20] improves the efficiency of this technique. Tawara et al. [21] introduced a novel method also based on importance sampling, which separates strong and weak diffuse illumination into two independent data sets together with a voxel grid containing information about photon density. Havran et al. [22] accelerate final gathering by performing the process in reverse, computing density estimations at each photon and propagating the irradiance to near-by gather ray hits.

Irradiance caching [23], [24] exploits the property that diffuse interreflection often varies gradually over Lambertian surfaces. By taking sparse yet detailed samples and interpolating between them, it becomes possible to efficiently reduce noise below perceptible levels in many scenes. This concept is extended to radiance caching [25], [26] in which incoming radiance at each sample is stored using hemispherical and spherical harmonic coefficients. This allows for much more accurate interpolation, especially over high-frequency BRDFs.

4 Our Method—Hierarchical Photon Maps

In Section 1, we outlined the problems encountered when using a constrained estimate area to compute the irradiance along final gather rays (FGRs). In this section, we propose a new method that allows us to arbitrarily increase the estimate area so that it better corresponds to the size of the footprint of the ray. In addition, we can also improve efficiency by stopping traversal at a lower density if traversing deeper into the tree would yield the same result. In order to achieve these aims, we first address several important problems:

- Selectively querying the photon map in order to limit the photon density.
- Ensuring that the integral of the power over the estimate area is accurately and correctly represented.
- Accounting for incoherent geometry for larger estimate areas.

4.1 Flux Propagation

Alongside compactness and simplicity of construction, representing a photon map using a balanced kd-tree boasts some additional useful properties. Most significantly, the absolute density of photons over a given area may be adjusted by constraining the depth to which the tree is traversed. Furthermore, using this technique still preserves *relative* photon density. Ordinarily, a kd-tree search algorithm will keep traversing the tree until it reaches each leaf node. By terminating the search at a shallower depth, however, we can effectively prune the tree and reduce the number of photons returned over an arbitrary area.

Given a balanced kd-tree, we observe that any given layer D contains approximately the same number of

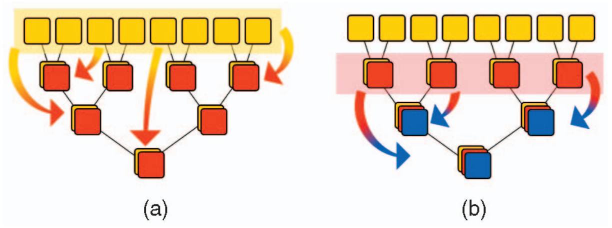

Fig. 2. Photon energy propagation. (a) The energy represented by the first cache level at each leaf photon is distributed among the second cache level in its ascendant spatially neighboring nodes. (b) The process is repeated from the second to the third cache levels and corresponding layers.

photons as the sum of the $D-1$ layers beneath it (for example, layer 4 of the tree in Fig. 2a highlighted in yellow contains 1 more photon than the sum of layers 1 to 3). Hence, querying the photon map to a traversal cutoff depth D will yield approximately twice as many photons as a query to depth $D-1$. Controlling the photon density in this manner is both simple and fast and requires minimal modification to the standard tree traversal code.

When controlling the depth to which we query the hierarchical photon map during density estimation, we cull photons from the search space. Therefore, we must ensure that the photons included in the radiance estimate also carry the exact energy lost by any excluded neighbors. To accomplish this, we allocate, for each photon residing at depth D in a tree with D_{layers} layers, $1+D_{layers}-D$ discreet energy cache levels. The purpose of these cache levels are to store the neighbors' potential contributions for each respective cutoff depth above.

All leaf photons have a single cache level, which is 1. Photons in the first layer below the leaves have two cache levels: 1 and 2. Level 1 is referenced when the traverse depth is equal to D_{layers}. Level 2 is referenced when the traverse depth is equal to $D_{layers}-1$. Photons in the second layer below the leaves have three cache levels: 1, 2, and 3. This trend continues to the root node that has D_{layers} cache levels. The sum of energies for all photons across each cache level is equal. Thus, queries to the photon map with a shallow cutoff depth will reference higher cache levels in order to compensate for the lower photon density.

We apply an energy propagation technique that operates sequentially on each layer of the tree from the leaf layer D_{layers} downward. We distribute the energy from each photon p_α in layer D by gathering the N nearest spatial neighbors in the layers above it (Figs. 2 and 3). For a uniform distribution, N should be six photons; twice the number of edges in a Delaunay triangulation of a group of points over $\mathbb{R}^2$, divided by the number of nodes.

In order to efficiently query the photon map, we use a heuristic to find an initial maximum search radius r_h that will yield an average of N photons per estimate. There are several established methods for computing this value [27]. In our implementation, we use the formula

$$r_h = \sqrt{N\bar{z}^2}, \tag{2}$$

where $\bar{z}$ is the average distance to the single nearest neighbor of a random subset of photons chosen from the map. This value of r_h will yield an average of N photons for a query

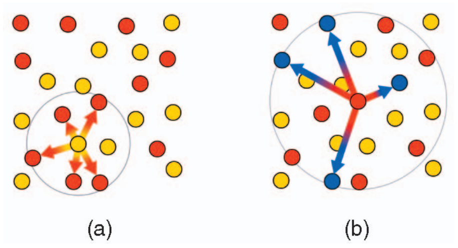

Fig. 3. Photon energy propagation. (a) First cache level photons are only distributed among their neighboring parents. (b) Photons higher up the tree are ignored.

using a traversal cutoff depth equal to $\lceil D_{tree} \rceil$, where, for a photon map containing M photons, $D_{tree} = \log_2(M+1)$.

Given that the photon density halves successively each time the traverse depth is reduced by one, we need to derive a new maximum search radius r that will still return N photons for an arbitrary traverse depth of D:

$$r = r_h 2^{\frac{1}{2}(D_{layers} - D)} \quad \text{for } 1 \leq D \leq D_{layers}. \tag{3}$$

This function assumes that the leaf layer of the tree will be full. Except in implementations where the number of photons is tightly user defined, this is unlikely to be the case. The value of r_h determined by the heuristic may not be optimal since a full tree can contain up to twice as many photons and still have the same number of layers. We adjust the heuristically determined radius to compensate for this discrepancy so that it better corresponds to a tree with a full leaf layer:

$$r'_h = \frac{r_h}{2^{\frac{1}{2}(\lceil D_{tree} \rceil - D_{tree})}}, \tag{4}$$

$$r = max\left(r_h, r'_h 2^{\frac{1}{2}(D_{tree} - D)}\right). \tag{5}$$

The energy conveyed from a given photon α to one of its N nearest neighbors β is a function of the distance of separation and the gather radius r:

$$\Phi(\alpha \rightarrow \beta) = \frac{\Phi_\alpha \times \left(r - \|x_\beta - x_\alpha\|\right)}{\sum_{\gamma=1}^{N} r - \|x_\gamma - x_\alpha\|}, \tag{6}$$

where x is the position of each photon. This approach ensures that the energy lost by not including photons after the traversal cutoff is gained by using a higher cache level.

Once the distribution step has been applied to each photon in layer D, the photons in layer $D-1$ are propagated to the layers below, and the process is repeated until the root is reached. At each iteration, the cache level in every photon below the propagated layer is initialized with the accumulated energy from the cache level beneath it to ensure energy conservation.

Algorithm 1. PROPAGATEENERGY()

```
1:  for $n_{cache} = 1$ to $D_{layers}$ do
2:    $D_{prop} = 1 + D_{layers} - n_{cache}$
3:    $D_{cutoff} = D_{prop} - 1$
4:    $r = max(r_h, r'_h 2^{\frac{1}{2} n_{cache}})$
5:    for $D = 1$ to $D_{cutoff}$ do
6:      for all $\alpha \in$ TREELAYER$[D]$ do
7:        $\Phi_\alpha[n_{cache}+1] = \Phi_\alpha[n_{cache}]$
8:      end for
9:    end for
10:   for all $\alpha \in$ TreeLayer$[D_{prop}]$ do
11:     $G \leftarrow$ GATHERPHOTONS$(x_\alpha, r, D_{cutoff})$
12:     for all $\beta \in G$ do
13:       $\Phi_\beta[n_{cache}+1] += \Phi(\alpha[n_{cache}] \rightarrow \beta)$
14:     end for
15:   end for
16: end for
```

Algorithm 1 provides a pseudocode example of the propagation step of our algorithm. The function GATHERPHOTONS(x, r, d_{cutoff}) returns a list of photons within radius r to point x, traversing the tree to a cutoff depth of d_{cutoff}. The value D_{prop} represents the depth of the propagated layer. The object TREELAYER represents an array of lists corresponding to the photons in each layer of the kd-tree, where TREELAYER$[D_{layers}]$ are the leaf nodes. $\Phi_p[n]$ represents the flux of photon p at cache level n.

4.2 Normal Coherence Checking

By allowing arbitrarily large estimate radii, it becomes much more probable that the assumption of a locally flat surface will no longer hold. Disregarding incoherent geometry can result in a number of problems, most notably that of boundary bias at edges and corners, which can cause perceptible energy loss in many scenes. In addition, light and shadow leaks under geometry may also appear in certain models (see [9], [10], and [13] for examples).

To overcome this problem, we apply normal coherence checking. For every photon within a given radiance estimate, we analyze the dot product, Δ_n, between the photon normal and that of the surface at the point of origin of the query, x. If any value of Δ_n lies beneath a given threshold, ϵ_n, then the estimate is determined to be incoherent. The purpose of this check is to instruct the tree traversal algorithm to reduce the search radius and to increase the traverse depth if the photons local to x are not coherent at the desired level in the hierarchy. In our implementation, we found that the threshold value used by Christensen and Batali [13] of $\epsilon_n = 0.7$ (45 degrees) worked well.

4.3 Irradiance Precomputation

In order to further improve performance, we can extend the concept of precomputed irradiance [7] to hierarchical photon maps. While this technique is limited to representing scattering from Lambertian surfaces, the performance benefits make this an acceptable compromise in many situations.

Once the flux has been propagated throughout the tree, we precompute a set of irradiance estimates at each photon, which directly correspond to the associated cache levels. The estimate at the lowest level uses a radius of r_h and a traversal cutoff depth equal to D_{layers}. Estimates at progressively higher levels are found by querying the photon map using shallower cutoff depths and wider search radii (5). In addition, we also calculate the normal coherence of the photons during each iteration. If a value of Δ_n is less than ϵ_n, we store the cutoff depth associated with the incoherent estimate in the photon. This flag is referenced during tree traversal, details of which are presented in Section 4.6.

Calculating the irradiance at any given point is now reduced to simply finding the nearest photon. This makes

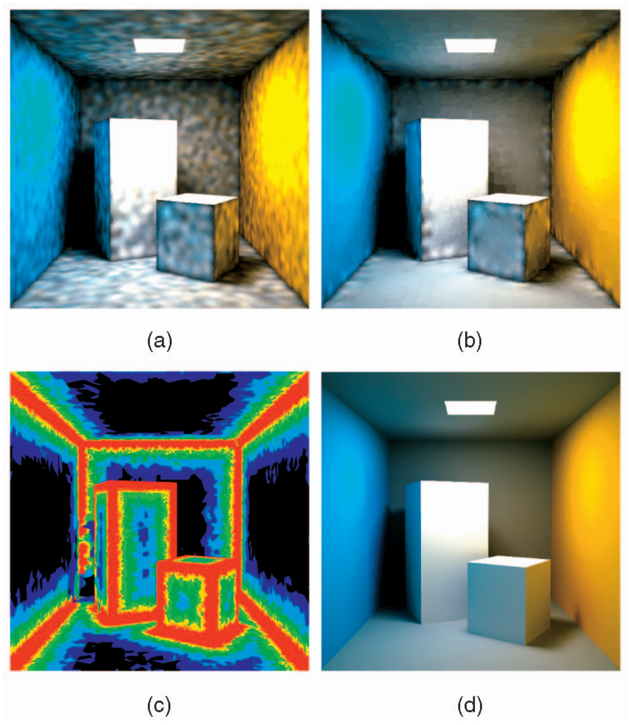

Fig. 4. Classic Cornell Box. In this example, the global photon map contains approximately 200,000 photons with 100 in the radiance estimate. (a) A direct visualization of a conventional photon map. (b) A direct visualization of a hierarchical photon map. (c) A color-coded visualization of the coherence at each point. Here, red areas represent regions of low coherence (in corners and shadow penumbras). (d) Rendered with full global illumination and tone mapped.

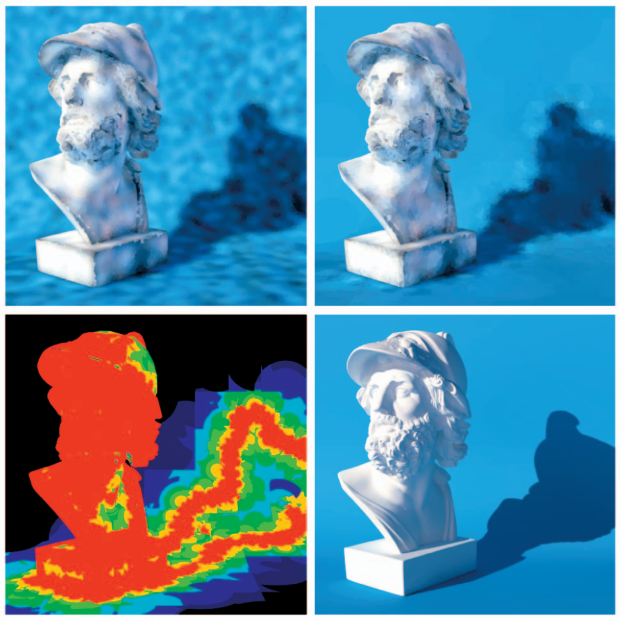

Fig. 5. Ajax Plaster Bust in Sunlight. This scene is designed to highlight the irradiance coherence checking. Notice how the photons are flagged as incoherent around the shadow penumbra. The global photon map contains approximately 250,000 photons with 100 in the radiance estimate. Less than 20 percent of the photons were referenced at render time. The frames in this figure correspond to those in Fig. 4.

the technique extremely fast since maintaining a max heap and filtering the radiance estimate at each query no longer becomes necessary. Moreover, the errors introduced by using a cruder lookup scheme are generally smoothed out by primary and secondary final gathering.

4.4 Irradiance Coherence Checking

It is commonly acknowledged that many scenes exhibit large regions of relatively uniform diffuse illumination. Aside from benefiting from the better integration of the photon map over the ray footprint, we can also adaptively search the kd-tree according to how alike the certain regions of illumination are. For example, a blank stretch of wall might exhibit a near-constant irradiance across its surface. Therefore, for Lambertian BRDFs, using a propagated value stored lower down in the tree will render the same results at a decreased traversal cost when compared to a value stored further up toward the leaves.

While it is possible to take advantage of uniform regions of illumination, we must also preserve discontinuities and details. Determining whether a photon is radiantly coherent for a given cache level may be accomplished at the same time as normal coherence checking for a marginal extra cost. By analyzing the local irradiance of each photon used in the irradiance estimate for each cache level, we can determine whether or not the illumination local to the photon is coherent.

To compare between irradiance values we transform the native RGB color coordinates used by our photon map into CIE XYZ color space and then into normalized xyY space. Using this representation, Y defines the luminance, and x and y define coordinates on the chromaticity plane. To determine whether an estimate at α is coherent, we compute two values, Δ_Y and Δ_{xy}, based upon the set of photons N that lie within the gather radius of the given cache level:

$$\left.\begin{aligned} \Delta_Y &= \max\left(\tfrac{|Y_\gamma - \overline{Y}|}{\overline{Y}}\right) \\ \Delta_{xy} &= \max\left(\|xy_\gamma - xy_\alpha\|\right) \end{aligned}\right\} \forall \gamma \in N. \tag{7}$$

Here, $\overline{Y}$ is the mean luminosity of all photons in N. Thus, Δ_Y represents the maximum normalized variance in irradiance luminosity at cache level 1. Δ_{xy} represents the maximum change in chromaticity between α and each photon in N. If either value exceeds thresholds ϵ_Y or ϵ_{xy}, respectively, then the estimate is determined to be incoherent. We found that a threshold value of $\epsilon_{xy} = 0.2$ worked well in all our test scenes. Choosing a suitable threshold for Δ_Y is more difficult because of the significant background noise present in the photon density estimation (see Figs. 4a and 5a). We want our error metric to be able to overlook this noise in uniform areas while still being sensitive to more prominent changes. We use a heuristic for determining a threshold value ϵ_Y, based upon the values of Δ_Y throughout the photon map:

$$\epsilon_Y = \overline{\Delta_Y} + \tau\sqrt{\frac{2}{M}\sum_{\gamma=1}^{M}(\Delta_{Y_\gamma} - \overline{\Delta_Y})^2}, \tag{8}$$

where M is random subset of photons, and τ is a user-defined value for adjusting the sensitivity of the heuristic. We use a value of $\tau = 2.0$ in all our test scenes. Our function estimates the peak background noise by treating the signal as a sinusoidal waveform and using the standard deviation to find a good upper bound. Given that the frequency of the noise in the irradiance estimate increases as the number of photons in the estimate decreases, it is also possible to increase the accuracy of the heuristic at the expense of fine detail. This can be achieved by looking up irradiance values in higher cache levels when calculating the values of Δ_Y. Notice how our error metric uses relative rather than absolute photon luminance to determine coherence. We chose this approach since it offered more intuitive control over the coherence threshold by comparing local photon intensities.

We store irradiance coherence values alongside the normal coherence in each photon. The irradiance coherence is then constrained so that it is not less than the normal coherence. Fig. 5 further demonstrates the concept of coherence checking. In this scene, the surface on which the bust is sitting exhibits a near-constant irradiance, except in the shadow penumbra. Notice how our method uses a smoother propagated approximation where possible and tightens the search space in areas of high variance. Removal of background noise also offers the additional benefit of lower variance throughout the photon map. This is particularly desirable when casting FGRs from high-frequency BRDFs, in which noise from density estimation is sometimes detectable.

4.5 Final Gathering

Calculating the area covered by the footprint of a FGR may be accomplished using ray differentials [28]. This method is very accurate since the change in differentials due to reflection and refraction can be accurately modeled. In our implementation, we use a simpler method for diffuse reflection based on the length of the ray, the solid angle Ω, and the orientation of the normal to the angle of incidence:

$$r_\Omega = \sqrt{\frac{i^2}{-d \cdot n}\left(\frac{1}{\left(1-\frac{\Omega}{2\pi}\right)^2} - 1\right)}. \tag{9}$$

Here, d represents the normalized direction of the ray, n represents the surface normal at the intersection point, and i represents the parametric distance of intersection along d.

Calculating the value of Ω depends on the sampling method used when evaluating the incoming radiance and the BRDF. In our implementation, the distribution of N gather rays over the unit hemisphere is determined by a probability density function $p(\theta, \phi)$, where

$$\frac{1}{N} \approx \int_\Omega p(\theta, \phi) d\omega. \tag{10}$$

Since the distribution of samples over the hemisphere is nonuniform, we use this function to calculate the solid angle of a ray in direction (θ, ϕ) based on the probability density in that region:

$$\Omega = \frac{1}{Np(\theta, \phi)}. \tag{11}$$

By rearranging (5), we find the derived footprint traverse depth D_Ω in a tree of depth D_{tree}, for any given radius r_Ω:

$$D_\Omega = D_{tree} - \max\left(0, 2\log_2\left(\frac{r_\Omega}{r_h}\right)\right). \tag{12}$$

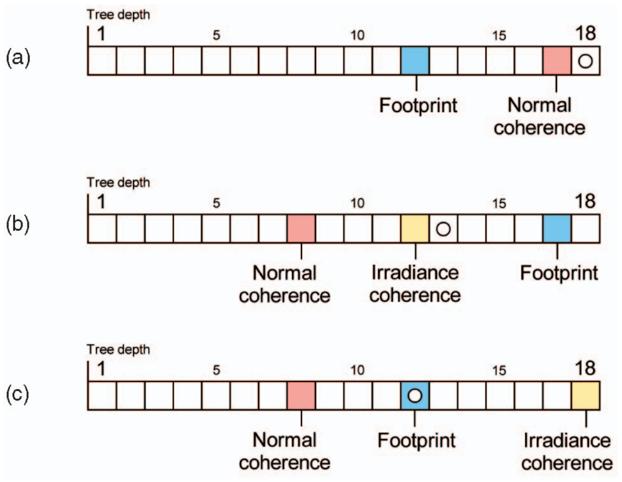

Fig. 6. Photon map coherence scenarios. (a) The ray footprint limits the traverse depth to 12, however, the nearest photon is only coherent at depth 18 (the leaf nodes). Normal coherence takes precedent, and so, the estimate area is reduced, and the traverse depth increased to 18. (b) The ray footprint has a traverse depth of 17; however, the irradiance is coherent at depth 13 and below. To save time, the search is terminated at depth 13. (c) The irradiance is completely incoherent; however, the ray footprint instructs termination at depth 12. Since the normals are coherent at a shallower depth than the footprint, the search is ended at depth 12.

4.6 Tree Traversal

In this section, we outline the key concepts of traversing the kd-tree of a hierarchical photon map. Since we use precomputed irradiance in all of our test renders, the following examples demonstrate searching for the single nearest coherent photon. Performing density estimation at render time requires finding *all* photons with the given search radius, however, the principles are the same as those described below.

Our tree traversal algorithm is based upon that described by Jensen in [29]. Finding the photons nearest to the point of intersection requires choosing an initial minimum valid radius within which to search. The kd-tree is then recursively traversed, with photons and subtrees lying outside the search radius being ignored. As new closer photons are discovered, the search radius is reduced accordingly such that it is always equal to the proximity of the nearest photon.

Traversing the kd-tree of a hierarchical photon map differs slightly from that of a conventional photon map, since we must allow for on-the-fly reduction in the search space without significantly impacting performance. For example, if the nearest photon at cutoff depth 10 is only coherent at depth 15 and below, then the search radius must be reduced and the tree searched further so as to ensure a finer degree of granularity. As each new nearest photon β is found, its coherence depth is compared with the current traversal cutoff depth D. Three comparisons are then performed:

- If the cutoff depth is **greater** than the irradiance coherence depth C_L, then the search is terminated (Fig. 6b). This is efficient because the chosen photon does not necessarily need to be the photon nearest to x so long as it is coherent.

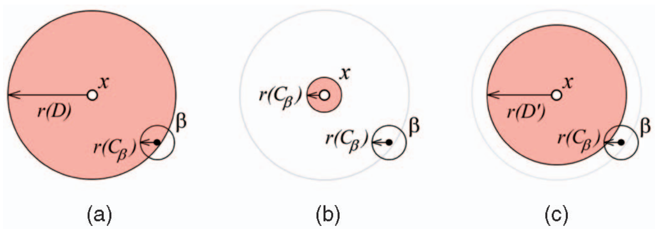

Fig. 7. Coherence interpolation. (a) A new nearest photon β is located near the edge of the search radius $r(D)$. (b) Without interpolation, the new search radius is snapped to that of the photon, $r(C_\beta)$. This is not indicative of the coherence at x since the incoherent photon resides well outside the search range at this level of granularity. (c) With interpolation, a new search radius $r(D')$ is derived that blends the existing traverse depth with the coherence at β.

- If the cutoff depth is **less than or equal** to the normal coherence C_n, then the cutoff depth is increased accordingly (Fig. 6a). Given a point of intersection x, we calculate the new depth D' as being a function of the current depth D, the position x_β and the two search radii derived from the normal coherence value of the new nearest photon $r(C_{n\beta})$, and the current depth $r(D)$ (5):

$$D' = D + (C_{n\beta} - D)\left(1 - \frac{\max\big(0, \|x_\beta - x\| - r(C_{n\beta})\big)}{r(D)}\right). \tag{13}$$

  Fig. 7 demonstrates why this interpolation function is important.
- If the cutoff depth is **less than or equal** to the irradiance coherence, then (13) is applied using irradiance coherence C_L. The derived cutoff depth is also capped at the ray footprint depth D_Ω (Fig. 6c).

When exploiting irradiance coherence, choosing an initial traverse depth and associated search radius is also important since too many search space reductions can impede performance. This is especially true if the scene is highly incoherent. We found that the mean photon coherence per unit surface area worked well as an initial search depth. We call this value D_{mean} and define it as

$$D_{mean} = \frac{\sum_{\gamma=1}^{R} C_\gamma 2^{-C_\gamma}}{\sum_{\gamma=1}^{R} 2^{-C_\gamma}}. \tag{14}$$

Here, R is the subset of all photons in M with known coherence depths. Since the initial search radius is based on the mean coherence, we can use (5) to derive it. The value of r_h should not exceed more than half that used to compute the irradiance estimates at each photon. This prevents incorrect coherence cutoffs, which would occur if a new nearest photon were at a greater distance from x than its coherence check radius.

Algorithm 2. FINDNEAREST(PHOTON)

```
Ensure: $D_{cutoff} = \min(D_{mean}, D_\Omega)$
Ensure: Nearest.Distance $= \frac{1}{2} r(D_{cutoff})$
 1: if Photon.Distance $<$ Nearest.Distance and
    Photon.Normal is coherent to $x$ then
 2:   Nearest = Photon
 3:   if $D_{cutoff} >$ Photon.$C_L$ then
 4:     stop ... this photon is coherent so stop searching
 5:   else if $D_{cutoff} \leq$ Photon.$C_n$ then
 6:     $D_{cutoff} = D'_{cutoff}$(Photon.$C_n$)
 7:   else if $D_{cutoff} \leq$ Photon.$C_L$ then
 8:     $D_{cutoff} = \min(D'_{cutoff}(\text{Photon.C}_\text{L}), \text{D}_\Omega)$
 9:   end if
10: end if
11: if Photon.Depth $= D_{cutoff}$ then
12:   return
13: end if
14: if $x$ is in Photon.LeftPartition then
15:   FindNearest(Photon.LeftChild)
16:   if Photon.RightPartition.Distance $<$ Nearest.Distance
      then
17:     FindNearest(Photon.RightChild)
18:   end if
19: else
20:   ... repeat for right-hand partition
21: end if
```

Algorithm 2 gives pseudocode describing a recursive tree traversal function that finds the nearest photon to a given point, x.

5 Results and Discussion

We tested our algorithm on a number of scenes which exhibit complex direct and indirect lighting conditions. For comparison with our algorithm, we used the classic photon map implementation, as described by Jensen in [29], using precomputed irradiance, as described by Christensen in [7]. To balance our tree, we split on the median photon of each partition, perpendicular to the dimension with the greatest photon spatial variance. All irradiance estimates were computed using kernel density estimation and smoothed using a Gaussian filter.

Since our algorithm is most efficient when using precomputed irradiance, we ran all tests using this method; both hierarchical and nonhierarchical. Both techniques required a standard preprocess to compute the irradiance values at each photon (n density estimations for n photons). When using our hierarchical technique, each scene required additional preprocessing to handle photon energy propagation and precomputation for the multiple layers in the hierarchy (between n and $2n$ extra queries depending on the population of the leaf layer of the tree). Except when the number of rays cast is comparatively small, this overhead is more than an offset by the time saved by our method.

Table 1 shows the results concerning Figs. 15, 16, and 17. Here, photon map querying μ gives the speedup of the hierarchical method over the conventional method timed for the photon map lookups only. Final gathering μ gives the speedup of our method for the diffuse component. Total rendering time μ gives the speedup for all illumination components (excluding precomputation). Given that our method accelerates photon map lookups, net speedup is highly dependent on the distribution of processing time among rendering components (ray tracing, shading, etc.). We therefore propose that the photon map querying

TABLE 1
Timing Figures for Scenes Rendered with Irradiance Coherence

	Cornell Box		Sponza Atrium		Sunlit Living Room	
Image Resolution	600 x 600		840 x 525		640 x 480	
Polygons	132		71,500		82,200	
Photons	501K		1,167K		1,056K	
Final gather rays	1000		1000		1000	
Radiance estimate	50		50		50	
Total rendering time μ	1.34		1.20		1.17	
Final gathering μ	1.38		1.22		1.19	
Photon map querying μ	**1.98**		**1.62**		**1.69**	
	Standard	Hierarch.	Standard	Hierarch.	Standard	Hierarch.
Φ propagation	-	3.4s	-	21.2s	-	24.1s
Irradiance precomputation	12.7s	24.4s	29.0s	82.9s	29.8s	64.5s
Photons referenced during render	100.0%	20.83%	93.73%	61.03%	95.55%	49.33%

speedup is the important figure to consider when measuring the efficiency of this method.

In addition, we also demonstrate that the precomputation overhead is reasonable by providing timings for standard irradiance precomputation against the preprocessing required by our method. We also supply data about the percentage of the photon map referenced using both methods. This demonstrates that the irradiance coherence checking is working well.

Using Ward's compressed RGBE format [30], each photon at depth D requires $4 + 4(D_{layers} - D)$ bytes to store the cache hierarchy and 2 bytes to store the normal and irradiance coherence. In total, this averages out to an extra $6n$ bytes per n photons in addition to the overhead from the standard data structure outlined in [29].

5.1 Error Testing

Results are presented that compare the root mean square (RMS) error of conventional photon mapping and hierarchical photon mapping to a converged conventional photon mapping image. For each test, the scene is rendered with increasing numbers of FGRs per pixel (from 20 to 2,980). The test scene is also rendered using our method and the same, progressively increasing number of FGRs. The RMS error between the rendered images and a converged image rendered by conventional photon mapping is plotted on a graph. Fig. 8 shows the results for the glossy Buddha scene (rendered in Fig. 11), Fig. 9 shows the results for the submerged Cornell box test (rendered in Fig. 12), and Fig. 10 shows the results for the living room with a lantern scene (rendered in Fig. 13).

In all cases, our new hierarchical method produces a rendered image that has substantially less error than conventional photon mapping for the same number of FGRs. This is observable in the rendered images and quantitatively from the RMSE tests presented in the graphs (Figs. 8a, 9a, and 10a). The dual of this is that we can produce an image similar in appearance (similar in error) with fewer FGRs than the conventional method and therefore achieve speedup of μ. This is demonstrated in the graphs (Figs. 8b, 9b, and 10b). Each graph shows the speedup, μ, achieved by the hierarchical method over the standard method using a number of FGRs for the hierarchical method chosen to create an image of equal RMSE.

The graphs (Figs. 8a, 9a, and 10a) also show that the new hierarchical method demonstrates good convergence to the converged render. The substantially reduced error is a

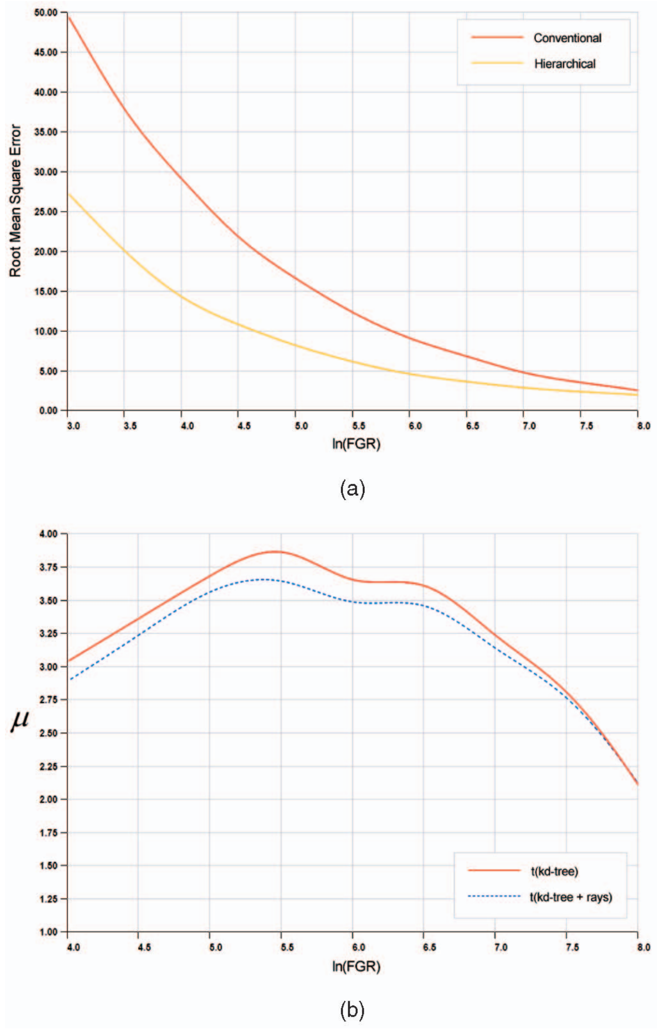

Fig. 8. (a) The RMS error between a series of renders at various sample levels and a converged render of the glossy Buddha scene. ln(FGR) is the natural log of the number of FGRs per pixel. (b) A speedup of $\mu\times$ is achieved by hierarchical photon mapping for rendering an image with the same RMSE as the conventional method. ln(FGR) refers to the FGRs used by the conventional method. For further explanation of the graphs, see Section 5.1. Rendered image shown in Fig. 11.

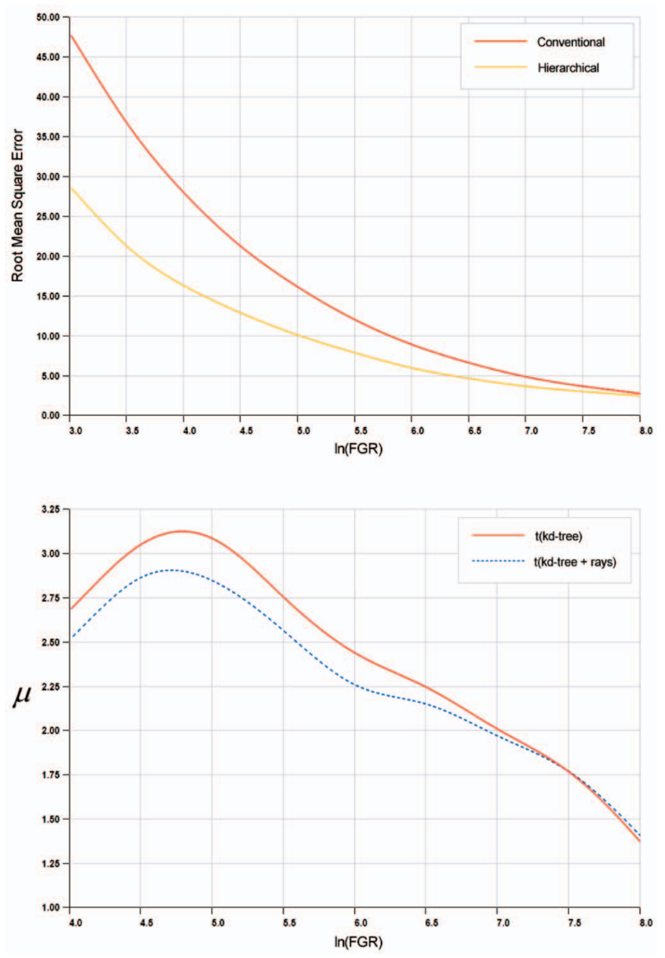

Fig. 9. RMSE and timing figures corresponding to Fig. 8 for the submerged Cornell box scene. Rendered image shown in Fig. 12.

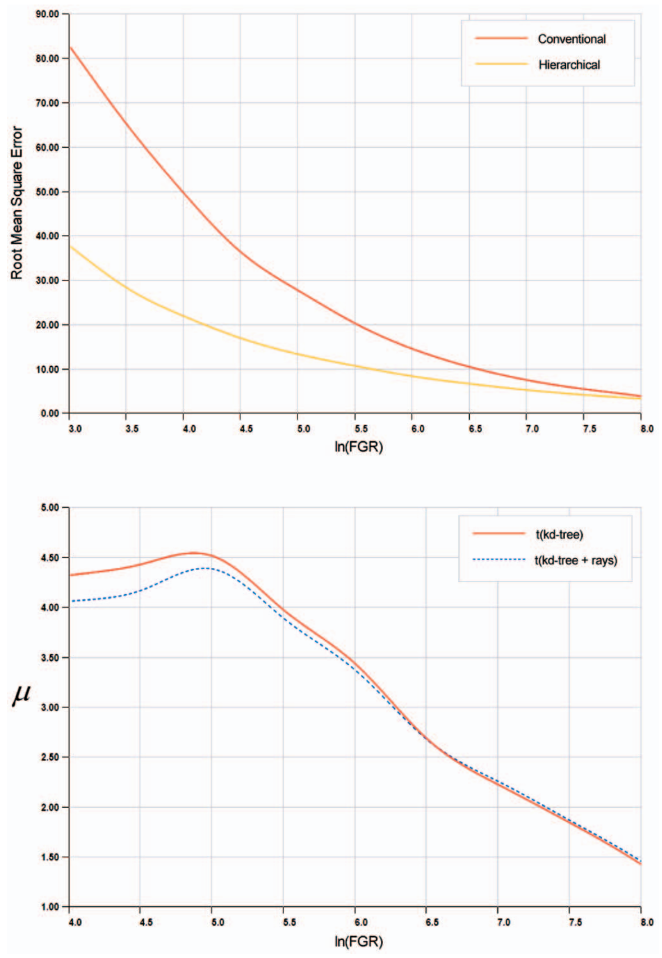

Fig. 10. RMSE and timing figures corresponding to Fig. 8 for the living room with lantern scene. Rendered image shown in Fig. 13.

result of integrating over an area of the footprint that more closely corresponds to the footprint of each ray.

5.2 About the Images

Fig. 11 is an extreme case whereby a large proportion of the scene exhibits a complex texture pattern. In addition to casting photons from emitting surfaces, we also baked the irradiance of each emitter using photons. During final gathering, only the photon map is used to determine outgoing radiance; the emissive properties of the material are not directly referenced. Notice how the rapid changes in color bleed around the edges of the walls are still accurately rendered. Fig. 14 is another example of a complex emitter. In this scene, we again bake the irradiance at the surface of the model into the photon map. The noise in the glossy reflection of the bunny on the ground plane is significantly reduced by the hierarchical evaluation.

Fig. 12 contains caustics produced by light being refracted and focused in a water-filled Cornell box. The high-frequency variation in the caustics illumination appears as noise on the walls and ceiling when rendered with the traditional photon mapping approach (Fig. 12a). Integrating over the area subtended by the solid angle of each ray using our hierarchical technique helps reduce the noise to a uniform level across the entire image.

Fig. 13 is a practical example of the algorithm at work. The living room is lit by a child's lantern that projects a pattern onto the walls and ceiling. Once again, the high-frequency changes in the direct illumination cause noise to appear in Fig. 13a, which is successfully reduced by our technique.

Fig. 15 is an example of how a highly coherent scene can improve rendering time. In this case, most surfaces exhibit smooth changes in illumination, which equates to a photon map query speedup of 1.98 compared to conventional final gathering. Figs. 16 and 17 are geometrically intricate models rendered with a high number of rays per pixel. Both scenes contain objects with incoherent irradiance and normals; however, our method is still faster than conventional photon mapping despite this complexity.

5.3 Discussion

Using the hierarchical method ensures that rapid, local changes in illumination are accurately rendered while reducing the variance for more distant surfaces. This results in a large overall reduction in variance across the whole image. The main results are the following:

- Noise within the image is substantially reduced for the same number of samples per pixel (Figs. 11, 12, 13, and 14 and the graphs in Figs. 8a, 9a, and 10a).

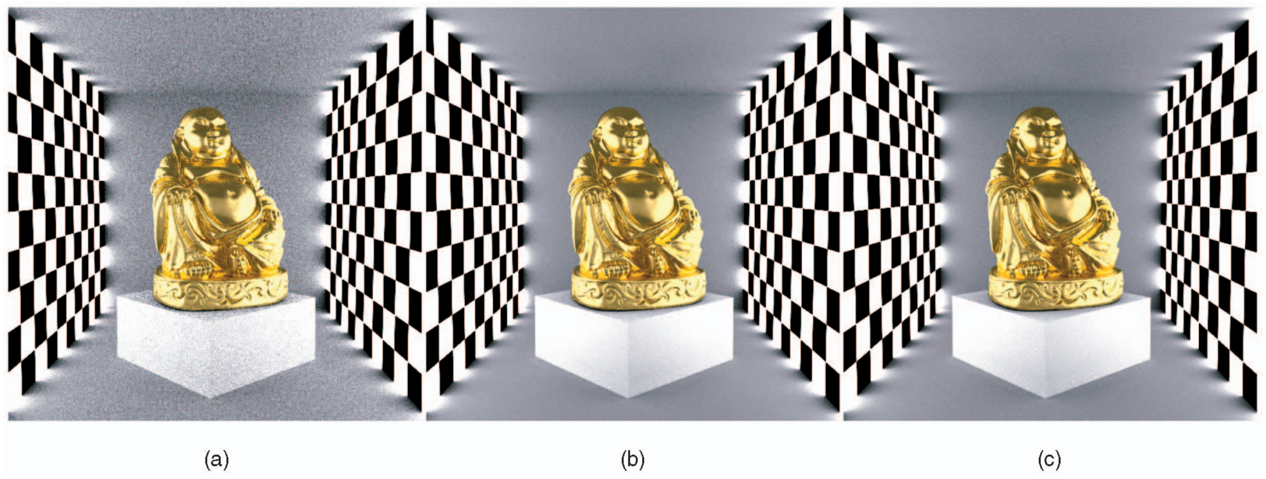

Fig. 11. Cornell Box with Glossy Buddha. 600 × 600 pixels. 414K photons (50 in the radiance estimate). (a) Conventional photon mapping, 50 FGRs per pixel. (b) Hierarchical photon mapping, 50 FGRs per pixel. (c) Conventional photon mapping, 300 FGRs per pixel (same visual quality as (b)).

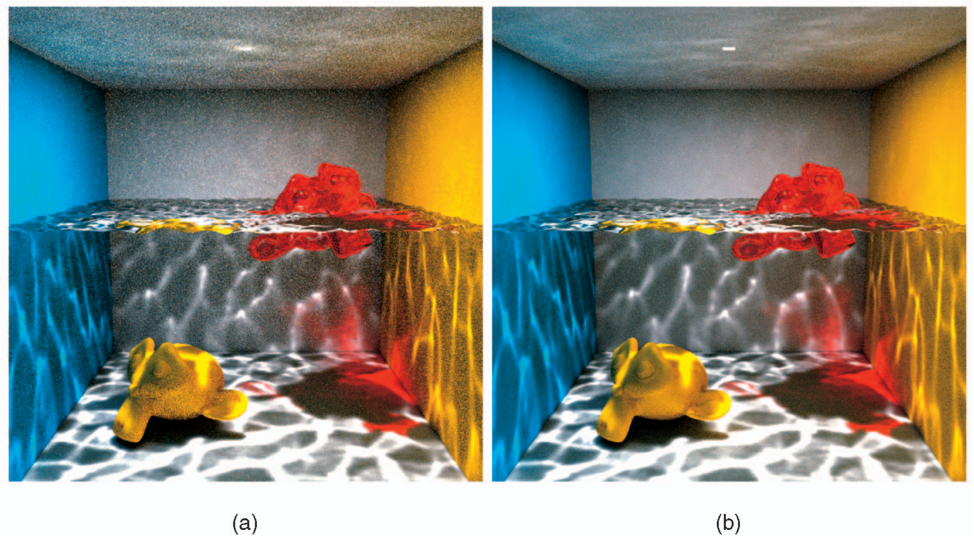

Fig. 12. Submerged Cornell Box. 600 × 600 pixels. 20 FGRs per pixel. 500K photons (50 in the radiance estimate). (a) Conventional photon mapping. (b) Hierarchical photon mapping.

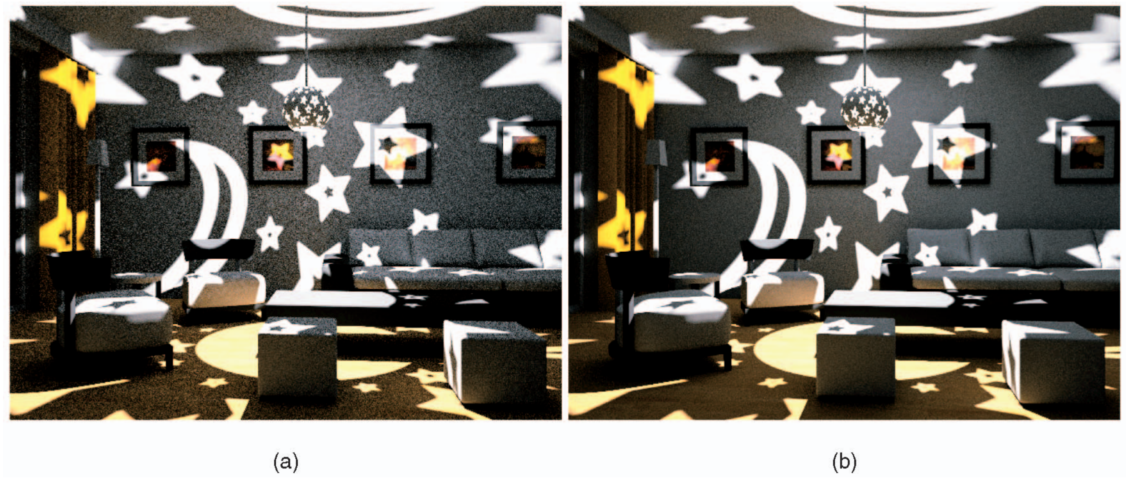

Fig. 13. Living Room with Lantern. 640 × 480 pixels. 50 FGRs per pixel. 576K photons (50 in the radiance estimate). (a) Conventional photon mapping. (b) Hierarchical photon mapping.

- Identical quality images can be produced by the hierarchical method in less time (Figs. 15, 16, and 17 and the graphs in Figs. 8b, 9b, and 10b).
- When considering render times, our method will take less time than traditional photon mapping using precomputed irradiance for the same number

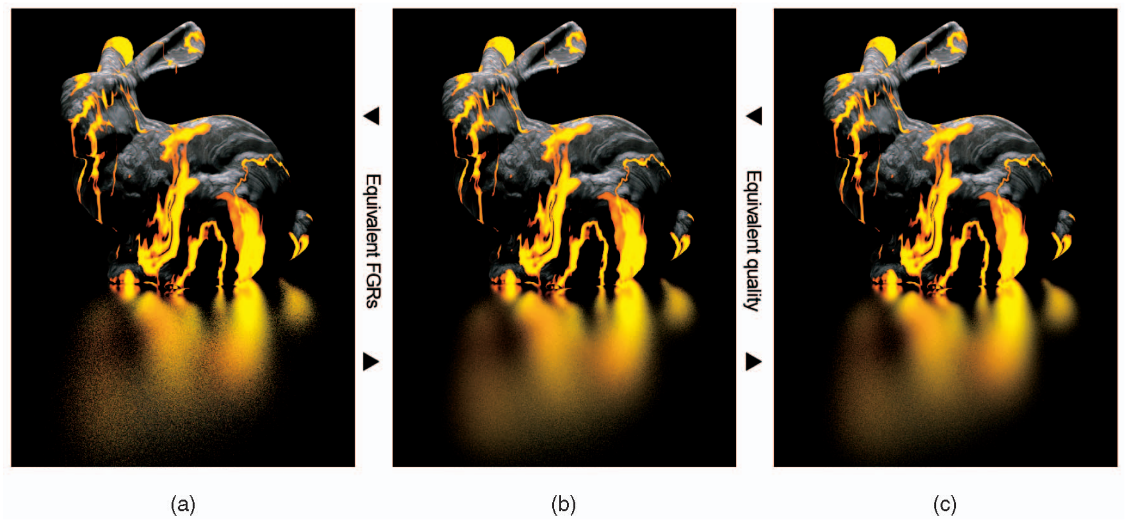

Fig. 14. Magma Stanford Bunny. 480 $\times$ 640 pixels. 400K photons (30 in the radiance estimate). (a) Path tracing, 20 FGRs per pixel. (b) Hierarchical photon mapping, 20 samples per pixel. (c) Path tracing, 80 samples per pixel.

of samples. This is a result of the depth pruning technique used to control the photon density. Only near-by surfaces, as well as those with low coherence, require the entire kd-tree be traversed. More distant objects and those with coherent normals and illumination use a more approximate solution stored further down the tree. Except for scenes rendered using comparatively few FGRs, the gain will outweigh the preprocess even for single images.

- Our algorithm may be easily combined with other acceleration and quality improvement techniques for photon mapping, since the tree traversal and density estimation algorithms used in the standard implementation are not extensively modified.

At its core, our algorithm utilizes a hierarchical paradigm that forms the basis of many algorithms in computer graphics and global illumination (see Section 3). Of these, the irradiance atlas [13] is of particular relevance since it, too, employs an adaptively sampled, hierarchical data structure derived from the photon map. The irradiance atlas is explicitly designed to handle scenes with very large numbers of photons. Thus, opting for this approach would prove advantageous as memory demands exceed system resources. Furthermore, the modular nature and size of the brick data structures are specifically designed for efficient memory access. Since our method does not cluster samples together in this fashion, traversing the photon tree would require more extensive swapping between cache and memory. Another key difference between the two algorithms is the method used to compute the radiance estimate. The irradiance atlas stores precomputed estimates derived from an underlying photon map and interpolates between voxels overlapping the ray footprint to determine the outgoing radiance. The hierarchical photon map, however, operates on the original photon data. This means that illumination gathered from surfaces with non-Lambertian BRDFs can still be accurately evaluated using density estimation. As our tests demonstrate, a precomputed solution is also effective for illumination scattered from Lambertian surfaces.

While using an arbitrary area when estimating irradiance is, in many cases, superior to using a fixed maximum, objectionable errors may still be introduced if the size of a high-energy interval in the illumination integral drops

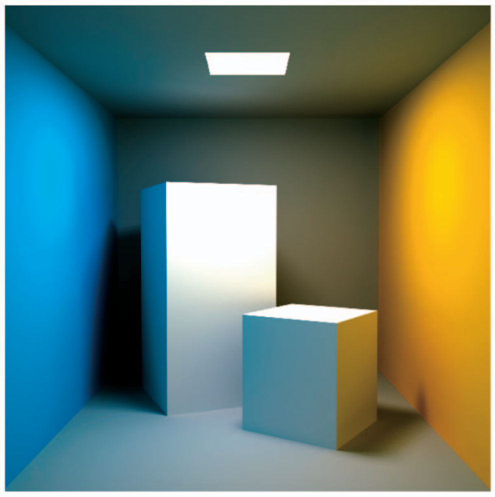

Fig. 15. Classic Cornell Box. Data about this image can be found in Table 1.

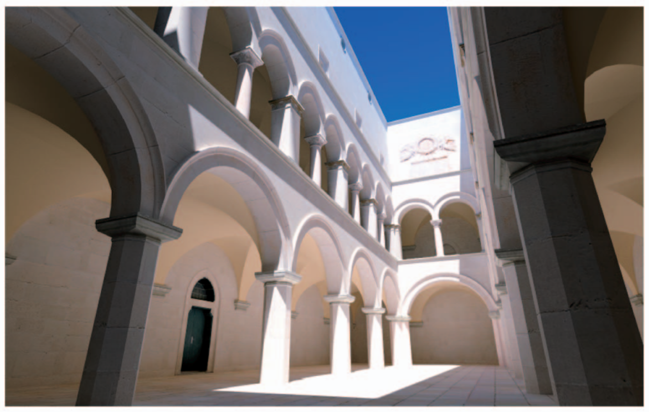

Fig. 16. The Sponza Atrium. Data about this image can be found in Table 1.

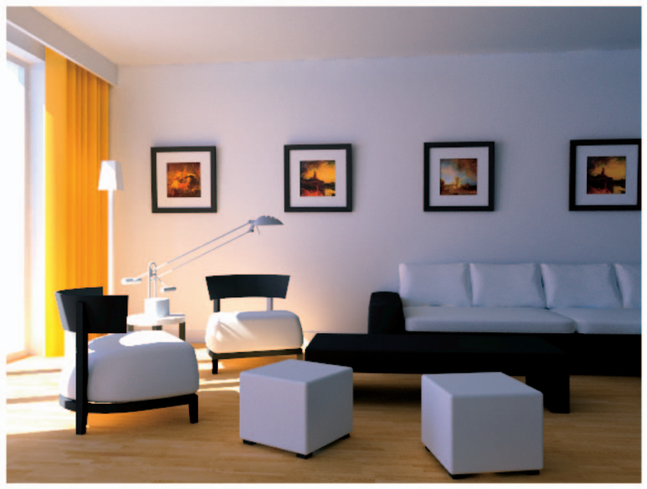
Fig. 17. Sunlit Living Room. Data about this image can be found in Table 1.

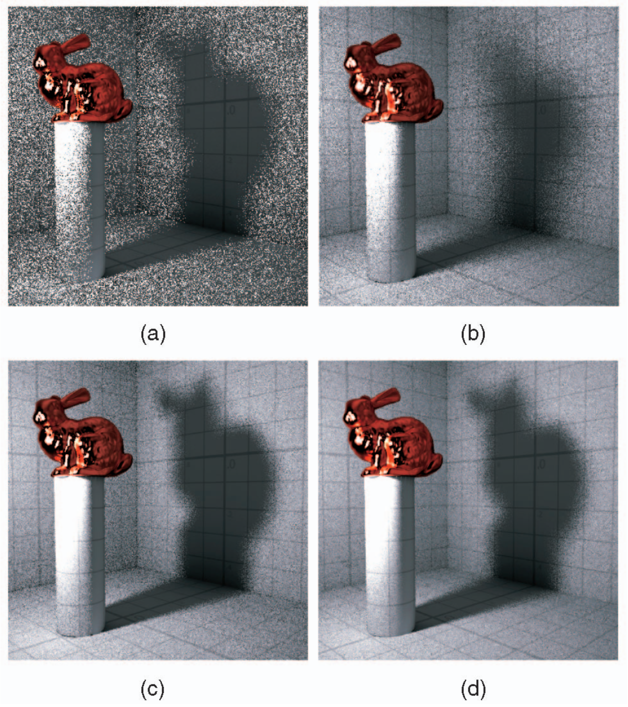

Fig. 18. An example of a condition whereby the assumption that the footprint of the gather ray is unoccluded results in an incorrect image. The bunny is lit by a small patch of light just in front of it. (a) and (c) were rendered using conventional photon mapping. (b) and (d) were rendered using hierarchical photon mapping. (a) and (b) were rendered using 500 FGRs per sample, and (c) and (d) were rendered using 5,000 FGRs per sample. Notice how the shadow from the hierarchical photon map is overly soft in (b), but also notice how the noise is substantially reduced.

below that of the ray footprint (see Fig. 18 for an example). In such cases, our algorithm reduces noise but also introduces bias. Importance sampling, either using the photon map [19], [20] or via an explicit sampling technique would be required to improve the accuracy of the results.

6 Conclusions and Future Work

We have introduced a new method of sampling the photon map for final gathering by integrating over the footprint of each ray. We have shown that this approach can effectively remove noise from scenes with complex, high-frequency illumination and reduce the rendering cost when compared to traditional implementations.

As future work, we would like to explore the possibility of extending the algorithm to work with unbalanced kd-trees [31], as these have shown to be more efficient at locating the nearest photons.

Acknowledgments

The authors would like to thank Branko Jovanovic for his kind donation of the living room scene and Torolf Sauermann at Jotero for his generous donation of the Buddha and Ajax 3D scans. They also wish to thank Marko Dabrovic for use of the Sponza Atrium model and Stanford University for use of the Bunny.

References

[1] H.W. Jensen, "Global Illumination Using Photon Maps," *Proc. Eurographics Workshop Rendering Techniques (Rendering Techniques '96),* pp. 21-30, 1996.
[2] T.J. Purcell, C. Donner, M. Cammarano, H.W. Jensen, and P. Hanrahan, "Photon Mapping on Programmable Graphics Hardware," *Proc. ACM SIGGRAPH/Eurographics Conf. Graphics Hardware (HWWS '03),* pp. 41-50, 2003.
[3] J.T. Kajiya, "The Rendering Equation," *Proc. ACM SIGGRAPH '86,* pp. 143-150, 1986.
[4] V. Havran, J. Bittner, R. Herzog, and H.-P. Seidel, "Ray Maps for Global Illumination," *Proc. Eurographics Symp. Rendering (ESR '05),* pp. 43-54, 2005.
[5] R.F. Tobler and S. Maierhofer, "Improved Illumination Estimation for Photon Maps in Architectural Scenes," *Proc. Int'l Conf. Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG '06),* pp. 257-261, 2006.
[6] H. Hey and W. Purgathofer, "Advanced Radiance Estimation for Photon Map Global Illumination," *Computer Graphics Forum, Proc. Eurographics '02,* vol. 21, no. 3, pp. 541-545, 2002.
[7] P.H. Christensen, "Faster Photon Map Global Illumination," *J. Graphics Tools,* vol. 4, no. 3, pp. 1-10, 1999.
[8] L. Williams, "Pyramidal Parametrics," *Proc. ACM SIGGRAPH '83,* pp. 1-11, 1983.
[9] D. Benson and J. Davis, "Octree Textures," *Proc. ACM SIGGRAPH '02,* pp. 785-790, 2002.
[10] D. DeBry, J. Gibbs, D.D. Petty, and N. Robins, "Painting and Rendering Textures on Unparameterized Models," *Proc. ACM SIGGRAPH '02,* pp. 763-768, 2002.
[11] H.W. Jensen and J. Buhler, "A Rapid Hierarchical Rendering Technique for Translucent Materials," *Proc. ACM SIGGRAPH '05,* p. 12, 2005.
[12] B. Walter, S. Fernandez, A. Arbree, K. Bala, M. Donikian, and D.P. Greenberg, "Lightcuts: A Scalable Approach to Illumination," *Proc. ACM SIGGRAPH '05,* vol. 24, no. 3, pp. 1098-1107, 2005.
[13] P.H. Christensen and D. Batali, "An Irradiance Atlas for Global Illumination in Complex Production Scenes," *Rendering Techniques,* pp. 133-142, 2004.
[14] Y. Yue, K. Iwasaki, Y. Dobashi, and T. Nishita, "Global Illumination for Interactive Lighting Design Using Light Path Pre-Computation and Hierarchical Histogram Estimation," *Proc. 15th Pacific Conf. Computer Graphics and Applications (PG '07),* pp. 87-98, 2007.
[15] A. Keller and I. Wald, "Efficient Importance Sampling Techniques for the Photon Map," *Proc. Vision Modelling and Visualization (VMV '00),* pp. 271-279, 2000.
[16] I. Peter and G. Pietrek, "Importance Driven Construction of Photon Maps," *Proc. Eurographics Rendering Workshop (Rendering Techniques '98),* G. Drettakis and N. Max, eds., pp. 269-280, 1998.
[17] S. Fan, S. Chenney, and Y. chi Lai, "Metropolis Photon Sampling with Optional User Guidance," *Proc. 16th Eurographics Symp. Rendering (Rendering Techniques '05),* pp. 127-138, 2005.

[18] F. Suykens and Y.D. Willems, "Density Control for Photon Maps," *Proc. Eurographics Workshop Rendering (Rendering Techniques '00),* pp. 23-34, 2000.
[19] H.W. Jensen, "Importance Driven Path Tracing Using the Photon Map," *Proc. Sixth Eurographics Workshop Rendering (Rendering Techniques '95),* P.M. Hanrahan and W. Purgathofer, eds., pp. 326-335, 1995.
[20] H. Hey and W. Purgathofer, "Importance Sampling with Hemispherical Particle Footprints," *Proc. 18th Spring Conf. Computer Graphics (SCCG '02),* pp. 107-114, 2002.
[21] T. Tawara, K. Myszkowski, and H.-P. Seidel, "Efficient Rendering of Strong Secondary Lighting in Photon Mapping Algorithm," *Proc. Theory and Practice of Computer Graphics (TPCG '04),* pp. 174-178, 2004.
[22] V. Havran, R. Herzog, and H.-P. Seidel, "Fast Final Gathering via Reverse Photon Mapping," *Computer Graphics Forum,* Proc. Eurographics '05, vol. 24, no. 3, pp. 323-333, 2005.
[23] G.J. Ward, F.M. Rubinstein, and R.D. Clear, "A Ray Tracing Solution for Diffuse Interreflection," *Proc. ACM SIGGRAPH '88,* pp. 85-92, 1988.
[24] G.J. Ward and P. Heckbert, "Irradiance Gradients," *Proc. Third Eurographics Workshop Rendering (Rendering Techniques '92),* pp. 85-98, 1992.
[25] J. Křivánek, P. Gautron, S. Pattanaik, and K. Bouatouch, "Radiance Caching for Efficient Global Illumination Computation," *IEEE Trans. Visualization and Computer Graphics,* vol. 11, no. 5, pp. 550-561, Sept./Oct. 2005.
[26] J. Křivánek, P. Gautron, K. Bouatouch, and S. Pattanaik, "Improved Radiance Gradient Computation," *Proc. 21st Spring Conf. Computer Graphics (SCCG '05),* pp. 155-159, 2005.
[27] F. Suykens, "On Robust Monte Carlo Algorithms for Multi-Pass Global Illumination," PhD dissertation, chapter 8, Katholieke Univ. Leuven, pp. 158-159, 2002.
[28] H. Igehy, "Tracing Ray Differentials," *Proc. ACM SIGGRAPH '99,* pp. 179-186, 1999.
[29] H.W. Jensen, *Realistic Image Synthesis Using Photon Mapping.* A.K. Peters, 2001.
[30] G. Ward, *Graphics Gems II,* James Arvo, ed. pp. 80-83, Academic Press, Real Pixels, 1991.
[31] I. Wald, J. Günther, and P. Slusallek, "Balancing Considered Harmful—Faster Photon Mapping Using the Voxel Volume Heuristic," *Computer Graphics Forum, Proc. Eurographics '04,* vol. 22, no. 3, pp. 595-603, 2004.

Ben Spencer received the BSc degree from Swansea University, UK. He is currently working toward the PhD degree, conducting research into ray tracing, rendering, and global illumination.

Mark W. Jones received the BSc and PhD degrees from Swansea University, UK. He is now a senior lecturer in the Department of Computer Science, Swansea University. His research interests include volume graphics, voxelization, distance fields, ray tracing, and global illumination.