Eurographics Symposium on Rendering 2015
J. Lehtinen and D. Nowrouzezahrai
(Guest Editors)

Volume 34 (2015), Number 4

Wavelet Point-Based Global Illumination

Beibei Wang[+*] Xiangxu Meng[+] Tamy Boubekeur[*]

[*]Telecom ParisTech - CNRS - Institut Mines-Telecom [+]Shandong University

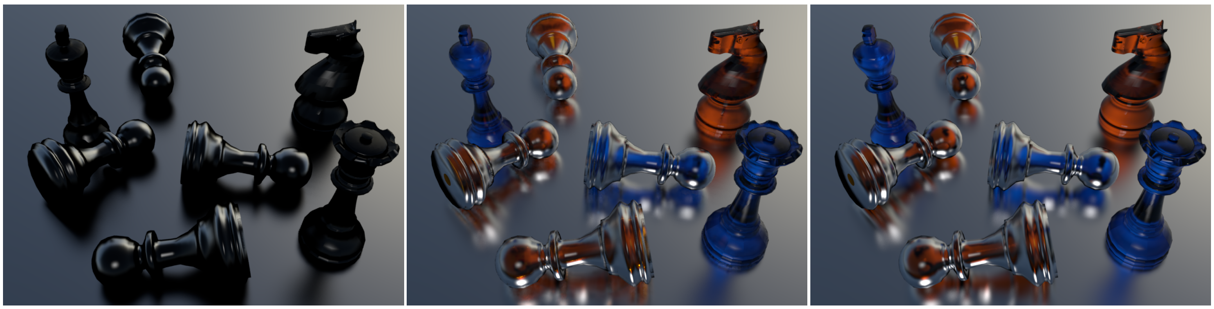

(a) PBGI (b) WPBGI (c) BPT

Figure 1: *Compared to traditional PBGI (a), our approach (b) models accurately non-diffuse indirect lighting effects and appears as an efficient substitute to bidirectional path tracing (c), trading a moderate image degradation for up to a 10x speed-up in our experiments.*

Abstract

Point-Based Global Illumination (PBGI) is a popular rendering method in special effects and motion picture productions. This algorithm provides a diffuse global illumination solution by caching radiance in a mesh-less hierarchical data structure during a pre-process, while solving for visibility over this cache, at rendering time and for each receiver, using microbuffers, which are localized depth and color buffers inspired from real time rendering environments. As a result, noise free ambient occlusion, indirect soft shadows and color bleeding effects are computed efficiently for high resolution image output and in a temporally coherent fashion. We propose an evolution of this method to address the case of non-diffuse inter-reflections and refractions. While the original PBGI algorithm models radiance using spherical harmonics, we propose to use wavelets parameterized on the direction space to better localize the radiance representation in the presence of highly directional reflectance. We also propose a new importance-driven adaptive microbuffer model to capture accurately incoming radiance at a point. Furthermore, we evaluate outgoing radiance using a fast wavelet radiance product and contain the induced larger memory footprint by encoding hierarchically the wavelets in the PBGI tree. As a result, our algorithm can handle non-lambertian BSDF in the light transport simulation, reproducing caustics and multiple reflections/refractions bounces with a similar quality to bidirectional path tracing in a large number of cases and for only a fraction of its computation time. Our approach is simple to implement and easy to integrate into any existing PBGI framework, with an intuitive control on the approximation error. We evaluate it on a collection of example scenes.

1. Introduction

Over the last decade, *global illumination* (GI) has become a standard requirement for almost any industrial computer graphics production, from movie special effects to motion pictures and TV shows. Among the vast repository of rendering algorithms that support (at least a subset of) GI effects, *Point-Based Global Illumination* (PBGI) is certainly one of the most widely used solution. The basic idea of PBGI is to decorrelate the scene complexity from the GI computation by substituting a shaded point cloud to the scene when computing any indirect lighting effect, only keeping the polygonal representation for direct visibility from the camera. This algorithm is particularly efficient at simulating a large number of diffuse effects which are visually appealing in GI, including ambient occlusion, indirect soft shadows and color bleeding. The algorithm itself can be easily adapted to compute multiple light bounces, subsurface scattering and more general volumetric effects while staying fast, embarassingly parallel and finely tunable by artists. However, although final receiver surface samples may be shaded using any BRDF, the indirect lighting effects captured by standard PBGI are restricted to diffuse inter-reflections and do not allow to reproduce caustics for instance, implying alternative GI solutions (e.g., *Monte Carlo Ray Tracing*, *Photon Mapping*) to tackle these cases.

PBGI At Glance The PBGI rendering algorithm [Chr08] can be summarized in two main stages: *radiance caching* and *many-views rasterization*. During the first stage, the virtual scene is sampled into a dense point cloud, each point is shaded from the primary light emitters and a hierarchical data structure (e.g., BVH) is generated to store them, propagating the leaf/point radiance estimates to the inner nodes in a bottom-up fashion. The resulting *PBGI tree* can therefore be seen as an hierarchical "spatial cache" of the scene's radiance. In the second stage, the indirect illumination of each receiver (e.g., unprojected image pixel) is evaluated by localizing a color+depth hemispherical buffer, called *microbuffer* (or MB), at the receiver position and filling it by rasterizing the PBGI tree onto it. This rasterization resembles the hierarchical Z-buffer algorithm, only substituting the PBGI tree to the actual scene and filling a specific MB for each single receiver. The mesh-less nature of the PBGI tree eases the extraction of a receiver-dependent adaptive level-of-detail (LoD) of the scene as a "cut" in the tree, reducing drastically the computational footprint of the many rasterizations required to fill the MBs of all the pixels in a high resolution image. The final receiver radiance (e.g., pixel color) is obtained by convolving the filled MB color component with the receiver reflectance distribution.

Limitations One key feature of PBGI is to reproduce noise-free diffuse GI effects efficiently by modeling radiance in the PBGI tree nodes using spherical harmonics (SH). Unfortunately, even using a larger number of coefficients, SH are not able to capture high frequencies efficiently, which precludes non-diffuse reflections or refractions in our case. Consequently, caustics stemming from metals, plastics, glass and other reflective or refractive materials are not handled with classical PBGI frameworks. Even when ignoring the performance issue induced by the larger number of SH coefficients, ringing artifacts quickly appear and the second step of the algorithm remains flawed: as the receiver color is evaluated by convolving its BRDF with a discretized incoming radiance (i.e., the MB), the case of glossy to nearly specular reflections cannot cope with the typical low resolution of the MB and again, the intrinsic speed of PBGI vanishes when increasing the MB size, causing additional artifacts as well.

Overview In this paper, we introduce *Wavelet PBGI* (or *WPBGI*) to address the problem of non-diffuse indirect lighting effects efficiently (Sec. 3).

First, we propose a new outgoing radiance model which captures directional reflections using Haar wavelets warped over the space of directions (Sec. 3.1). Our model characterizes efficiently, at each node of the PBGI tree, the localized regions on the (hemi)sphere which exhibits high frequency outgoing radiance. In order to drastically shrink the significantly larger memory footprint induced by such an approach, we propose to store the radiance wavelets hierarchically, expressing these nodes attributes w.r.t. to their (average) parent one.

Second, when rasterizing the tree for a particular MB at rendering time, our wavelet hierarchy allows to compute adaptive importance-driven cuts with very fine structures (e.g., specular spots) next to coarser ones. Consequently, we introduce an adaptive MB model which drives the LoD extraction and exhibits a spatially varying resolution based on the classical geometric factors (distance, incidence angle) as well as on appearance parameters, such as the estimated incoming radiance and the BSDF glossiness (Sec. 3.2). We additionally extend this MB model to the full spherical domain to support refractions.

Third, we propose a multi-bounce indirect lighting solution, exploiting the hierarchical structure in a form of generalized rasterization, where the PBGI tree is splatted onto itself to emulate the light propagation (Sec. 3.3). Additionally, we use a fast wavelet product to quickly evaluate outgoing radiance from the adaptive incoming one and the BSDF.

As a result, our improved PBGI algorithm allows to render images with caustics effects and many flavors of indirect non-diffuse effects, up to close to perfectly specular as well as refractive materials , for only a fraction of the cost of a bidirectional path tracing solution, with a similar quality (Sec. 4).

2. Previous Work

PBGI. PBGI was first proposed by Christensen [Chr08] to evaluate diffuse light transport by substituting a mesh-less hierarchy to the scene and using a z-buffer inspired approach to solve for visibility at each receiver/pixel. The notion of point-based substitutes was actually introduced by Bunnell [Bun05] for real time ambient occlusion and indirect illumination. Ritschel et al. [REG*09] proposed warped microbuffer instead of cube ones to allow for importance sampling based on the receiver BRDF, with an efficient GPU implementation. Holländer et al. [HREB11] further improved fine-grained parallelism of the adaptive cut computation. A number of approaches have been proposed to improve the cut computation, including importance-driven point projection based on an initial clustering [MW11], cut picking algorithm for HDR imaging [Tab12], and tree-cut/microbuffers factorization based on spatial coherence [WHB*13]. The PBGI memory issue has been tackled with an out-of-core framework for PBGI, providing a cache-coherent tree construction and traversal [KTO11], and with an in-core solution which quantizes all tree nodes against a small set of representatives, learned on-the-fly [BB12].

Wavelets for Rendering. Wavelets have been extensively used to represent light transport functions [GSCH93,

x, n	Point position and normal
L_o, L_i	Outgoing radiance and incoming radiance
ρ	BRDF
ω_i, ω_o	Local incident, outgoing directions
V	Binary visibility
$\mathcal{T}_2, \mathcal{T}_4$	2D and 4D wavelet transform
U	Coeffs. with 4D wavelet transform of ρ
D	Coeffs. with 2D wavelet transform of L_o
W	Coeffs. with 2D wavelet transform of L_i
L_{oj}, F_j	Original outgoing radiance of point / node j
$\widetilde{L}_{oj}, \widetilde{F}_j$	Reconstructed outgoing radiance of point / node j
Ψ	Haar wavelet basis function
C_j	Coeffs of point j in wavelet expansion of L_{oj}
G_j, H_j	Average and detail coeffs vector of node j
j^-, j^+	Index of the left and the right child of node j
P_j	Path from the root to the parent of node j
s_j	Sign of node j: 1 left, -1 right
m	Prescribed level
$\mathcal{F}, \mathcal{F}_\rho, \mathcal{F}_l$	Total, BRDF and lighting importance function
γ	Radiance threshold
M	Number of pixels of the microbuffer
ω_k	Direction of the kth pixel in the microbuffer

Table 1: *Notations.*

CSSD95, KTHS06], to approximate environment maps [NRH03], to evaluate, through importance sampling, products of complex functions [CJAMJ05], such as measured BRDFs and distant lighting and to model the multiple components of a rendering pipeline, including lighting, reflectance and visibility [NRH04, SM06]. In combination with other techniques [LSSS04], such as clustered principal component analysis for instance, further compression can be achieved. In our work, we use the off-the-shelve Haar basis, which is easy to implement and met our expectations at capturing high frequencies. However, the framework we describe in this paper could be further explored using alternative basis e.g., tuned for the spherical domain [SS95]. Yet, performances may suffer beyond simple basis.

Other Related Work. (Hemi)Spherical harmonics reconstruct low-frequency functions efficiently and have been used to store irradiance environment maps [RH01], transfer and lighting functions [SKS02], incoming [KGPB05] and outgoing radiance [Chr08]. However, for high frequency BRDF or lighting, they require a large number of coefficients and suffer from "ringing" artifacts. Spherical Gaussians have also been widely used to represent light transport functions [TS06, GKMD06, LWDB10] and BRDFs [WRG*09, XSD*13]. Lehtinen et al. [LZT*08] proposed a hierarchical mesh-less basis to represent diffuse light transport. Alternatively, Keller et al. [Kel97] proposed to approximate indirect lighting using many virtual point lights. Walter et al. [WFA*05] improved this approach by organizing them into a binary tree and defining lightcuts, which are similar in essence to the cuts used in PBGI.

3. Wavelet PBGI

In this section, we describe the WPBGI components that differ from the traditional PBGI algorithm and refer to the notations listed in Tab. 1. An overview of the WPBGI work flow is given in Fig. 2. We start by recalling the radiance equation with direct lighting:

$$L_o(x,\omega_o) = \int_{\Omega_{2\pi}} L_i(x,\omega_i)\rho(x,\omega_i,\omega_o)V(x,\omega_i)(\omega_i \cdot n)d\omega_i. \tag{1}$$

3.1. Wavelet Radiance Model

To properly capture $L_o(x,\omega_o)$, the radiance distribution of non-diffuse surfaces, we need a compact representation that can describe high frequency variations efficiently, spending only few numbers to model large, slowly varying regions. Wavelets do have this property. Therefore, we propose to model the radiance distribution of each PBGI sample/tree leaf using Haar wavelets. To do so, at caching time, the distribution is estimated for all samples and expressed as a collection of two-dimensional functions $L_{oj}(\omega)$, where j is the index of the point. Each of these functions is projected onto the Haar basis $\Psi = \{\psi_i\}$ by writing it as a series of expansions:

$$L_{oj}(\omega) = \sum_i c_i \psi_i(\omega),$$

with $C_j = \{c_i\}$ representing the vector of coefficients in the basis. To evaluate the coefficients, $L_{oj}(\omega)$ is sampled in a cube map yielding six small (typically 32×32) images for each point sample. We then Haar-transform each image and store the resulting coefficients in a tree structure similarly to Sun et al. [SM06]: the coefficients of the mother scaling basis function are located at the root and each node contains three detail coefficients and children indices. When the detail coefficients of a node and of all its children fall below a threshold, the node is discarded, which ultimately leads to a sparse representation. This method also allows on-demand point-wise reconstruction (i.e., compressed radiance estimation at the point for a given direction) and avoids decompressing the entire wavelet for each single direction query. This compressed radiance $\widetilde{L}_{oj}(\omega)$ is reconstructed as follows:

$$\widetilde{L}_{oj}(\omega) = C_j \Psi(\omega).$$

Hierarchical Wavelet Coding With our radiance model in hand, we can now build the PBGI tree in a bottom-up fashion, as a complete binary bounding sphere hierarchy, with its leaves being formed by the points of the initial scene sampling and its internal nodes providing economic substitutes for their related sub-trees, including average position/normal, bounding radius and radiance distribution. We observe that Haar wavelets support linear operators and choose earlier to express our wavelet radiance models in the

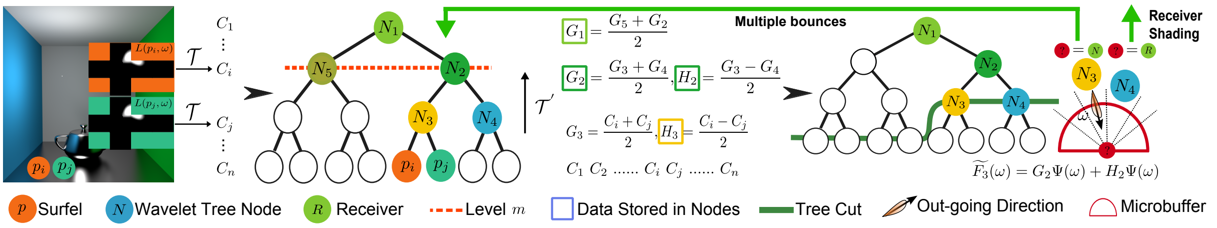


Figure 2: ***Principle***. *At caching time, starting from a dense sampling of the scene, the outgoing radiance distribution at each point p_i is sampled in a cube map and wavelet-transformed ($\mathcal{T}$) to Haar coefficients C_i. The resulting enriched point cloud is further structured in a spatial hierarchy, with the radiance distribution of the internal nodes being propagated bottom-up using a wavelet analysis ($\mathcal{T}'$) on their children coefficients vectors. The tree can then be used either to compute multiple light bounces, by splatting it onto itself with its own nodes as receivers, or at rendering time, to shade any receiver (e.g., unprojected image pixel). In both cases, importance driven cuts are gathered from the tree for each receiver using an adaptive microbuffer.*

global coordinate system. Therefore, the radiance approximation for an internal node j can be obtained directly by averaging its children coefficients:

$$C_j = \frac{C_{j^-} + C_{j^+}}{2}.$$

Taking inspiration from progressive coding schemes, we drastically reduce the memory footprint of our WPBGI tree by encoding the tree *itself* in a wavelet fashion, expressing the radiance wavelet coefficient vector of a node w.r.t. to the coefficient vector of its parent. In our binary tree setup, this boils down to a 1D Haar transform decomposing C_j into an average vector $G_j = (G_{j^-} + G_{j^+})/2$ and a detail vector $H_j = (G_{j^-} - G_{j^+})/2$.

At caching time, we can retain only G_0 (the root average coefficient vector), discard the other average vectors and store only the detail vectors in the nodes, ignoring the ones with a L^2 norm smaller than a user-defined compression threshold (set to 0.002 in our experiments). To avoid storing the entire list of nodes vectors at any intermediate state, we compute this compressed representation during a *post-order depth-first* traversal of the PBGI tree [BB12]. At rendering time, we reconstruct the radiance coefficient vector of a given node j as:

$$C_j := G_{\text{root}} + \sum_{k \in P_j} s_k H_k.$$

Since wavelet compression provides domain-localization, evaluating radiance in a given direction does not require decompressing the full coefficient vectors.

Assuming a full wavelet decomposition of the tree has obvious impact on reconstruction time. This pitfall can be greatly weakened by performing the compression only partially, retaining the average vector G up to a prescribed level m (set to 20 in our experiments). This allows reconstructing the radiance in constant time for any node upper than level m and significantly shrinking the reconstruction time for the other nodes by boot-strapping it directly from the level-m ancestor r of the node:

$$C_j := G_m + \sum_{k \in \{P_j \setminus P_r\}} s_k H_k$$

3.2. Importance Driven Microbuffer

In the PBGI framework, the color response at a particular receiver x and in a particular direction ω_o is evaluated using a microbuffer (MB) of M pixels, which turns Eq. (1) into:

$$L_o(x, \omega_o) = \sum_{k=0}^{M} L_i^k \rho(x, \omega_k, \omega_o) V(x, \omega_k)(\omega_k \cdot n).$$

Note the slight abuse of notation here: L_i^k stands for the radiance, emitted toward x, by a node projecting on the MB pixel k (i.e., ω_k direction). As $V(x, \omega_k)$ is solved by the MB depth component, we ignore this term and retain the simpler form:

$$\widetilde{L_o} = \sum_{k=0}^{M} K L_i^k \rho,$$

where K represents the other terms.

In the context of practical non-diffuse reflections, most of the incident energy is concentrated around a few dominant incoming directions, causing salient light accumulation patterns. To properly capture these caustic effects without resorting to a tremendous number of MB pixels (see Fig. 3.c), we propose *Adaptive MBs* (or AMB) driven by an importance function accounting for both the reflectance distribution at the receiver (see Fig. 3.e) and the incoming lighting (see Fig. 3.g). To cope with the refractions of transparent materials, we implement our AMBs as full spherical color+depth quad trees, considering the tree leaves as AMB pixels. For each receiver, an AMB is initialized uniformly (e.g., 16 × 16) and immediately refined according to the reflectance distribution before being progressively updated based on the incoming lighting during the receiver-dependent tree cut extraction (traversing and splatting). To do so, when the receiver is a final image pixel (i.e., unpro-

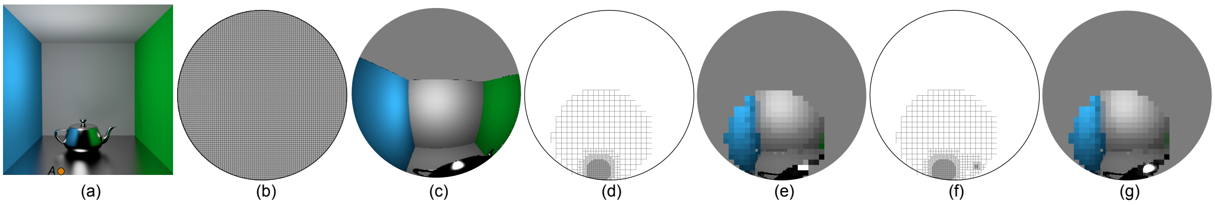


Figure 3: ***Importance driven adaptive microbuffer.*** *(a) A glossy receiver A in the scene (orange dot). (b) Uniform MB structure. (c) Filled uniform MB. (d) Adaptive MB initial structure, based on the BRDF of A only. (e) Corresponding filled AMB. (f) AMB structure refined progressively when gathering the cut. (g) Corresponding filled AMB.*

jected point with fixed outgoing direction), we define the importance function of a MB pixel q as:

$$\mathcal{F}(q) = \mathcal{F}_l(q)\mathcal{F}_\rho(q).$$

We compute $\mathcal{F}_\rho$ similarly to Ritschel et al. [REG*09] but using the subdivision of our quad tree structure instead of warping (see Fig. 3.d). The second term is defined as:

$$\mathcal{F}_l(q) = \begin{cases} 1 & \text{if } \dfrac{L_i(q)}{2^{\Delta_q}} > \gamma, \\ 0 & \text{otherwise,} \end{cases}$$

with Δ_q the refinement level of q (see Fig. 3.f) and the threshold γ typically set to 1. When the receiver is not a final image pixel (i.e., multiple bounce simulation at caching time), we simplify the importance function to: $\mathcal{F}(q) := \mathcal{F}_l(q)$. During the rasterization step, we record all the nodes splatted to each AMB pixel instead of retaining only the nearest one. This guarantees that, when a AMB pixel is subdivided, all the nodes recorded in this pixel will be re-traversed and re-splatted to accurately fill the refined MB pixels (see Fig. 3.g).

3.3. Generalization to Multiple Bounces

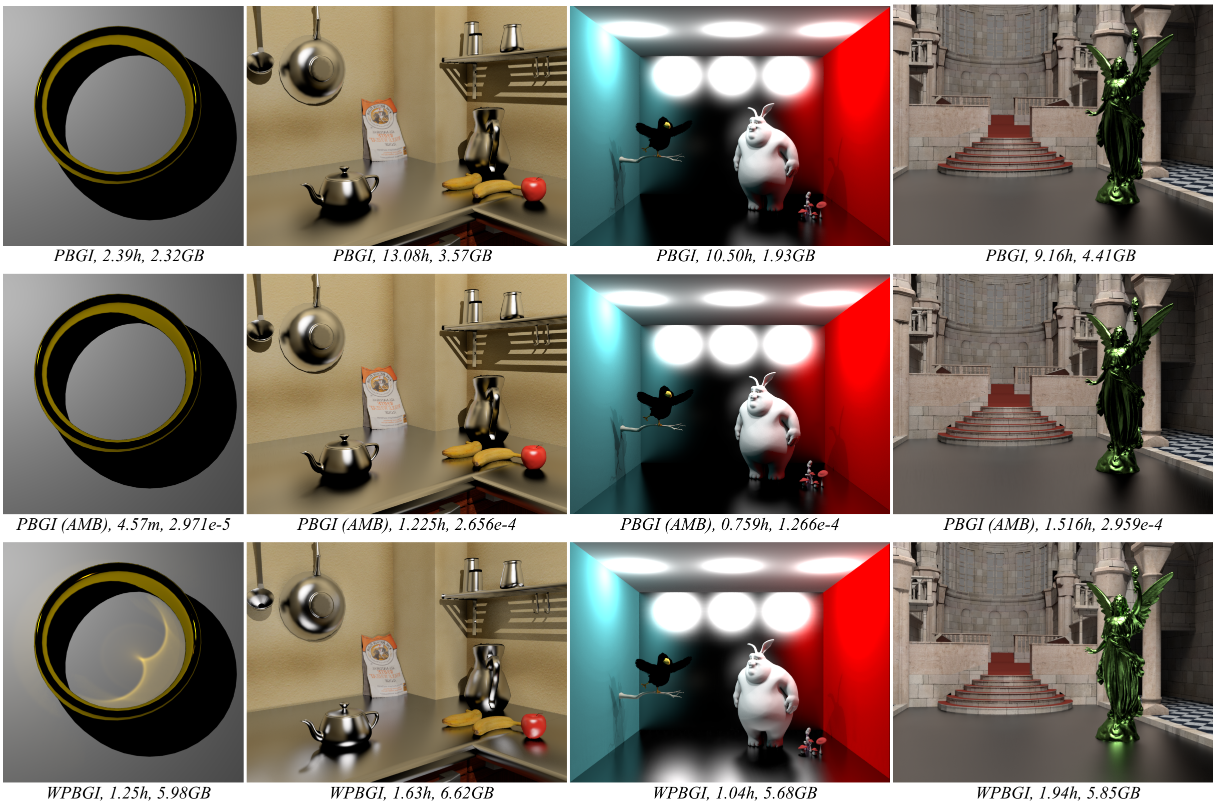
Similarly to the original approach of Christensen [Chr08], we handle multiple indirect light bounces by treating the PBGI leaves (i.e., original point samples) as receivers. For the sake of clarity, let's consider in the following that every node in the PBGI tree can have two distinct roles at the same time: *sender* and *receiver*. Taking inspiration from hierarchical radiosity methods and observing that the traversal cost can be amortized over senders contributing similarly to several receiver leaves, we consider pairs of nodes S/R, with S (resp. R) a node acting as the sender (resp. receiver). To avoid performing a complete cut search for each single receiver leaf, we seek for pairs N_i/N_j such that N_i contributes *similarly* to all the leaves which have N_j for ancestor, so that AMB_{N_j} can be factorized among all of them. To do so, for each pair, we consider the sender-receiver solid angle α_{SR} and the receiver-sender solid angle α_{RS}. Assuming the wavelet radiance coefficients of the PBGI tree have been initialized from direct lighting as described in Sec. 3.1, we summarize our multi-bounce method in Alg. 1. The threshold ε (0.05 in all our experiments) allows to control the accuracy of the multi-bounce propagation and may be increased between each bounce. Note that the test $\alpha_{SR} < \varepsilon$ may be replaced by a more accurate stopping predicate, at the cost of longer computation.

Algorithm 1 One iteration of multi-bounce propagation.

```
Require: $Q$ a queue of node pairs
Require: $T$ an initialized PBGI tree
Require: $\varepsilon$ a user defined threshold
  $Root/Root \to Q$
  while $Q$ not empty do
    $S/R \leftarrow$ head of $Q$
    pop head of $Q$
    if $R$ is a leaf or $\alpha_{RS} < \varepsilon$ then
      if $S$ is a leaf or $\alpha_{SR} < \varepsilon$ then
        splat $S$ in $AMB_R$
      else
        $S^+/R \to Q$
        $S^-/R \to Q$
      end if
    else
      $S/R^+ \to Q$
      $S/R^- \to Q$
    end if
  end while
  for all Leaf $l$ in $T$ do
    if $!AMB_l$ then
      $AMB_l \leftarrow$ closest AMB among ancestors
    end if
    Update $C_l$ from $AMB_l$
  end for
  Update $T$ bottom-up
```

Wavelet Radiance Product. Representing radiance at a point in a wavelet format motivates us to perform the entire radiance estimation in the wavelet domain, which is several times faster than performing the convolution in the spatial domain. Indeed, the reflectance distribution $\rho(\omega_i, \omega_o)$, expressed in local coordinate with the z axis aligned to the normal direction, can be represented by a collection U of 4D wavelet coefficients:

$$U = \mathcal{T}_4\rho(\omega_i, \omega_o).$$



PBGI, 2.39h, 2.32GB | *PBGI, 13.08h, 3.57GB* | *PBGI, 10.50h, 1.93GB* | *PBGI, 9.16h, 4.41GB*

PBGI (AMB), 4.57m, 2.971e-5 | *PBGI (AMB), 1.225h, 2.656e-4* | *PBGI (AMB), 0.759h, 1.266e-4* | *PBGI (AMB), 1.516h, 2.959e-4*

WPBGI, 1.25h, 5.98GB | *WPBGI, 1.63h, 6.62GB* | *WPBGI, 1.04h, 5.68GB* | *WPBGI, 1.94h, 5.85GB*

Figure 4: *Comparison between simple PBGI , PBGI with AMB (with the MSE to PBGI) and WPBGI.*

Similarly, the incoming radiance at a point can be wavelet-transformed and expressed as 2D coefficients:

$$W = \mathcal{T}_2 L_i(\omega_i).$$

Thus, the 2D wavelet transform D of the target outgoing radiance is directly obtained with:

$$D = UW.$$

Consequently, the actual outgoing radiance L_o is retrieved with the inverse transform of D:

$$L_o = \mathcal{T}_2^{-1} D.$$

Note that, since D is expressed in local coordinates, we first decode it from the frequency domain to the spatial domain, and sample it in a cube map which is then, as mentioned earlier, wavelet-transformed. Note also that both U and W are represented as sparse trees [SM06].

4. Results

We implemented our WPBGI algorithm in the Mitsuba Renderer [Jak10]. We use tile rendering and run one thread per-tile. At construction time, the top part of the tree is processed sequentially, until enough subtrees are generated to run multiple threads. Comparisons are performed against (i) a simple PBGI implementation [Chr08] (PBGI) with internal nodes storing a single diffuse radiance value, (ii) PBGI with a large number of SH coefficients to model radiance (SHPBGI), (iii) progressive photon mapping [HOJ08] (PPM), (iv) primary sample space Metropolis Light Transport (MLT) [KSKAC02] and (v) bidirectional path tracing (BPT), which we consider as ground truth. Performances are measured on a 2.67GHz Intel i7 (8 cores) with 9GB of main memory. Images are rendered at a 1024 × 768 pixels resolution (except for the *Ring*, *Cornel Box* and *Sphere* scenes,

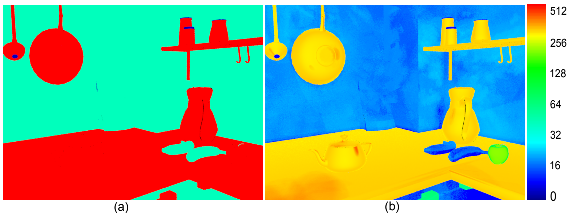
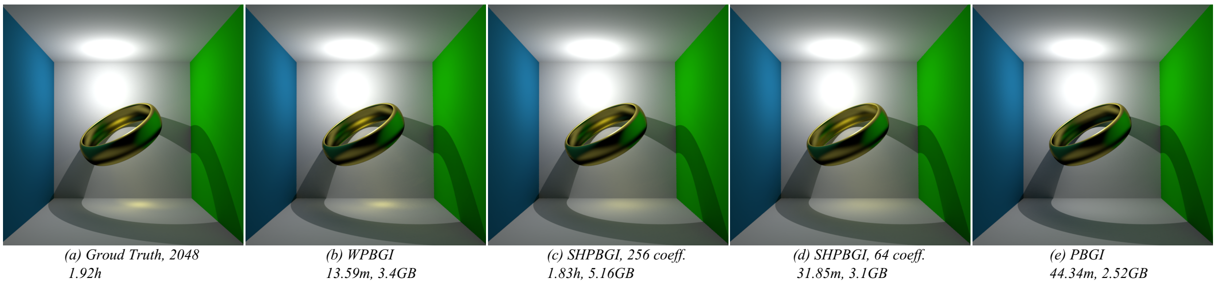


Figure 13: *MB resolution (a) without and (b) with AMB.*


(a) Groud Truth, 2048
1.92h

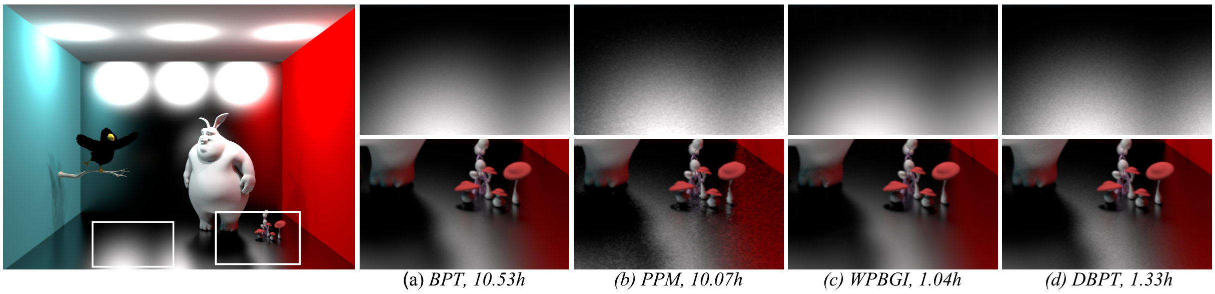
(b) WPBGI
13.59m, 3.4GB

(c) SHPBGI, 256 coeff.
1.83h, 5.16GB

(d) SHPBGI, 64 coeff.
31.85m, 3.1GB

(e) PBGI
44.34m, 2.52GB

Figure 5: *Comparison with spherical harmonics (18 442 tri.). (a) BPT, 2048 samples per pixel. (b) WPBGI, with 6 × 32 × 32 cube map resolution and a discard threshold of 0.002. (c) SHPBGI with 64 × 3 coef. (d) SHPBGI with 256 × 3 coef. for glossy samples and internal nodes. (e) PBGI (without AMB). Total rendering time and peak memory usage are given below each rendering.*


(a) *BPT, 10.53h*

(b) PPM, 10.07h

(c) WPBGI, 1.04h

(d) DBPT, 1.33h

Figure 6: *Bunny scene (217 270 tri.).*

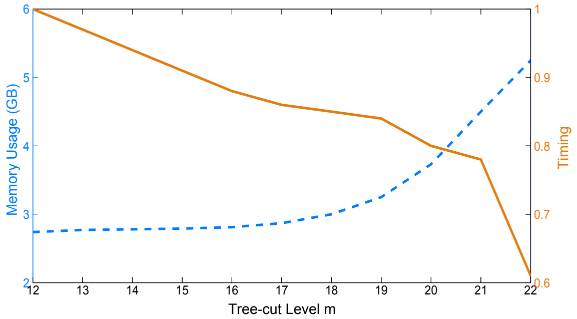


Figure 14: *Memory usage (blue) and rendering time (orange, w.r.t. level-12 timings) according to the bootstrapping value m.*

at 1024 × 1024) with 32 × 32 tiles, using adaptive sampling for anti-aliasing. In all comparisons, we measure numerical differences with the mean squared error (MSE). The BSDFs in all the example scenes are *rough conductors* for glossy reflection materials and *rough dielectric* for refraction materials, which both implement realistic microfacet scattering models [WMLT07].

Comparison to Simple PBGI. We start by comparing to simple PBGI in Fig. 4. As expected, WPBGI produces convincing caustics effects, while PBGI only supports diffuse ones. Surprisingly, beyond the reproduction of these phenomena, the global WPBGI performance also shows a significant speed-up, up to one order of magnitude, thanks to our importance driven AMBs. However, this comes at the price of a larger memory footprint (twice on average). When enabling AMB with PBGI (Fig. 4), we observe a significant performance improvement, for a comparable quality, which underlines the usefulness of AMBs, even without WPBGI.

Comparison to PBGI with Spherical Harmonics. Fig 5 compares WPBGI to a fully featured PBGI implementation, using a large number of SH coefficients at each node to model outgoing radiance. We observe that, even using 3x256 coefficients per node, spending 50% more memory and computing for about 10 times longer, the SHPBGI approach still cannot capture the high frequency non-diffuse features, while our solution comes very close to the BPT solution, still being an order of magnitude faster.

Comparison to Other Methods. In Fig. 6, 7, 8, 9 and 10 we evaluate the accuracy and the visual quality of our approach against BPT and PPM, observing in each case almost invisible differences with BPT, which is typically an order of magnitude slower than WPBGI. We also observe that PPM

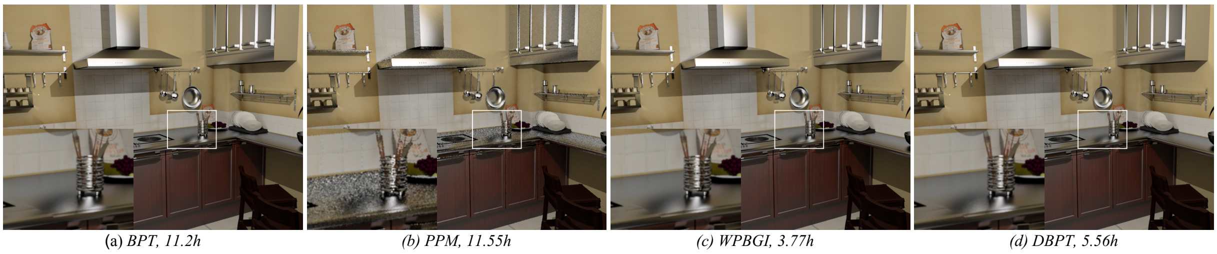

(a) *BPT, 11.2h* (b) *PPM, 11.55h* (c) *WPBGI, 3.77h* (d) *DBPT, 5.56h*

Figure 7: *Kitchen scene (192 116 tri.).*

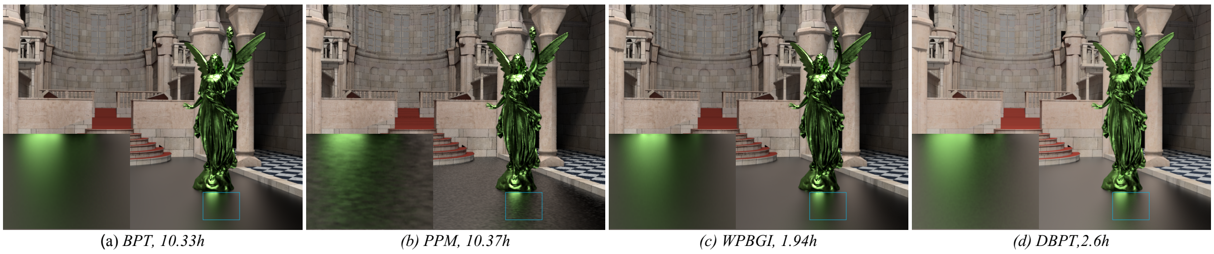

(a) *BPT, 10.33h* (b) *PPM, 10.37h* (c) *WPBGI, 1.94h* (d) *DBPT,2.6h*

Figure 8: *Sibenik scene (167 858 tri.).*

suffers from noise, in particular in the vicinity of shiny reflections, when restricting it to similar rendering time as the ground truth. Additionally, we compare to a *degraded* BPT (DBPT) solution, using less samples to reach similar rendering time as WPBGI; we can easily detect an abundant amount of noise in each case. In Fig. 11, we can observe that our WPBGI is significantly faster than MLT for a similar quality, with an even slightly smoother result. However, we suspect MLT shall provide better results in more complex lighting environment. In Fig. 12, we vary the BRDF glossiness of the object and observe a satisfying behavior of WPBGI up to close to perfectly specular settings, producing images free from visual artifacts, with a preserved speed-up. This excludes the last example, for which the perfectly specular BRDF underlines the limits of WPBGI.

Performance and Timings. In Tab. 2, we report timings and measured errors for the different examples illustrating this paper. The resolutions (ω *res*) used to sample the outgoing radiance prior to wavelet transform correspond to the size of a single face of the cube map and the *memory* value is the peak memory usage during the whole process. The *pre. time* value indicates the point cloud generation and WPBGI tree construction time, while the *render time* value includes the microbuffer initialization, WPBGI tree traversal, nodes splatting and final pixel color evaluation. Here, we can assess the benefits of our approach, with a speed-up ratio for the total time ranging from 2.9 to 10.1 compared to the BPT solution, while the MSE is negligible. Fig. 14 summarizes the memory usage statistics of the WPBGI tree and the timings for the *Corner* scene (5M pts) when varying the bootstrap value *m*. Our hierarchical coding approach shows here its capacity to contain the memory footprint, with an effective speed-up for a moderate memory cost up to level 16. Beyond this value, WPBGI keeps going faster but the memory usage grows significantly, establishing *m* as a natural time-versus-memory control parameter. We also depict the gain of our adaptive scheme by visualizing the actual per-receiver MB resolution when enabling/disabling it in Fig. 13, using a low value for diffuse receivers. The adaptive scheme took 1.63 hours to render while the non-adaptive one took more than 24 hours. Last, WPBGI inherits the temporal coherence of PBGI: we demonstrate this behavior in the accompanying video.

Limitations and Future Work. There are several limitations to our approach that can trigger future work. First, the resolution of the cube map used to sample the outgoing radiance could be made adaptive itself. Second, tailoring the MB refinement based on radiance intensity can lead to unnecessary fine resolutions, with a moderate impact on the final image. Although we experimented with other metrics (e.g., radiance derivative on the MB domain), we did not find a better solution so far and we believe there is a large space from improvement here. Third, our importance driven method does not account for occlusions: a node may be considered important even if it is later occluded during the cut rasterization in the MB. Better predicting occlusion during the importance estimation would reduce the computation time significantly. Last, we use Haar wavelet to represent outgoing radiance, which are easy to implement and to use but requires an intermediate cube map. Ideally, a unified radiance/microbuffer model would provide a more elegant solution to the multiple bounces setting and could avoid the use of an intermediate cube map if designed on top of *spherical wavelets* [SS95].

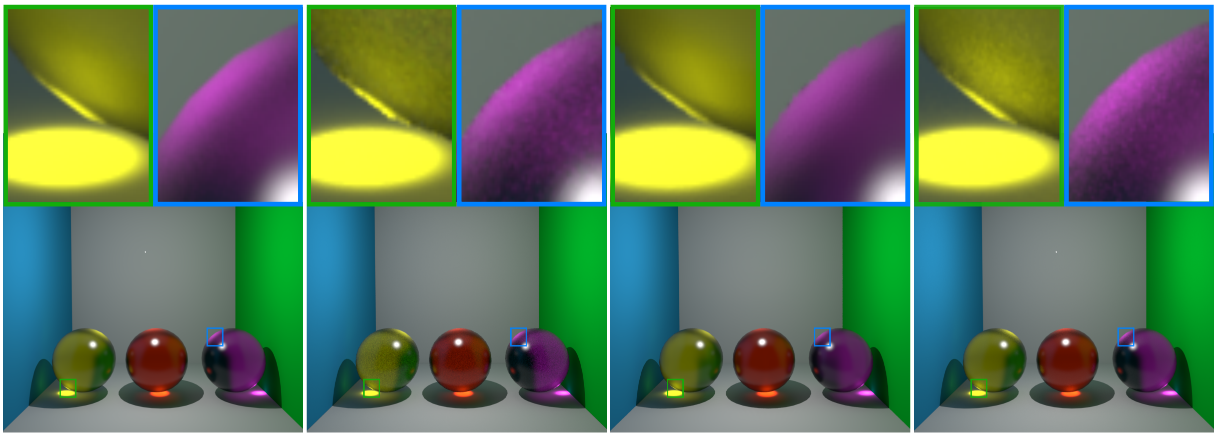

(a) *BPT, 22.80h* *(b) PPM, 22.01h* *(c) WPBGI, 2.29h* *(d) DBPT,1.43h*

Figure 9: *Sphere scene comparison (2 890 tri.).*

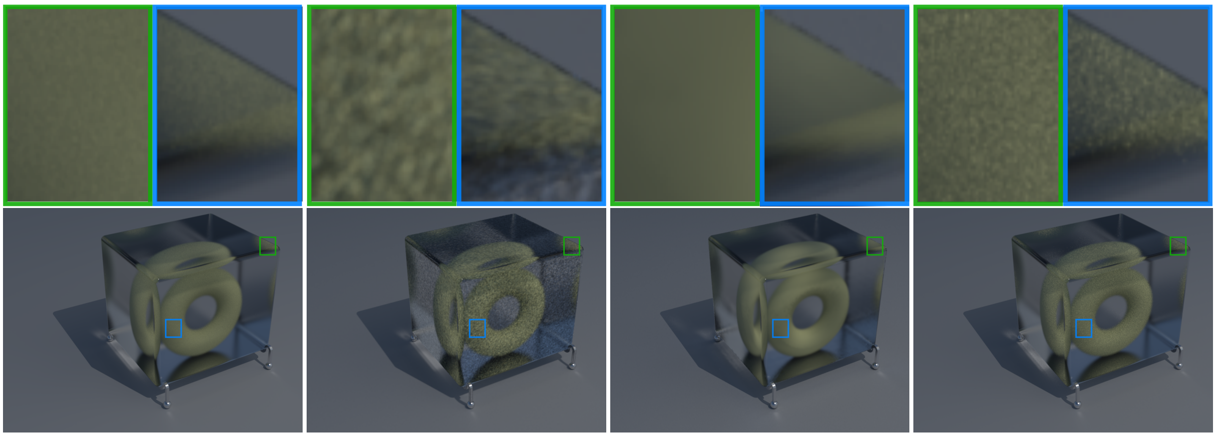

(a) *BPT, 40.01h* *(b) PPM, 20.23h* *(c) WPBGI, 4.37h* *(d) DBPT, 5.17h*

Figure 10: *Torus scene (217 270 tri.), with close-up views at the top.*

5. Conclusion

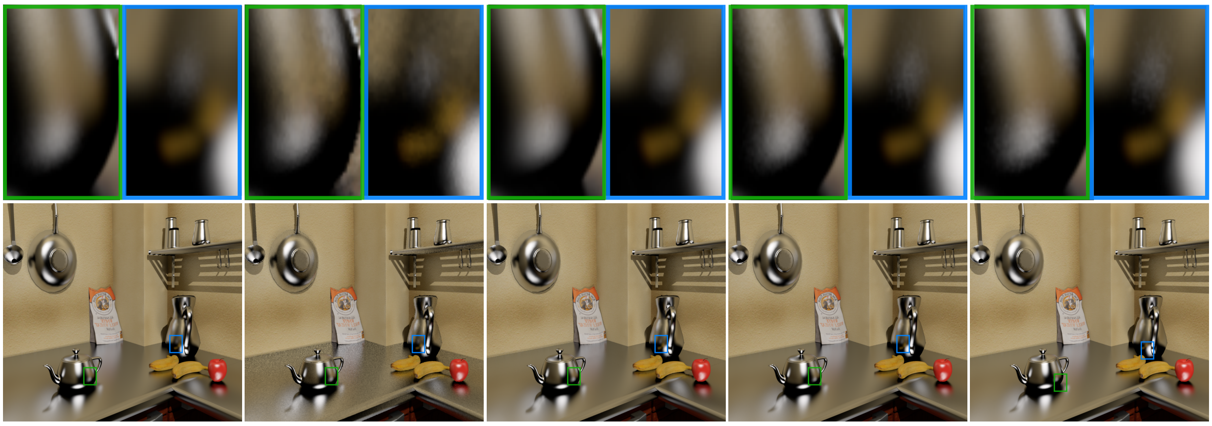
We have proposed *Wavelet PBGI*, an evolution of the PBGI algorithm able to simulate the indirect lighting induced by non-lambertian materials. First, we have introduced a new wavelet radiance model capturing efficiently the high frequency outgoing signal stemming from the point samples and structuring them in a compact, wavelet-inspired tree hierarchy. Second, we solved accurately for visibility over this new multi-resolution radiance cache by introducing an adaptive importance-driven microbuffer model. Third, we described an economic strategy to simulate multiple indirect light bounces, using a fast wavelet radiance product. As a result, our approach competes with bidirectional path tracing, reproducing faithfully the popular physically-based rendering features, such as caustics and multiple bounces, while providing a speed-up ranging from 3x to 10x, without noticeable image degradation. Our approach is easy to implement in any PBGI framework and has a reduced set of intuitive control parameters.

Acknowledgements. We thank the reviewers for their feedback. This work has been partially supported by the European Commission under contracts FP7-323567 HARVEST4D and FP7-287723 REVERIE, by the ANR iSpace&Time project, by the China Scholarship Council, the national natural science foundation of China under grant No.61472224, 61472225, the national high-tech research and development plan of China under grant No.2012AA01A306, the special funding of independent innovation and transformation of achievements in Shandong Province under grant No.2014ZZCX08201 and the special funds of Taishan scholar construction project.

References

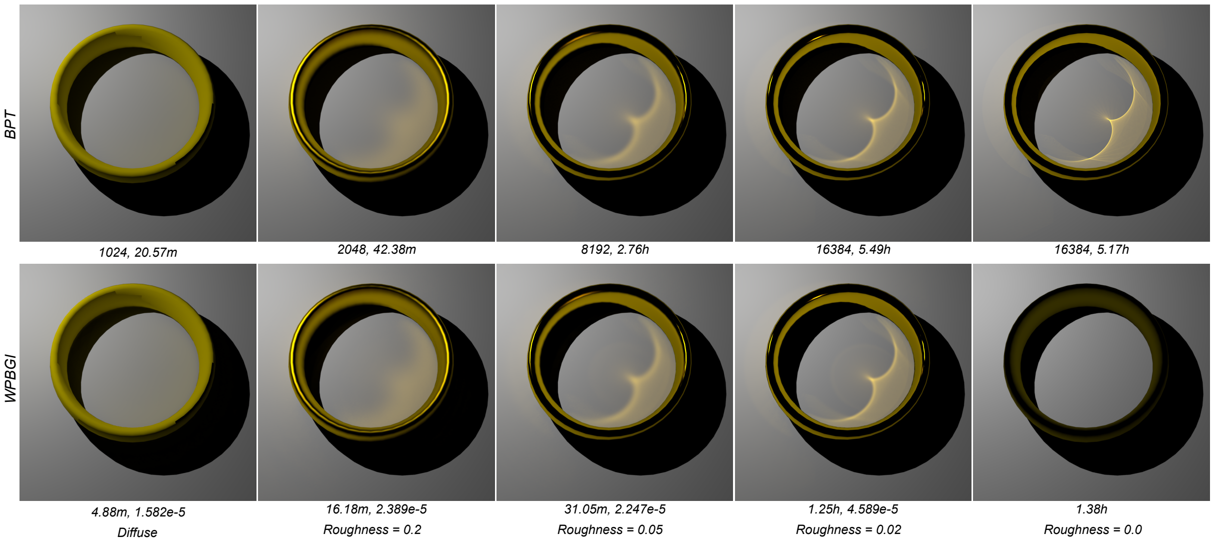
[BB12] BUCHHOLZ B., BOUBEKEUR T.: Quantized point-based global illumination. *Comp. Graph. Forum (Proc. EGSR 2012) 31*, 4 (2012), 1399–1405.

[Bun05] BUNNELL M.: Dynamic ambient occlusion and indirect lighting. *GPU Gems 2* (2005), 223–233.



(a) *BPT, 6.63h* (b) *PPM, 7.79h* (c) *WPBGI, 1.63h* (d) *DBPT,1.68h* (e) *MLT,5.83h*

Figure 11: *Corner scene (47 788 tri.). The close-up views illustrates the benefit of using AMBs.*



1024, 20.57m — 2048, 42.38m — 8192, 2.76h — 16384, 5.49h — 16384, 5.17h

4.88m, 1.582e-5 Diffuse — 16.18m, 2.389e-5 Roughness = 0.2 — 31.05m, 2.247e-5 Roughness = 0.05 — 1.25h, 4.589e-5 Roughness = 0.02 — 1.38h Roughness = 0.0

Figure 12: *Comparison to BPT for low (diffuse) to high frequency (specular) BRDFs (1 154 tri., $\gamma = 0.1$). The number of samples and the total rendering time are given below the top (BPT) images. The total rendering time and the MSE error with the reference are given below the bottom (WPBGI) images. The roughness value corresponds to the glossiness index a rough conductor BRDF.*

[Chr08] CHRISTENSEN P.: *Point-based approximate color bleeding*. Tech. Rep. 08-01, Pixar Technical Notes, 2008.

[CJAMJ05] CLARBERG P., JAROSZ W., AKENINE-MÖLLER T., JENSEN H. W.: Wavelet importance sampling: Efficiently evaluating products of complex functions. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 24*, 3 (2005), 1166–1175.

[CSSD95] CHRISTENSEN P., STOLLNITZ E., SALESIN D., DEROSE T.: Wavelet radiance. In *Photorealistic Rendering Techniques*. Springer, 1995, pp. 295–309.

[GKMD06] GREEN P., KAUTZ J., MATUSIK W., DURAND F.: View-dependent precomputed light transport using nonlinear gaussian function approximations. In *SI3D '06* (2006), pp. 7–14.

[GSCH93] GORTLER S. J., SCHRÖDER P., COHEN M. F., HANRAHAN P.: Wavelet radiosity. In *Proc. SIGGRAPH* (1993), pp. 221–230.

[HOJ08] HACHISUKA T., OGAKI S., JENSEN H. W.: Progressive photon mapping. *ACM Trans. Graph. 27*, 5 (2008), 130:1–130:8.

[HREB11] HOLLÄNDER M., RITSCHEL T., EISEMANN E., BOUBEKEUR T.: Manylods: Parallel many-view level-of-detail selection for real-time global illumination. *Comp. Graph. Forum (Proc. EGSR 2011) 30*, 4 (2011), 1233–1240.

[Jak10] JAKOB W.: Mitsuba renderer. http://www.mitsuba-renderer.org/, 2010.

[Kel97] KELLER A.: Instant radiosity. In *ACM SIGGRAPH 1997* (1997), pp. 49–56.

[KGPB05] KRIVANEK J., GAUTRON P., PATTANAIK S., BOUA-

		BPT		WPBGI						Error
scene	max path.	num. samples	total (h)	pts. (M)	ω res	memory (GB)	pr. time (m)	render time (h)	total time (h)	*MSE*
Ring	3	16384	5.49	5	128^2	5.98	3.22	1.14	1.20	4.589e-5
Corner	3	16384	6.63	8	32^2	6.62	3.05	1.57	1.63	1.570e-4
Bunny	3	16384	10.53	4	32^2	5.68	12.28	0.84	1.04	6.527e-5
Kitchen	3	16384	11.20	5	32^2	5.26	6.57	3.66	3.77	7.948e-5
Sibenik	3	16384	10.33	10	32^2	5.85	11.07	1.75	1.94	2.099e-4
Sphere	5	16384	22.80	2	32^2	5.05	78.56	0.98	2.29	1.370e-4
Torus	5	65536	40.01	1	32^2	6.53	201.61	1.01	4.37	3.116e-4

Table 2: *Performance measures.*

TOUCH K.: Radiance caching for efficient global illumination computation. *IEEE TVCG 11*, 5 (2005), 550–561.

[KSKAC02] KELEMEN C., SZIRMAY-KALOS L., ANTAL G., CSONKA F.: A simple and robust mutation strategy for the metropolis light transport algorithm. *Comput. Graph. Forum 21*, 3 (2002), 531–540.

[KTHS06] KONTKANEN J., TURQUIN E., HOLZSCHUCH N., SILLION F.: Wavelet radiance transport for interactive indirect lighting. In *Proc. EGSR* (2006).

[KTO11] KONTKANEN J., TABELLION E., OVERBECK R. S.: Coherent out-of-core point-based global illumination. In *Comp. Graph. Forum (Proc. EGSR 2011)* (2011), pp. 1353–1360.

[LSSS04] LIU X., SLOAN P.-P., SHUM H.-Y., SNYDER J.: All-Frequency Precomputed Radiance Transfer for Glossy Objects. In *Proc. Eurographics Workshop on Rendering* (2004), pp. 337–344.

[LWDB10] LAURIJSSEN J., WANG R., DUTRÉ P., BROWN B. J.: Fast estimation and rendering of indirect highlights. In *Proc. EGSR* (2010), pp. 1305–1313.

[LZT*08] LEHTINEN J., ZWICKER M., TURQUIN E., KONTKANEN J., DURAND F., SILLION F. X., AILA T.: A meshless hierarchical representation for light transport. *ACM Trans. Graph. 27*, 3 (2008), 37:1–37:9.

[MW11] MALETZ D., WANG R.: Importance point projection for GPU-based final gathering. *Comp. Graph. Forum 30*, 4 (2011), 1327–1336.

[NRH03] NG R., RAMAMOORTHI R., HANRAHAN P.: All-frequency shadows using non-linear wavelet lighting approximation. *ACM Trans. Graph. 22*, 3 (2003), 376–381.

[NRH04] NG R., RAMAMOORTHI R., HANRAHAN P.: Triple product wavelet integrals for all-frequency relighting. *ACM Trans. Graph. 23*, 3 (2004), 477–487.

[REG*09] RITSCHEL T., ENGELHARDT T., GROSCH T., SEIDEL H.-P., KAUTZ J., DACHSBACHER C.: Micro-rendering for scalable, parallel final gathering. *ACM Trans. Graph. (Proc. SIGGRAPH Asia 2009) 28*, 5 (2009).

[RH01] RAMAMOORTHI R., HANRAHAN P.: An efficient representation for irradiance environment maps. In *Proc. SIGGRAPH* (2001), pp. 497–500.

[SKS02] SLOAN P.-P., KAUTZ J., SNYDER J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21*, 3 (2002), 527–536.

[SM06] SUN W., MUKHERJEE A.: Generalized wavelet product integral for rendering dynamic glossy objects. *ACM Trans. Graph. 25*, 3 (2006), 955–966.

[SS95] SCHRÖDER P., SWELDENS W.: Spherical wavelets: Efficiently representing functions on the sphere. In *Proc. SIGGRAPH* (1995), pp. 161–172.

[Tab12] TABELLION E.: Point-based global illumination directional importance mapping. In *ACM SIGGRAPH Talk* (2012).

[TS06] TSAI Y.-T., SHIH Z.-C.: All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Trans. Graph. 25*, 3 (2006), 967–976.

[WFA*05] WALTER B., FERNANDEZ S., ARBREE A., BALA K., DONIKIAN M., GREENBERG D. P.: Lightcuts: A scalable approach to illumination. *ACM Trans. Graph. 24*, 3 (2005), 1098–1107.

[WHB*13] WANG B., HUANG J., BUCHHOLZ B., MENG X., BOUBEKEUR T.: Factorized point-based global illumination. *Computer Graphics Forum 32*, 4 (2013), 117–123.

[WMLT07] WALTER B., MARSCHNER S. R., LI H., TORRANCE K. E.: Microfacet models for refraction through rough surfaces. In *Proc. EGSR* (2007), pp. 195–206.

[WRG*09] WANG J., REN P., GONG M., SNYDER J., GUO B.: All-frequency rendering of dynamic, spatially-varying reflectance. *ACM Trans. Graph. 28*, 5 (Dec. 2009), 133:1–133:10.

[XSD*13] XU K., SUN W.-L., DONG Z., ZHAO D.-Y., WU R.-D., HU S.-M.: Anisotropic spherical gaussians. *ACM Transactions on Graphics 32*, 6 (2013), 209:1–209:11.