Hierarchical Russian Roulette for Vertex Connections

YUSUKE TOKUYOSHI, SQUARE ENIX CO., LTD., Japan
TAKAHIRO HARADA, Advanced Micro Devices, Inc., USA

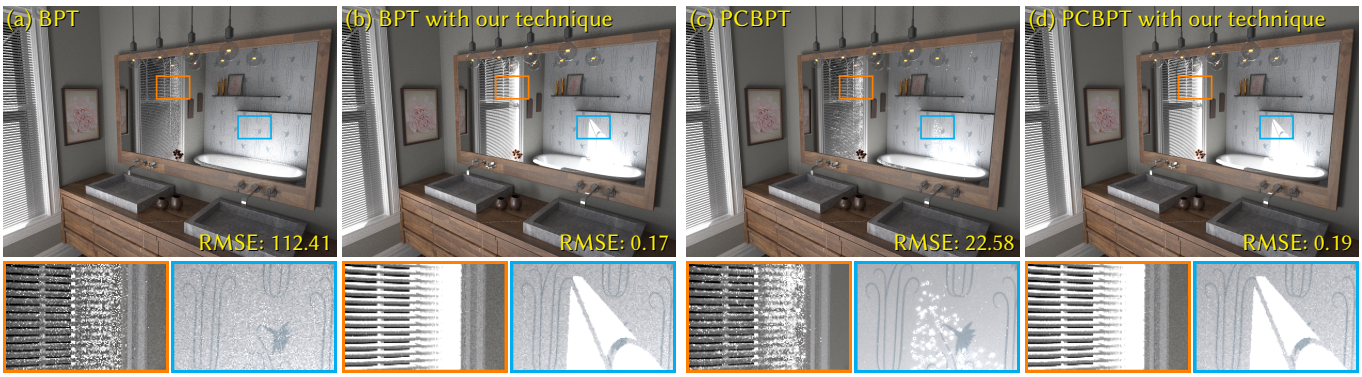


Fig. 1. Equal-time (15 min) comparison of rendering with and without our connection technique (1600×1200 screen resolution). This Bathroom scene is lit by caustics, and the BRDF of the mirror is extremely glossy (GGX roughness: 0.0001). BPT (a) and probabilistic connection BPT (PCBPT) (c) produce intensive variance for specular-diffuse-glossy and glossy-diffuse-glossy reflections shown in closeups. Our technique reduces this variance significantly (b, d).

While bidirectional path tracing is a well-established light transport algorithm, many samples are required to obtain high-quality results for specular-diffuse-glossy or glossy-diffuse-glossy reflections especially when they are highly glossy. To improve the efficiency for such light path configurations, we propose a *hierarchical Russian roulette* technique for vertex connections. Our technique accelerates a huge number of Russian roulette operations according to an approximate scattering lobe at an eye-subpath vertex for many cached light-subpath vertices. Our method dramatically reduces the number of random number generations needed for Russian roulette by introducing a hierarchical rejection algorithm which assigns random numbers in a top-down fashion. To efficiently reject light vertices in each hierarchy, we also introduce an efficient approximation of anisotropic scattering lobes used for the probability of Russian roulette. Our technique is easy to integrate into some existing bidirectional path tracing-based algorithms which cache light-subpath vertices (e.g., probabilistic connections, and vertex connection and merging). In addition, unlike existing many-light methods, our method does not restrict multiple importance sampling strategies thanks to the simplicity of Russian roulette. Although the proposed technique does not support perfectly specular surfaces, it significantly improves the efficiency for caustics reflected on extremely glossy surfaces in an unbiased fashion.

CCS Concepts: • **Computing methodologies → Ray tracing**.

Additional Key Words and Phrases: global illumination, bidirectional path tracing, vertex connection, Russian roulette, light culling

Authors' addresses: Yusuke Tokuyoshi, SQUARE ENIX CO., LTD., 6-27-30 Shinjuku, Shinjuku-ku, Tokyo, 160-8430, Japan, yusuke.tokuyoshi@gmail.com; Takahiro Harada, Advanced Micro Devices, Inc. USA, Takahiro.Harada@amd.com.

0730-0301/2019/7-ART36 $15.00
https://doi.org/10.1145/3306346.3323018

ACM Reference Format:
Yusuke Tokuyoshi and Takahiro Harada. 2019. Hierarchical Russian Roulette for Vertex Connections. *ACM Trans. Graph.* 38, 4, Article 36 (July 2019), 12 pages. https://doi.org/10.1145/3306346.3323018

1 INTRODUCTION

Monte Carlo light transport simulation is widely used for photorealistic rendering nowadays, however developing a robust algorithm for various scenes is still a challenging problem. Bidirectional path tracing (BPT) [Lafortune and Willems 1993; Veach and Guibas 1994] is a well-established light transport algorithm which constructs various paths by connecting subpaths traced from a light source and eye (i.e., *light subpath* and *eye subpath*). By using multiple importance sampling (MIS) [Veach and Guibas 1995], the combination of such path sampling techniques reduces the estimation error significantly. However, BPT produces high variance for specular-diffuse-glossy (SDG) or glossy-diffuse-glossy (GDG) paths (Fig. 1a) because of a lack of sampling techniques suitable for these path configurations. Hence, a large number of samples are necessary to render high-quality images for highly glossy surfaces. Probabilistic connections [Popov et al. 2015] increase the sample count by reusing hundreds of light subpaths, however it is still insufficient and produces splotch-like artifacts due to correlated variance (Fig. 1c). This paper tackles the problem of reusing millions of light subpaths for BPT.

To improve the efficiency of connections in BPT, we propose an acceleration technique for *Russian roulette* operations [Arvo and Kirk 1990] according to an approximate scattering lobe at an eye vertex (i.e., vertex of an eye subpath). Russian roulette was used to reject unimportant connections to reduce the number of shadow rays [Veach 1998]. However, this stochastic rejection was still expensive when many light subpaths were cached and reused for every eye vertex to increase the number of path samples. This is because many light vertices (i.e., vertices of light subpaths) are candidates to

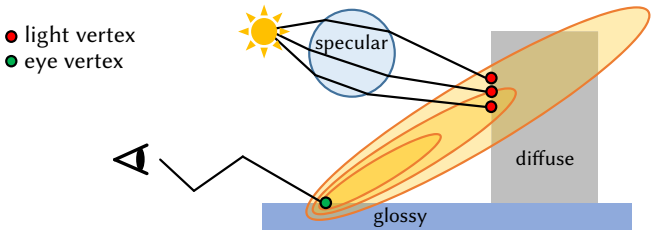


Fig. 2. Our hierarchical Russian roulette finds light vertices within the stochastic scattering range whose shape is determined using the BRDF at an eye vertex. The size of the range is randomly changed for each pair of light and eye vertices.

connect to an eye vertex, and Russian roulette is performed for each candidate. Our approach accelerates this huge number of Russian roulette operations by using a hierarchical rejection algorithm.

Our method is based on stochastic light culling [Tokuyoshi and Harada 2016, 2017] which randomly determines the range of influence from each light source on the basis of Russian roulette. We extend this idea to vertex connections performed at each eye vertex by using the range from the eye vertex (Fig. 2). One critical problem of stochastic light culling was correlation of variance. While Russian roulette for vertex connections generally uses a different random number for each pair of eye and light vertices, stochastic light culling generates a single random number for each light source (or light vertex) in preprocessing. Thus, the correlation of variance between eye vertices was produced. To avoid the correlation, we introduce a hierarchical Russian roulette algorithm which generates random numbers on-the-fly in a top-down fashion. Using this hierarchical algorithm, we are able to assign a different random number for each pair of eye and light vertices in a rapid way. For efficient rejection of light vertices, we also introduce a *squared ellipsoidal lobe* (SEL) approximation for anisotropic glossy reflections, which represents the range from the eye vertex with a tight ellipsoid.

Our method scales to over millions of light subpaths. In addition, simple Russian roulette trivially gives the exact probability density function (PDF) for an arbitrary path unlike probabilistic connections or many-light methods such as lightcuts [Walter et al. 2005]. Thus, our method is easy to combine with other sampling techniques by using MIS in a straightforward manner. Although our Russian roulette probability ignores the visibility and BRDF at each light vertex unlike probabilistic connections, the use of millions of light subpaths is more suitable for highly glossy reflections than a few hundreds of light subpaths used in probabilistic connections. Combining our technique with BPT results in a fast reduction of the error for SDG or GDG reflections (Fig. 1b and Fig. 1d). Our technique is easy to integrate into existing BPT-based methods which cache light vertices, such as light vertex cache BPT [Davidovič et al. 2014], vertex connection and merging [Georgiev et al. 2012a] (a.k.a., unified path space sampling [Hachisuka et al. 2012]), as well as probabilistic connection BPT. Although our technique works only for rough specular reflections, it is efficient for caustics reflected on extremely glossy surfaces.

Our contributions are as follows.

- To accelerate a huge number of Russian roulette operations for vertex connections, we propose a hierarchical rejection algorithm using a stochastic scattering range from an eye vertex.
- For this hierarchical rejection, we introduce a top-down random number assignment method to decorrelate variance.
- We also introduce an SEL approximation to represent the scattering range with a tight ellipsoid for anisotropic glossy reflections.

2 RELATED WORK

Bidirectional Path Tracing (BPT). Classic BPT [Lafortune and Willems 1993; Veach and Guibas 1994] connects subpaths deterministically. Davidovič et al. [2014] cached light vertices during light subpath tracing, and then, for vertex connection at each eye vertex, they randomly sampled a fixed number of light vertices from the cache to avoid the GPU divergence. We employ this caching approach, and apply Russian roulette for all the cached light vertices instead of sampling a fixed number of connections. Popov et al. [2015] proposed importance sampling of connections by reusing hundreds of light subpaths on the basis of an importance cache method [Georgiev et al. 2012b]. Popov et al. also introduced a conservative weighting strategy of multiple importance sampling (MIS) for correlated paths. This probabilistic connection technique is efficient, but the reuse of hundreds of light subpaths is still insufficient for extremely glossy surfaces. Combination with our technique reusing millions of light subpaths alleviates this limitation. Guided BPT [Vorba et al. 2014] improved the efficiency of subpath sampling via online learning using parametric mixture models [Herholz et al. 2018, 2016]. Since our method performs for connections between generated subpaths, it can also be combined with such a subpath generation technique to further improve the robustness. Vertex connection and merging (VCM) is the combination of BPT and progressive photon mapping [Hachisuka et al. 2008; Knaus and Zwicker 2011] using MIS. We evaluate the efficiency of VCM using probabilistic connections and our connection technique in Sect. 6.

Many-Light Methods. Efficient connections between eye vertices and many light vertices have been studied in many-light methods [Dachsbacher et al. 2014] such as lightcuts [Walter et al. 2005]. For indirect illumination, many-light methods represent cached light vertices as virtual point lights (VPLs) [Keller 1997]. Although these methods are efficient, they are difficult to combine with BPT using powerful PDF-based MIS strategies such as the *balance heuristic* and *power heuristic* [Veach and Guibas 1995]. This is because their PDF cannot be obtained for paths sampled by path tracing. Therefore, non-PDF-based heuristics [Kollig and Keller 2006; Walter et al. 2012] have been used for combining with path tracing. Estevez and Kulla [2018] proposed importance sampling of many lights for direct illumination to use PDF-based MIS for area lights. Although this importance sampling can also be used for sampling VPLs, it is also difficult to use the above PDF-based MIS heuristics for indirect illumination. Unlike these many-light methods, our method does not restrict MIS strategies for BPT, since it accelerates simple Russian roulette operations. Another issue in many-light methods is correlation of variance, which induces banding artifacts. To reduce

Table 1. Notations used in this paper.

Symbol	Domain	Description
$\mathbf{y}$	$\mathbb{R}^3$	Vertex in a light subpath
$\mathbf{z}$	$\mathbb{R}^3$	Vertex in an eye subpath
$P(\mathbf{y}, \mathbf{z})$	$[0, 1]$	Probability of Russian roulette
$\mathbf{n}$	S^2	Surface normal
$\boldsymbol{\omega}$	S^2	A direction on a scattering lobe
$\boldsymbol{\omega}'$	S^2	Incident direction
$\rho(\mathbf{z}, \boldsymbol{\omega}', \boldsymbol{\omega})$	$[0, \infty]$	BRDF at $\mathbf{z}$
$q_{\mathbf{z}}(\boldsymbol{\omega})$	$[0, \infty]$	Approximated scattering lobe at $\mathbf{z}$
ξ_i	$[0, 1)$	ith uniform random number
$R(\boldsymbol{\omega}; \mathbf{z}, \xi_i)$	$[0, \infty]$	Stochastic scattering range from $\mathbf{z}$
$D(\cdot)$	$[0, \infty]$	GGX distribution
$K(\cdot)$	$[0, 1]$	Squared ellipsoidal lobe (SEL) function
α_x, α_y	$[0, 1]$	GGX roughness
$\acute{\alpha}_x, \acute{\alpha}_y$	$[0, 1]$	SEL roughness
M	$\mathbb{N}$	Number of light subpaths

this correlation for lightcuts, Walter et al. [2006] pregenerated and stored several random numbers into each node of their light tree, and then selected which set of pregenerated random numbers to use for each eye vertex. Walter et al.'s approach reduces the correlation, but cannot completely avoid it. Our method generates random numbers on-the-fly, and thus completely eliminates the variance correlation without increasing the memory usage. On the other hand, the proposed method is restricted to partially stratified sampling, while lightcuts can use fully stratified sampling.

Light Culling. Light culling is an acceleration technique for many lights [Olsson et al. 2014], and frequently used in recent video games. This technique performs shading only for directly visible eye vertices inside a predetermined range from each light source. For real-time rendering, fast GPU-based culling algorithms specialized for directly visible eye vertices (e.g., tiled culling [Stewart 2015]) have been developed, however the use of fixed light ranges was a biased and inconsistent estimator. For unbiased culling, Tokuyoshi and Harada [2016] introduced a stochastic light culling method which randomly determines the light range based on Russian roulette using a precomputed single random number for each light. To apply light culling to glossy VPLs, Dachsbacher and Stamminger [2006] derived a tighter bounding spheroid for the isosurface of the reflected radiance for the Phong BRDF model [1975]. Tokuyoshi and Harada [2017] extended this to compute the bounding spheroid for the GGX microfacet BRDF [Walter et al. 2007]. Thus, glossy reflective caustics can now be sampled for real-time rendering by culling VPLs in an unbiased fashion, though it is limited only to directly visible eye vertices. For indirect eye vertices in path tracing, a bounding sphere tree of light ranges [Tokuyoshi and Harada 2016] was used for direct illumination of many lights by assuming low-frequency directional distributions of light. Since correlation of variance was produced owing to the precomputed random numbers, that light-range tree had to be updated by generating new random numbers iteratively to reduce correlation artifacts. We propose a new method which avoids the correlation of variance as well as supporting high-frequency reflections for BPT. This paper builds upon Tokuyoshi and Harada's previous work [2018] that discusses BPT using stochastic light culling.

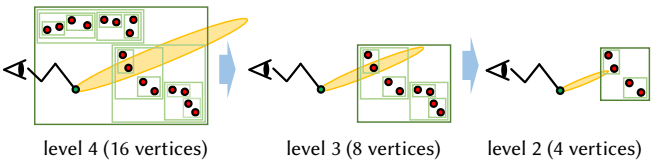


Fig. 3. Hierarchical culling using BVH. For a conservative intersection test between the scattering range and each BVH node, the largest range is used at the node. Descending the hierarchy level, this range shrinks stochastically according to the number of light vertices contained by the node.

3 HIERARCHICAL RUSSIAN ROULETTE

In our method, Russian roulette is performed for all the cached light vertices taking the BRDF at a given eye vertex into account. This Russian roulette is accelerated by range-based culling, thus a few light vertices are efficiently sampled from many light vertices. While existing light culling methods use the range from a light vertex, we use the range from an eye vertex to perform light vertex culling at each vertex in eye subpath tracing. This section introduces an efficient hierarchical algorithm for Russian roulette.

3.1 Stochastic Scattering Range

For the connection between a light vertex $\mathbf{y}$ and eye vertex $\mathbf{z}$, we use the following acceptance probability for Russian roulette:

$$P(\mathbf{y}, \mathbf{z}) = \min\left(\frac{C q_{\mathbf{z}}\left(\overrightarrow{\mathbf{zy}}\right)}{\|\mathbf{y} - \mathbf{z}\|^2}, 1\right), \tag{1}$$

where $C \in [0, \infty)$ is a user-specified parameter to control the tradeoff between variance and performance, and the spherical function $q_{\mathbf{z}}(\cdot)$ is approximately equal to the scattering lobe:

$$q_{\mathbf{z}}(\boldsymbol{\omega}) \approx \rho\left(\mathbf{z}, \boldsymbol{\omega}', \boldsymbol{\omega}\right) |\boldsymbol{\omega} \cdot \mathbf{n}|. \tag{2}$$

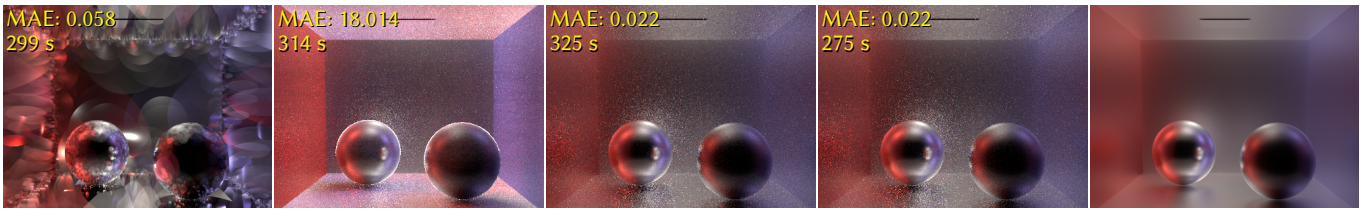
For notations, please see Table 1. To accelerate this Russian roulette for all the light vertices in the cache, rejected light vertices are culled by using the range of acceptance in world space. Let ξ_i be the ith uniform random number, then the bound of the world-space acceptance range from $\mathbf{z}$ is a spherical function and given by the distance $\|\mathbf{y} - \mathbf{z}\|$ satisfying $P(\mathbf{y}, \mathbf{z}) = \xi_i$ as follows:

$$R(\boldsymbol{\omega}; \mathbf{z}, \xi_i) = \sqrt{\frac{C q_{\mathbf{z}}(\boldsymbol{\omega})}{\xi_i}}.$$

Thus, only light vertices enclosed by $R(\boldsymbol{\omega}; \mathbf{z}, \xi_i)$ are evaluated for vertex connections.

3.2 Hierarchical Rejection

3.2.1 Culling Using BVH. To search light vertices within the scattering range $R(\boldsymbol{\omega}; \mathbf{z}, \xi_i)$ rapidly, we build a binary bounding volume hierarchy (BVH) of light vertices. Using this BVH, the intersection between the scattering range and bounding box of light vertices is tested hierarchically in a top-down fashion. However, while the shape of the scattering range depends only on the approximated



(a) Pregeneration (b) Non-stratified sampling (single precision) (c) Non-stratified sampling (double precision) (d) Semi-stratified sampling (single precision) (e) Reference

Fig. 4. Comparison of random number assignment algorithms for hierarchical Russian roulette. The image quality is evaluated with the mean absolute error (MAE) metric. To emphasize errors, only the results of inefficient glossy-to-glossy connections at the second eye vertex are visualized without MIS weights (16 samples/pixel × 1024 iterations). The pregeneration approach (a) uses a pregenerated single random number for each light vertex similar to Tokuyoshi and Harada [2016]. This produces undesirable artifacts due to the correlation of variance. On-the-fly assignment of non-stratified random numbers (b) produces noticeable precision error of floating point arithmetic. Therefore, expensive double precision has to be used to reduce this error (c). Our on-the-fly semi-stratified random number assignment (d) proposed in Sect. 3.3.3 is not only more numerically stable, but also faster than the non-stratified assignment.

lobe $q_{\mathbf{z}}(\omega)$, the size of the range depends on the random variable $\sqrt{C/\xi_i}$ and is different between light vertices (i.e. leaf nodes). Therefore, to conservatively perform the intersection test for each BVH node, the largest size is used at an internal node (Fig. 3). This size is given by $\sqrt{C/\min_{i\in L}\xi_i}$ where L is the index set of leaves covered by the node. This minimum random number $\min_{i\in L}\xi_i$ for each node has to be obtained efficiently.

3.2.2 Problem of Pregenerated Random Numbers. One trivial approach to obtain the minimum random number for each node is to precompute them using a bottom-up fashion. First, a single random number is assigned to each light vertex in a preprocessing step in a similar way to a stochastic light culling method [Tokuyoshi and Harada 2016]. Then during the BVH construction phase, minimum random numbers are propagated from leaf nodes to higher level nodes. However, since the same precomputed random numbers are used for all the eye vertices, this approach produces the correlation of variance between eye vertices (Fig. 4a). This correlation induces inefficiency if eye vertices are densely sampled (e.g., supersampling). It also degrades the efficiency of variance reduction postprocessing techniques (e.g., image denoising). To avoid the correlation of variance, a different random number has to be assigned to each pair of light and eye vertices. However, a full bottom-up propagation at each eye vertex is prohibitively expensive. We propose a method to generate the minimum random number on-the-fly in a top-down BVH traversal without generating random numbers for all the light vertices and executing a bottom-up propagation.

3.3 On-the-fly Minimum Random Number Generation

We first discuss O(1) methods to generate the minimum value among uniform random numbers in Sect. 3.3.1. Then, we extend the discussion to a top-down hierarchical algorithm to generate a minimum random number for each BVH node. To improve the numerical stability for this algorithm, we use a *semi-stratified sampling* approach that generates uniform and partially stratified random numbers using overlapped strata.

3.3.1 PDF of a Minimum Random Number. The minimum value among N uniform random numbers $\{\xi_1, \ldots, \xi_N\}$ can be generated by considering its PDF. This PDF is analytically obtained as follows:

$$p_{\min,N}(u) = N(1-u)^{N-1},$$

where $u \in [0, 1)$. For the derivation, please refer to the supplemental material. The inverse of the cumulative distribution function for this PDF also has a closed form solution. Therefore, the minimum random number is simply obtained by generating only a single uniform random number $\xi \in [0, 1)$ as follows:

$$\min\{\xi_1, \ldots, \xi_N\} = 1 - (1-\xi)^{\frac{1}{N}}. \tag{3}$$

Stratified Sampling. Although we could use an on-the-fly random number generation based on Eq. (3) for our method, it suffers from precision error for large N (Fig. 4b). However, for the case of 1D stratified sampling, we can avoid the numerical error and simplify the formulation. The minimum value of stratified random numbers is always within the lowest stratum. Hence, it is given by

$$\min\{\xi_1, \ldots, \xi_N\} = \frac{\xi}{N}. \tag{4}$$

The PDF for this case is $p_{\min,N}(u) = N$ if $u \in [0, 1/N)$, otherwise $p_{\min,N}(u) = 0$.

3.3.2 Hierarchical Generation. Using the method in Sect. 3.3.1, we generate the minimum random number $\min_{i\in L}\xi_i$ for each node on-the-fly during a top-down BVH traversal. This is the inverse process of the bottom-up propagation described in Sect. 3.2.2. Unlike the bottom-up approach, our top-down algorithm does not generate random numbers for culled nodes, thus it performs efficiently at each eye vertex in eye subpath tracing.

For our BVH, the minimum random number for a parent node must be equal to the minimum of its two child values. Therefore, in top-down hierarchical generation, the parent's minimum random number is transmitted to one child (illustrated with a blue node in Fig. 5). Then, for the other child (i.e., orange node in Fig. 5), a new minimum random number larger than this parent value is generated. The child inheriting the minimum value from the parent is randomly selected according to the number of leaves covered by the child node. This is because the probability that the minimum value appears in the subtree is proportional to the number of leaves

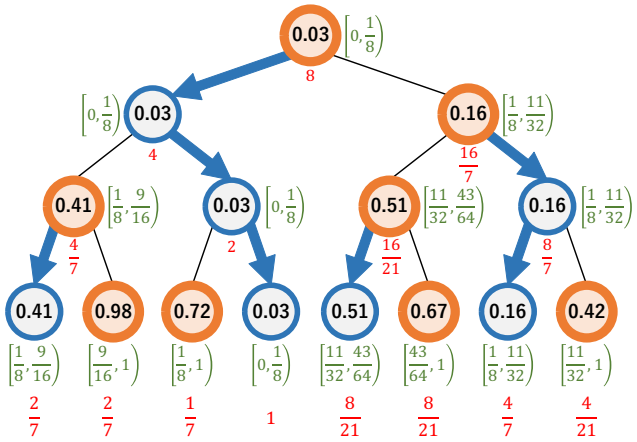


Fig. 5. On-the-fly assignment of minimum random numbers during the top-down tree traversal. At each orange node, the minimum value of random numbers within descendants is generated using Eq. (7). Then, the generated minimum random number is transmitted to one child (blue). The green interval for each node is the stratum for our semi-stratified random number generation (i.e., $[b, b')$ for an orange node). The red value for each node is the PDF (Eq. 8) of our generation and propagation, and it is constant in the stratum. At the leaf level, although our semi-stratified algorithm produces overlapped strata, the sum of the PDFs is one for these overlapped area.

if random numbers are uniformly distributed for leaves. Therefore, this child selection probability is given as follows:

$$P_{\text{selection}}(j) = \frac{|L_j|}{|L_j| + |L_{j'}|}, \tag{5}$$

where j is the node index, L_j is the index set of leaf nodes covered by the node j, and the node index j' is the sibling of the node j. For non-stratified sampling, the above new minimum random number larger than the parent's value can be generated on the basis of Eq. (3) as follows: $\min_{i \in L} \xi_i = \xi_{\min} + (1 - \xi_{\min})\left(1 - (1 - \xi)^{1/|L|}\right)$, where $\xi_{\min} \in [0, 1)$ is the parent's minimum random number transmitted to the sibling node, and $\xi_{\min} = 0$ when generating at the root node. However, this form is numerically unstable as mentioned in Sect. 3.3.1. To improve the numerical stability using stratified sampling, we stratify the random number space only for sibling nodes. Although this semi-stratified approach does not guarantee full stratification for the descendants of those sibling nodes, it generates uniform random numbers for leaf nodes.

3.3.3 Semi-stratified Algorithm. Algorithm 1 shows the pseudo code of our hierarchical Russian roulette using on-the-fly semi-stratified random number assignment. Our method first generates a minimum random number at the root node by assuming 1D stratified sampling. This minimum random number is obtained using Eq. (4) by generating a single uniform random number ξ as follows:

$$\min_{i \in L_{\text{root}}} \xi_i = \frac{\xi}{|L_{\text{root}}|}, \tag{6}$$

where L_{root} is the index set of leaves covered by the root node (i.e., all the light vertices in the cache). In our top-down algorithm, the minimum random number $\xi_{\min}$ at a parent node is transmitted to one child, and a new minimum random number is generated for the other child. For these propagation and generation of random numbers, we use numerically stable stratified sampling. To stratify the random number space between two sibling nodes for minimum random numbers, we transmit the supremum of the stratum $b \in [0, 1]$ from the parent to a child in addition to $\xi_{\min}$. Then, an adjacent stratum $[b, b')$ is used for the other child. The supremum $b' \in (b, 1]$ is determined by stratifying $[b, 1)$ with the number of leaves $|L|$ covered by this child node as follows:

$$b' = b + (1 - b)\frac{1}{|L|}.$$

Although this stratum $[b, b')$ can overlap with strata of the descendants of the sibling node, it is simple to compute. Using the stratum $[b, b')$, the new minimum random number is obtained merely by generating a single uniform random number ξ as follows:

$$\min_{i \in L} \xi_i = b + (1 - b)\frac{\xi}{|L|}. \tag{7}$$

Eq. (6) is a special case of Eq. (7) using $b = 0$ for the root node. This process performs recursively in the BVH traversal. Using this algorithm, we assign a different random number for each pair of light and eye vertices without executing a full bottom-up traversal of the entire tree.

Our on-the-fly random number assignment converts undesirable banding artifacts to high-frequency noise as shown in Fig. 4. Although stratification is partial in our approach, it is beneficial for

ALGORITHM 1: Hierarchical Russian roulette using a binary BVH.

```
function HierarchicalRussianRoulette(z, root)
    |L_root| ← GetLeafCount(root);
    ξ ← GenerateRandomNumber();
    ξ'_min ← ξ / |L_root|;
    b' ← 1 / |L_root|;
    Traverse(z, root, ξ'_min, b');
end
function Traverse(z, node, ξ_min, b)
    if R(ω; z, ξ_min) ∩ AABB(node) ≠ ∅ then
        if IsInternal(node) then
            |L_0| ← GetLeafCount(node.child_0);
            |L_1| ← GetLeafCount(node.child_1);
            ξ ← GenerateRandomNumber();
            if ξ < |L_0| / (|L_0|+|L_1|) then l ← 0; m ← 1;
            else l ← 1; m ← 0;
            ξ ← GenerateRandomNumber();
            ξ'_min ← b + (1 − b) ξ / |L_m|;
            b' ← b + (1 − b) 1 / |L_m|;
            Traverse(z, node.child_l, ξ_min, b);
            Traverse(z, node.child_m, ξ'_min, b');
        else
            y ← GetLightVertex(node);
            if ξ_min < P(y, z) then Connect(y, z);
        end
    end
end
```

the performance and numerical stability (Fig. 4d). For more implementation details, please refer to the C++ example code in the supplemental material.

3.3.4 PDF for Our Semi-stratified Algorithm. Our top-down algorithm avoids overlaps of strata for sibling nodes, but it ignores full stratification for the descendants of these sibling nodes. Therefore, overlaps of strata are produced for leaf nodes. However, the sum of PDFs for these overlapped strata is always one (i.e., uniform distribution), thus our approach is unbiased. For the random number generation within a stratum $[b, b')$, the PDF is given as

$$p_{[b,b')}(u) = \begin{cases} \frac{1}{b'-b} & \text{if } u \in [b, b') \\ 0 & \text{otherwise} \end{cases}.$$

The probability of the generation event occurring at a node is the product of child selection probabilities (Eq. 5) from the root node to that node. The generated random number is propagated to a descendant node according to the product of child selection probabilities from the above generation node to the descendant node. Let X be the set of node indices from the root to a node, then the PDF of the generated random number is yielded as follows:

$$p_X(u) = p_{[b,b')}(u) \prod_{j \in X} P_{\text{selection}}(j), \tag{8}$$

where $P_{\text{selection}}(j) = 1$ if j is the root. When arriving at a leaf node, $\prod_{j \in X} P_{\text{selection}}(j) = 1/|L_{\text{root}}|$ is obtained, thus the PDF is given as $p_X(u) = \frac{1}{(b'-b)|L_{\text{root}}|}$ for $u \in [b, b')$. Fig. 5 illustrates the generation and propagation of random numbers, strata, and their PDFs in our algorithm. As shown in this figure, the sum of the PDFs for the overlapped area is one at leaf nodes.

4 ANISOTROPIC LOBE APPROXIMATION FOR THE ACCEPTANCE PROBABILITY

While the shape of the stochastic scattering range $R(\boldsymbol{\omega}; \mathbf{z}, \xi_i)$ depends on the lobe representation $q_{\mathbf{z}}(\boldsymbol{\omega})$, it is desirable for the shape to be simple for efficient culling implementation (i.e., intersection test between $R(\boldsymbol{\omega}; \mathbf{z}, \xi_i)$ and each BVH node). Although a loose bounding spheroid for this scattering range can be used for a conservative intersection test [Dachsbacher and Stamminger 2006; Tokuyoshi and Harada 2017], it increases false positives especially for anisotropic BRDFs. To build a tighter ellipsoidal range for anisotropic scattering lobes yielded by microfacet BRDFs [Cook and Torrance 1982], we introduce a squared ellipsoidal lobe (SEL) function for the lobe representation. For the efficient implementation of the ellipsoid-box intersection test, please refer to the supplemental material.

4.1 GGX Distribution

Our SEL function is based on the GGX distribution [Trowbridge and Reitz 1975; Walter et al. 2007]:

$$D\left(\boldsymbol{\omega}; \mathbf{Q}, \alpha_x, \alpha_y\right) = \frac{\chi^+(v_z)}{\pi \alpha_x \alpha_y \left(v_x^2/\alpha_x^2 + v_y^2/\alpha_y^2 + v_z^2\right)^2},$$

where $\mathbf{Q} = \begin{bmatrix} \boldsymbol{\omega}_x^{\mathrm{T}} & \boldsymbol{\omega}_y^{\mathrm{T}} & \boldsymbol{\omega}_z^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ is the 3×3 orthogonal matrix which consists of the lobe axes $\boldsymbol{\omega}_x$, $\boldsymbol{\omega}_y$, and $\boldsymbol{\omega}_z$, and $\begin{bmatrix} v_x, v_y, v_z \end{bmatrix}^{\mathrm{T}} = \mathbf{Q}\boldsymbol{\omega}^{\mathrm{T}}$ is the direction transformed into the lobe space. This orthogonal matrix $\mathbf{Q}$ represents the orientation of the lobe. $\chi^+(v_z)$ is the Heaviside function: 1 if $v_z > 0$ otherwise 0. For the isotropic case, this distribution can be rewritten using $\alpha = \alpha_x = \alpha_y$ as follows:

$$D(v_z, \alpha) = \frac{\alpha^2 \chi^+(v_z)}{\pi \left(1 - v_z^2 + \alpha^2 v_z^2\right)^2}.$$

For microfacet BRDFs, the GGX distribution is used for a normal distribution function (NDF, i.e., distribution of microfacet normals) using the following parameterization: $D\left(\boldsymbol{\omega}_h; \mathbf{T}, \alpha_x, \alpha_y\right)$ where $\boldsymbol{\omega}_h = \frac{\omega + \omega'}{\|\omega + \omega'\|}$ is the halfvector, and the orthogonal matrix $\mathbf{T} = \begin{bmatrix} \mathbf{t}_x^{\mathrm{T}} & \mathbf{t}_y^{\mathrm{T}} & \mathbf{n}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ consists of the tangent $\mathbf{t}_x$, bitangent $\mathbf{t}_y$, and normal $\mathbf{n}$ of the surface. For this case, $\begin{bmatrix} \alpha_x, \alpha_y \end{bmatrix}$ represent the surface roughness. The GGX distribution is also used to approximate the scattering lobe (e.g., for image based lighting [Estevez and Lecocq 2018]). In this paper, the GGX distribution is referred to as a *GGX NDF* only when it is used as an NDF. An approximated scattering lobe using the GGX is referred to as a *GGX scattering lobe*. Our lobe approximation using an SEL is built upon a GGX scattering lobe.

4.2 Squared Spheroidal Lobe (SSL)

For the microfacet BRDF, a scattering lobe can be approximated with an isotropic GGX distribution using a half angle:

$$q_{\mathbf{z}}(\boldsymbol{\omega}) \propto D\left(\cos\frac{\theta}{2}, \acute{\alpha}\right), \tag{9}$$

where θ is the angle between $\boldsymbol{\omega}$ and the lobe axis $\boldsymbol{\omega}_z$ (i.e., the dominant direction of the scattering lobe), and $\acute{\alpha}$ is the roughness of the scattering lobe. Using this representation, the scattering range $R(\boldsymbol{\omega}; \mathbf{z}, \xi_i) \propto \sqrt{q_{\mathbf{z}}(\boldsymbol{\omega})}$ becomes a spheroid [Tokuyoshi and Harada 2017]. In this paper, the right side of Eq. (9) without the normalization factor is referred to as the *squared spheroidal lobe*, which is given by

$$\pi \acute{\alpha}^2 D\left(\cos\frac{\theta}{2}, \acute{\alpha}\right) = \frac{4\acute{\alpha}^4}{\left(1 - \cos\theta + \acute{\alpha}^2(1 + \cos\theta)\right)^2}. \tag{10}$$

4.3 Squared Ellipsoidal Lobe (SEL)

The microfacet BRDF produces an anisotropic scattering lobe, even if the NDF is isotropic. Since Eq. (10) is an isotropic spherical function, it cannot represent such anisotropic scattering effects. Therefore, we generalize the squared spheroidal lobe to a squared ellipsoidal lobe (SEL). We yield the SEL using two roughness parameters $\begin{bmatrix} \acute{\alpha}_x, \acute{\alpha}_y \end{bmatrix}$ as follows:

$$\boxed{K\left(\boldsymbol{\omega}; \mathbf{Q}, \acute{\alpha}_x, \acute{\alpha}_y\right) = \frac{4\acute{\alpha}_{\max}^4}{\left(U - v_z + \acute{\alpha}_{\max}^2 (U + v_z)\right)^2}},$$

$$U = \sqrt{\left(\acute{\alpha}_{\max}^2/\acute{\alpha}_x^2\right) v_x^2 + \left(\acute{\alpha}_{\max}^2/\acute{\alpha}_y^2\right) v_y^2 + v_z^2},$$

where $\acute{\alpha}_{\max} = \max\left(\acute{\alpha}_x, \acute{\alpha}_y\right)$. For the derivation, please refer to the supplemental material. The square root of this SEL is a simple ellipsoid whose center and semiaxes in the lobe space are $\left[0, 0, \frac{1-\acute{\alpha}_{\max}^2}{2}\right]$ and $\left[\acute{\alpha}_x, \acute{\alpha}_y, \frac{1+\acute{\alpha}_{\max}^2}{2}\right]$, respectively.

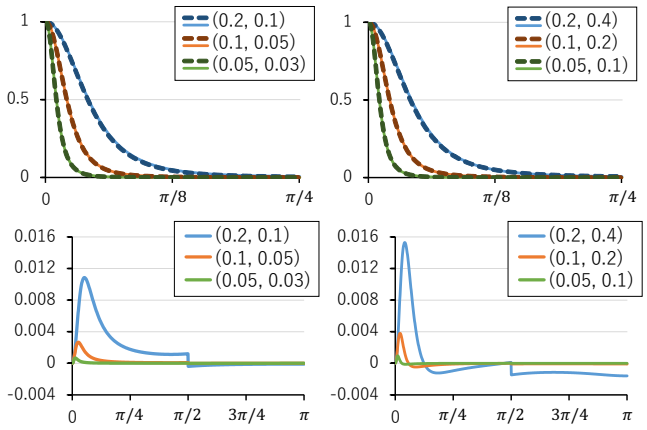


Fig. 6. The upper graphs are plots of GGX lobes (i.e., left side of Eq. (11), solid lines) and our SELs (i.e., right side of Eq. (11), dashed lines) for different GGX roughness parameters. The lower graphs are the differences between these GGX lobes and SELs. The horizontal axis is $\arctan(v_x/v_z)$ for $v_y = 0$. The approximation error of Eq. (11) is smaller for lower roughness.

Approximation of the GGX. The SEL is a spherical function, while the GGX is a hemispherical distribution. However, for small roughness, the following approximation can be used:

$$\pi\alpha_x\alpha_y D\left(\boldsymbol{\omega};\mathbf{Q},\alpha_x,\alpha_y\right) \approx K\left(\boldsymbol{\omega};\mathbf{Q},\frac{\alpha_x}{2},\frac{\alpha_y}{2}\right). \tag{11}$$

Fig. 6 shows the approximation accuracy of Eq. (11) for different roughness parameters. The approximation error is more negligible for smaller roughness values.

4.4 Approximation for an Anisotropic Scattering Lobe

For BPT using MIS, the approximation accuracy for the PDF is important only for around the peak of the scattering lobe (i.e., the product of the BRDF and cosine term as written in Eq. (2)). Therefore, we use the original lobe value at an approximated peak direction $\boldsymbol{\omega}_z$ for the coefficient of an SEL as follows:

$$\boxed{q_{\mathbf{z}}(\boldsymbol{\omega}) = cK\left(\boldsymbol{\omega};\mathbf{Q},\acute{\alpha}_x,\acute{\alpha}_y\right)},$$

where $c = \rho(\mathbf{z},\boldsymbol{\omega}',\boldsymbol{\omega}_z)|\boldsymbol{\omega}_z\cdot\mathbf{n}|$ is the coefficient. For microfacet BRDFs, since the masking-shadowing function and Fresnel are low frequency, the peak direction is approximated with the perfect specular reflection direction $\boldsymbol{\omega}_z = 2(\boldsymbol{\omega}'\cdot\mathbf{n})\mathbf{n}-\boldsymbol{\omega}'$ for smooth surfaces. The lobe axes $\boldsymbol{\omega}_x,\boldsymbol{\omega}_y$ and roughness $\left[\acute{\alpha}_x,\acute{\alpha}_y\right]$ are obtained analytically by *spherical warping* [Xu et al. 2013]. Although spherical warping was derived for anisotropic spherical Gaussians, it is also applicable to the GGX distribution. Therefore, this paper yields the SEL using the following process:

(1) If the NDF is not a GGX, it is approximated with a GGX NDF.
(2) A GGX scattering lobe centered at $\boldsymbol{\omega}_z$ is computed using spherical warping of the GGX NDF.
(3) This GGX scattering lobe is converted to an SEL using Eq. (11).

For the detail of spherical warping, please refer to Appendix A.

5 INTEGRATION INTO BIDIRECTIONAL PATH TRACING

This paper combines our hierarchical Russian roulette-based connections and probabilistic connections [Popov et al. 2015] (or regular vertex connections) by using multiple importance sampling (MIS). This is because our connections are efficient for specular-diffuse-glossy (SDG) or glossy-diffuse-glossy (GDG) paths, while probabilistic connections are efficient for low-frequency illumination effects. Probabilistic connections sample important light vertices for a given eye vertex from the cache using a probability mass function (PMF) which takes the visibility, geometric term, and BRDFs at light and eye vertices into account. However, probabilistic connections have to limit the number of reused light subpaths to a small number (e.g., 100), because its computational overhead and memory requirement are proportional to the product of the reused subpath count and the number of PMF records. Using hundreds of samples is still insufficient for extremely glossy surfaces. Although the acceptance probability of our method ignores the visibility and BRDF at a light vertex, reusing millions of light subpaths alleviates the limitation of probabilistic connections as well as regular vertex connections.

Multiple Importance Sampling. In order to use powerful MIS strategies such as the balance heuristic, the product of the sample count and PDF for each technique must be obtained. Simple Russian roulette for M light subpaths trivially gives this sampling density:

$$d_t(\bar{\mathbf{x}}) = MP(\mathbf{y}_{s-1},\mathbf{z}_{t-1})p_t(\bar{\mathbf{x}}), \tag{12}$$

where $\bar{\mathbf{x}}$ is a path constructed from a light subpath $\{\mathbf{y}_0,\ldots,\mathbf{y}_{s-1}\}$ and eye subpath $\{\mathbf{z}_0,\ldots,\mathbf{z}_{t-1}\}$, and $p_t(\bar{\mathbf{x}})$ is the PDF for regular vertex connections at the eye vertex $\mathbf{z}_{t-1}$. $P(\mathbf{y}_{s-1},\mathbf{z}_{t-1})$ is the acceptance probability of Russian roulette given by Eq. (1) for the connection between $\mathbf{y}_{s-1}$ and $\mathbf{z}_{t-1}$. On the other hand, probabilistic connections ignore the PMF and use $p_t(\bar{\mathbf{x}})$ as the sampling density instead. This is because the PMF for probabilistic connections cannot be obtained for paths constructed by other techniques, and the sample count is modified to one by considering the worst case scenario of the correlation of paths induced by the subpath reuse. Although this modification is conservative to the path correlation, it does not improve the efficiency for SDG or GDG paths. Combining with our method, we are able to sample SDG or GDG paths efficiently as well as other paths, since $P(\mathbf{y}_{s-1},\mathbf{z}_{t-1})$ takes glossy reflections into account (Eq. 12). Although our Russian roulette for many light vertices can also produce the path correlation, the error can be significantly smaller than connecting an eye vertex to all the light vertices by setting a small coefficient C for the acceptance probability $P(\mathbf{y}_{s-1},\mathbf{z}_{t-1})$.

Optimization. To improve the efficiency of this combination, this paper applies our Russian roulette-based connections only to glossy surfaces at eye vertices, because probabilistic connections are more efficient for diffuse surfaces. In addition, we do not reuse $\mathbf{y}_0$ and $\mathbf{y}_1$ (i.e., light vertices on a light source and the first intersected surface) for our connections. This is because a connected path $\{\mathbf{y}_0,\mathbf{y}_1,\mathbf{z}_{t-1}\}$ is never an SDG or GDG configuration. This optimization does not only improve the performance and sampling efficiency, but also reduces the memory usage for cached light vertices.

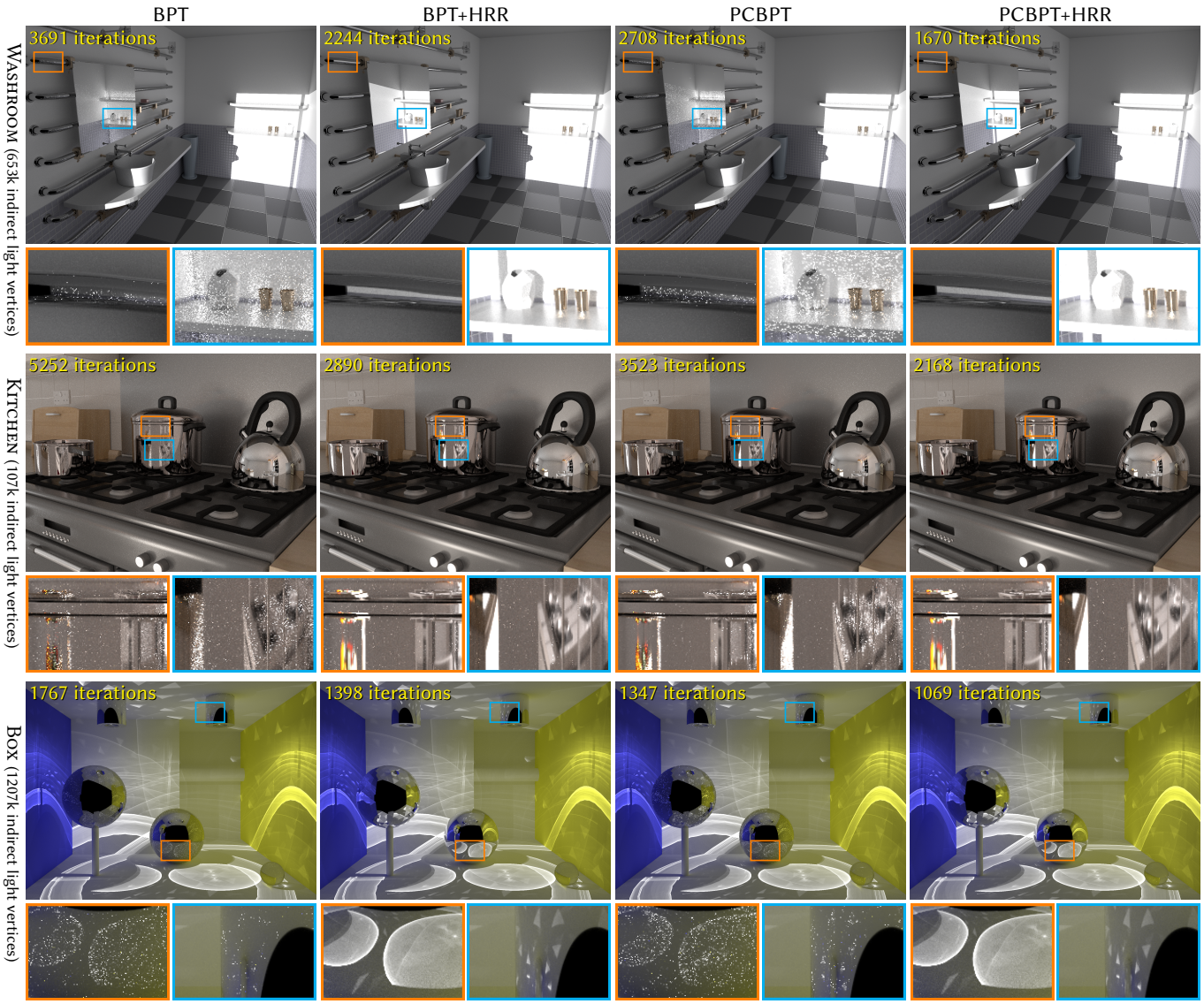


Fig. 7. Equal-time (15 min) comparison of rendering with and without our connections (HRR). All scenes are lit by caustics and have highly glossy materials. Combining our technique with BPT or PCBPT reduces variance significantly for caustics reflected on highly glossy surfaces.

6 EXPERIMENTAL RESULTS

Here we present the rendering results of BPT-based algorithms with and without our hierarchical Russian roulette. Images are rendered at 1600×1200 screen resolution on an AMD Ryzen™ Threadripper™ 2990WX Processor. In our implementation, BPT samples indices of cached light vertices uniformly for vertex connections similar to light vertex cache BPT [Davidovič et al. 2014]. The number of reused light subpaths M is the same as the number of eye subpaths (i.e., $M = 1600 \times 1200$) for BPT, vertex connection and merging (VCM), and our hierarchical Russian roulette-based connections (HRR). For probabilistic connection BPT (PCBPT), we reuse 100 light subpaths, and build importance records on 0.04% of eye subpaths as same as Popov et al. [2015]. For HRR, we set $C = \frac{\Phi}{16M}$ according to a stochastic light culling method [Tokuyoshi and Harada 2017], where Φ is the total radiant flux of light sources. The image quality is evaluated with the root-mean-square error (RMSE) metric and mean absolute error (MAE) metric.

BPT. Fig. 7 shows rendering results of BPT and PCBPT with and without HRR. BPT induces undesirable firefly noises for specular-diffuse-glossy (SDG), glossy-diffuse-glossy (GDG), and specular-diffuse-specular (SDS) paths. PCBPT produces less variance than BPT, however it still induces noticeable correlated noises due to the insufficient sample count on glossy surfaces. By combining with our HRR, these noises are reduced significantly for SDG and GDG

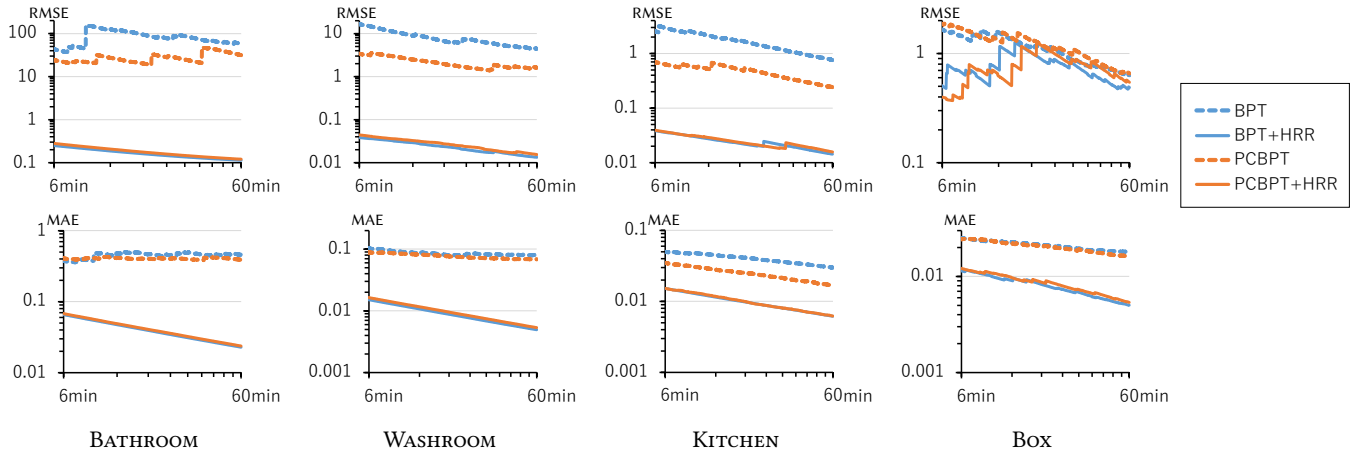


Fig. 8. Plots of RMSE (upper) and MAE (lower) for Fig. 1 and Fig. 7. The oscillations for RMSEs are due to firefly noises which often occur on SDG, GDG, and SDS paths for BPT and PCBPT. Our HRR reduces fireflies for SDG and GDG reflections. Fireflies are also produced on highlights via specular interreflections, especially for the Box scene. The error convergence for the Box scene is visible in the MAE metric which is less sensitive to fireflies.

Table 2. Computation time (sec) per iteration for each pass (Washroom scene). Combining additional techniques increases eye-subpath tracing time since additional subpaths are connected, though it improves the robustness. The BVH construction time for HRR is amortized when combining with VCM.

	BPT	**BPT+HRR**	PCBPT	**PCBPT+HRR**	VCM	**VCM+HRR**	PCVCM	**PCVCM+HRR**
Light-subpath tracing	0.076	**0.071**	0.038	**0.059**	0.073	**0.070**	0.060	**0.061**
BVH construction	-	**0.043**	-	**0.043**	0.044	**0.044**	0.044	**0.044**
Importance records	-	-	0.010	0.010	-	-	0.010	0.010
Eye-subpath tracing	0.173	**0.312**	0.290	**0.435**	0.232	**0.360**	0.359	**0.482**
Total	0.248	**0.426**	0.338	**0.546**	0.350	**0.475**	0.473	**0.597**

reflections. Plots of RMSE and MAE for each scene are shown in Fig. 8. For all the scenes, rendering with HRR reduces error more than without HRR. While PCBPT is more efficient than BPT, BPT with HRR has comparable quality to PCBPT with HRR. Our method reduces fireflies which cause oscillations of RMSE for SDG and GDG reflections, but fireflies due to other paths (e.g., SDS paths) sampled with low probability still remain. For the Box scene, intensive fireflies are also produced on highlights via specular interreflections which are mainly sampled by unidirectional path tracing. Although our technique reduces error for this scene, large oscillations due to highlights still occur for RMSE. For the error convergence for these scenes, please see MAEs which are less sensitive to such outliers.

VCM. Fig. 9 shows the results of VCM using probabilistic connections (PCVCM). For vertex merging, only indirect light vertices (i.e., light vertices excluding $\mathbf{y}_0$ and $\mathbf{y}_1$) are cached to improve the efficiency [Georgiev 2013]. The merging radius is determined using ray differentials [Igehy 1999], and the radius reduction parameter for progressive photon mapping is set by 0.75 to maximize the variance convergence rate [Georgiev et al. 2012a]. Fig. 9a uses five pixels for the initial radius in screen space. For this scene, it produces noticeable light leaks from outside the room. Such light leaks are often less desirable than high-frequency noise, because it is difficult to remove them in postprocessing unlike noise. Although progressive photon mapping reduces such undesirable bias by shrinking the merging radius iteratively, thousands of iterations is still insufficient for this scene. The use of a smaller initial radius reduces light leaks significantly, while it increases variance (Fig. 9b). Combining with HRR, such variance on glossy surfaces is alleviated without increasing the visually unacceptable bias (Fig. 9c). Although HRR does not reduce the variance for SDS paths, the PDF of our connections can be higher than vertex merging for glossy reflections. This is because the PDF of vertex merging is reduced by shrinking the radius iteratively, while our PDF is not. Fig. 10 shows the improvement of the RMSE convergence speed for PCVCM using our technique.

Anisotropy. Fig. 11 shows rendering results of anisotropic reflections. PCBPT (Fig. 11a) produces elongated splotches on the anisotropic reflection surface because of the variance caused by the insufficient sample count. Fig. 11b uses a squared spheroidal lobe (SSL) approximation using isotropic spherical warping [Wang et al. 2009] for the acceptance probability of HRR, however it is still inefficient for such strong anisotropy. On the other hand, the probability using the proposed squared ellipsoidal lobe (SEL) reduces variance significantly (Fig. 11c).

Performance. Table 2 shows the computation time for each pass. The most significant difference is in eye-subpath tracing which is the main bottleneck. Combining HRR increases connected paths in addition to existing vertex connections, and thus the computation time in each iteration is increased by both the BVH traversing

(a) PCVCM (5 pixel initial radius) (b) PCVCM (0.6 pixel initial radius) (c) PCVCM+HRR (0.6 pixel initial radius)

Fig. 9. PCVCM with and without HRR (60 min) for the Bathroom scene. PCVCM with a 5 pixel initial merging radius (a) produces noticeable light leaks (orange box) for this scene. The use of a 0.6 pixel radius (b) avoids these leaks rapidly, while it increases noise (blue box). The combination with HRR (c) reduces the noise for highly glossy reflections without increasing the undesirable bias.

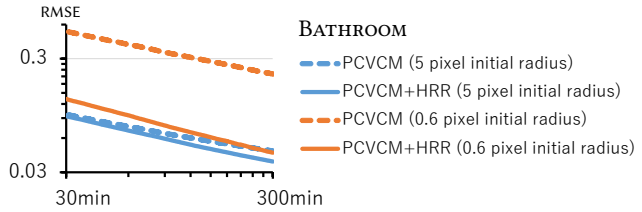
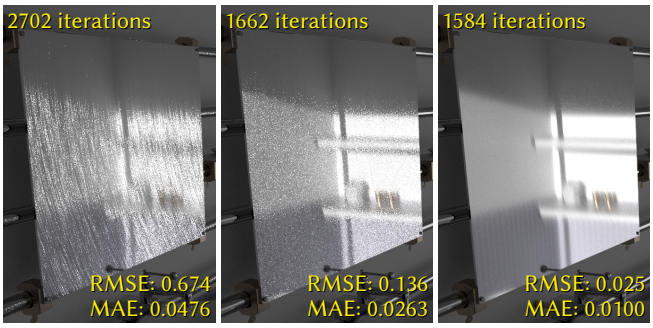


Fig. 10. Plots of RMSE for PCVCM with and without HRR. Combination with HRR improves the error convergence speed for SDG and GDG reflections.



(a) PCBPT (b) PCBPT+HRR (SSL) (c) PCBPT+HRR (SEL)

Fig. 11. Closeups of the Washroom scene (15 min) with anisotropic roughness ($[\alpha_x, \alpha_y] = [0.0001, 0.01]$). Although HRR using a squared spheroidal lobe (SSL) (b) improves the quality from PCBPT (a), it still produces noticeable variance. Using our squared ellipsoidal lobe (SEL) (c), the variance is further reduced significantly.

overhead and additional connections. However, as mentioned in previous paragraphs, the combination with HRR results in fast error reduction for glossy surfaces with a smaller number of iterations than without HRR. One overhead of HRR is the BVH construction. However, when combining our connections with VCM (or PCVCM), the BVH of light vertices can also be used for the range query of photons. Thus, our BVH construction cost is amortized for this combination. The difference for light-subpath tracing is caused by the number of cached light vertices. BPT using light vertex cache stores all the light vertices into the cache, while PCBPT caches only 100 light subpaths. For the combination of PCBPT and HRR (or vertex merging), indirect light vertices for M subpaths are cached additionally. On the other hand, for the combination of HRR and the cache-based BPT (or VCM), HRR reuses indirect light vertices in the cache used for BPT (or VCM). Therefore, the memory requirement and caching cost are also amortized for these combinations. For computation time for other scenes, please refer to the supplemental material. In our implementation, the total memory usage of the BVH and indirect light vertex cache for HRR is 11.8 MB for the Bathroom scene, 69.8 MB for the Washroom scene, 11.4 MB for the Kitchen scene, and 129 MB for the Box scene.

Scalability. Fig. 12 shows the effect of path reuse in our technique for different resolutions (i.e., different number of reused subpaths M). Using $C = \frac{\Phi}{16M}$, the acceptance probability $P(\mathbf{y}, \mathbf{z})$ given by Eq. (1) is almost inversely proportional to M, and thus the expected connection count $MP(\mathbf{y}, \mathbf{z})$ converges to a constant for $M \to \infty$. Nevertheless, our method produces smaller error for a higher resolution. This is because the use of a smaller C for a larger M prevents the probability from being clamped to 1. Therefore, our technique is efficient for a large number of reused subpaths.

7 LIMITATIONS

Perfectly Specular Surfaces. Hierarchical Russian roulette does not support perfectly specular surfaces. Thus, vertex merging has to be used for unconnectable specular-diffuse-specular (SDS) paths (Fig. 13). However, our technique works efficiently for extremely glossy reflections (e.g., GGX roughness 0.0001) that are hard to distinguish from perfectly specular reflections in the final rendered image as shown in Fig. 1 and Fig. 7.

Fireflies. Vertex connections can produce fireflies caused by the squared inverse distance between eye and light vertices. For SDG or GDG paths in BPT, fireflies on the edges of glossy objects cannot be avoided in an unbiased fashion. Although our method reduces fireflies significantly, fireflies can still occur with low probability for this corner case in BPT (Fig. 14a). This is because the probability of Russian roulette has to be clamped to 1 (Eq. 1). However, such fireflies can be removed easily in postprocessing [Zirr et al. 2018]

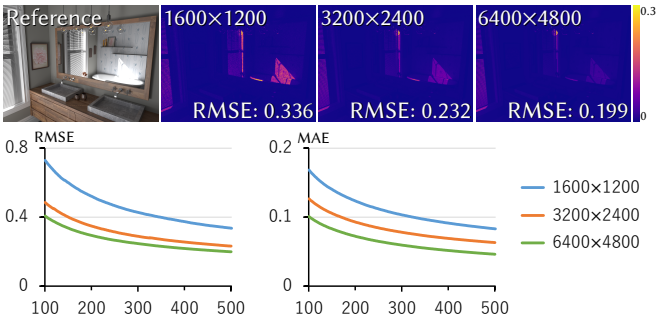


Fig. 12. Comparison between different resolutions for PCBPT+HRR. The upper images are differences from the reference at 500 iterations. The lower graphs are plots of RMSE and MAE, where the horizontal axis is the iteration count. Since the number of reused light subpaths is equal to the resolution, our method produces smaller error for a higher resolution for caustics reflected on the mirror.

(Fig. 14b). Combination with vertex merging using a small merging radius also suppresses fireflies (Fig. 14c).

Path Correlation. Russian roulette for many vertex connections using a small acceptance probability produces less correlation of paths than connecting an eye vertex to all the light vertices. However, it still has the potential to produce noticeable path correlation when light vertices are densely distributed in local, similar to VCM. If an MIS heuristic taking the density of light vertices into account is available, the robustness can be further improved. We refer the reader to Jendersie's research [2019] for MIS considering the light vertex density for VCM.

8 FUTURE WORK

Anisotropic Refractions. For our scattering lobe representation, an existing spherical warping technique for anisotropic reflections [Xu et al. 2013] is used, while spherical warping for anisotropic refractions is not presented yet. Although simple isotropic spherical warping can be used instead [Estevez and Lecocq 2018], it can be inefficient for anisotropic NDFs. Derivation of spherical warping for anisotropic refraction lobes is our future work.

Multi-Lobes. Our current implementation of hierarchical Russian roulette performs for a single lobe representation, thus it has to be used for each lobe for multi-lobe BRDFs. However, hierarchical Russian roulette can support multi-lobes by representing the scattering range using a union of ellipsoids. We would like to investigate the efficiency of this multi-lobe representation in the future.

BRDF at a Light Vertex. Since our technique ignores the BRDF at a light vertex for the Russian roulette probability, it is not efficient for glossy-to-glossy interreflections and glossy-diffuse-specular (GDS) paths unlike SDG paths. For GDS paths, we can use a reverse form of the proposed hierarchical Russian roulette, i.e., intersection test between the scattering range from a light vertex and BVH of eye vertices. However, this approach has to cache eye vertices in addition to light vertices, and the implementation is more complex than the proposed technique. An efficient method taking the BRDF at a light vertex into account is a future work.

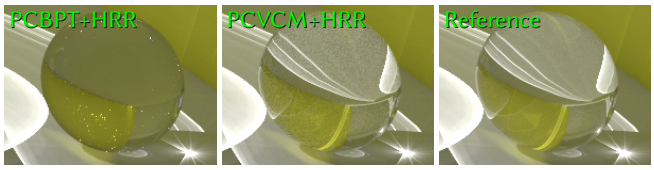


Fig. 13. Closeups of the Box scene (60 min) for perfectly SDS paths. Since our HRR is an acceleration technique for vertex connections, it does not improve these unconnectable paths.

(a) PCBPT+HRR (b) PCBPT+HRR+OR (c) PCVCM+HRR

Fig. 14. Closeups of the Box scene (60 min). Even combining BPT with our technique (a), firefly noise can occur for SDG reflections when connected light and eye vertices are very close. However, these fireflies can be removed easily by using outlier rejection (OR) [Zirr et al. 2018] in postprocessing (b). Combination with VCM (c) also suppresses them.

9 CONCLUSIONS

We have presented an acceleration technique for a huge number of Russian roulette operations for connections between an eye vertex and many cached light vertices. This technique rejects light vertices by testing the intersection between the BVH of light vertices and the world-space acceptance range from an eye vertex. To generate a different random number for each Russian roulette operation, we have proposed an efficient hierarchical random number assignment algorithm. Since this algorithm performs in the top-down BVH traversal, it generates random numbers only for visited nodes. In addition, to improve the efficiency of this hierarchical Russian roulette, we also used a tight ellipsoidal range. To build the ellipsoidal range, we introduced a squared ellipsoidal lobe function to approximate an anisotropic scattering lobe created by the microfacet BRDF. Since our technique accelerates Russian roulette unlike existing many-light methods, it is easy to integrate into BPT-based algorithms without changing the heuristic of multiple importance sampling. Although combining with an additional sampling technique induces an overhead, the combination with our technique improves the robustness for specular-diffuse-glossy or glossy-diffuse-glossy paths which are often problematic for BPT. Our technique does not support perfectly specular surfaces, however it works efficiently for extremely glossy reflections that are hard to distinguish from perfectly specular reflections.

ACKNOWLEDGMENTS

We would like to thank Mareck for the Bathroom scene, Jay-Artist for the Kitchen scene, and T. Hachisuka for the Box scene. The Bathroom scene and Kitchen scene are distributed by B. Bitterli.

We thank Y. Kimura for providing the original model of the Washroom scene. We would also like to thank the anonymous reviewers for their insightful comments and constructive suggestions.

REFERENCES

J. Arvo and D. Kirk. 1990. Particle Transport and Image Synthesis. *SIGGRAPH Comput. Graph.* 24, 4 (1990), 63–66. https://doi.org/10.1145/97879.97886

R. L. Cook and K. E. Torrance. 1982. A Reflectance Model for Computer Graphics. *ACM Trans. Graph.* 1, 1 (1982), 7–24. https://doi.org/10.1145/357290.357293

C. Dachsbacher, J. Křivánek, M. Hašan, A. Arbree, B. Walter, and J. Novák. 2014. Scalable Realistic Rendering with Many-Light Methods. *Comput. Graph. Forum* 33, 1 (2014), 88–104. https://doi.org/10.1111/cgf.12256

C. Dachsbacher and M. Stamminger. 2006. Splatting Indirect Illumination. In *I3D '06*. 93–100. https://doi.org/10.1145/1111411.1111428

T. Davidovič, J. Křivánek, M. Hašan, and P. Slusallek. 2014. Progressive Light Transport Simulation on the GPU: Survey and Improvements. *ACM Trans. Graph.* 33, 3 (2014), 29:1–29:19. https://doi.org/10.1145/2602144

A. C. Estevez and C. Kulla. 2018. Importance Sampling of Many Lights with Adaptive Tree Splitting. *Proc. ACM Comput. Graph. Interact. Tech.* 1, 2 (2018), 25:1–25:17. https://doi.org/10.1145/3233305

A. C. Estevez and P. Lecocq. 2018. Fast Product Importance Sampling of Environment Maps. In *SIGGRAPH '18 Talks*. 69:1–69:2. https://doi.org/10.1145/3214745.3214760

I. Georgiev. 2013. Combining Photon Mapping and Bidirectional Path Tracing. In *SIGGRAPH Asia '13 Course: State of the Art in Photon Density Estimation*. 15:475–15:515. https://doi.org/10.1145/2542266.2542281

I. Georgiev, J. Křivánek, T. Davidovič, and P. Slusallek. 2012a. Light Transport Simulation with Vertex Connection and Merging. *ACM Trans. Graph.* 31, 6 (2012), 192:1–192:10. https://doi.org/10.1145/2366145.2366211

I. Georgiev, J. Křivánek, S. Popov, and P. Slusallek. 2012b. Importance Caching for Complex Illumination. *Comput. Graph. Forum* 31, 2pt3 (2012), 701–710. https://doi.org/10.1111/j.1467-8659.2012.03049.x

T. Hachisuka, S. Ogaki, and H. W. Jensen. 2008. Progressive Photon Mapping. *ACM Trans. Graph.* 27, 5 (2008), 130:1–130:8. https://doi.org/10.1145/1409060.1409083

T. Hachisuka, J. Pantaleoni, and H. W. Jensen. 2012. A Path Space Extension for Robust Light Transport Simulation. *ACM Trans. Graph.* 31, 6 (2012), 191:1–191:10. https://doi.org/10.1145/2366145.2366210

S. Herholz, O. Elek, J. Schindel, J. Křivánek, and H. P. A. Lensch. 2018. A Unified Manifold Framework for Efficient BRDF Sampling Based on Parametric Mixture Models. In *EGSR '18 EI&I*. 41–52. https://doi.org/10.2312/sre.20181171

S. Herholz, O. Elek, J. Vorba, H. Lensch, and J. Křivánek. 2016. Product Importance Sampling for Light Transport Path Guiding. *Comput. Graph. Forum* 35, 4 (2016), 67–77. https://doi.org/10.1111/cgf.12950

H. Igehy. 1999. Tracing Ray Differentials. In *SIGGRAPH '99*. 179–186. https://doi.org/10.1145/311535.311555

J. Jendersie. 2019. Variance Reduction via Footprint Estimation in the Presence of Path Reuse. In *Ray Tracing Gems: High-Quality and Real-Time Rendering with DXR and Other APIs*. Apress, 557–569. https://doi.org/10.1007/978-1-4842-4427-2_31

A. Keller. 1997. Instant Radiosity. In *SIGGRAPH '97*. 49–56. https://doi.org/10.1145/258734.258769

C. Knaus and M. Zwicker. 2011. Progressive Photon Mapping: A Probabilistic Approach. *ACM Trans. Graph.* 30, 3 (2011), 25:1–25:13. https://doi.org/10.1145/1966394.1966404

T. Kollig and A. Keller. 2006. Illumination in the Presence of Weak Singularities. In *MCQMC '04*. 245–257. https://doi.org/10.1007/3-540-31186-6_15

E. P. Lafortune and Y. D. Willems. 1993. Bi-Directional Path Tracing. In *Compugraphics '93*. 145–153.

O. Olsson, M. Billeter, and E. Persson. 2014. Efficient Real-Time Shading with Many Lights. In *SIGGRAPH Asia '14 Courses*. 11:1–11:310. https://doi.org/10.1145/2659467.2659475

B. T. Phong. 1975. Illumination for Computer Generated Pictures. *Commun. ACM* 18, 6 (1975), 311–317. https://doi.org/10.1145/360825.360839

S. Popov, R. Ramamoorthi, F. Durand, and G. Drettakis. 2015. Probabilistic Connections for Bidirectional Path Tracing. *Comput. Graph. Forum* 34, 4 (2015), 75–86. https://doi.org/10.1111/cgf.12680

J. Stewart. 2015. Compute-Based Tiled Culling. In *GPU Pro 6: Advanced Rendering Techniques*. A K Peters/CRC Press, 435–458.

Y. Tokuyoshi and T. Harada. 2016. Stochastic Light Culling. *J. Comput. Graph. Tech.* 5, 1 (2016), 35–60.

Y. Tokuyoshi and T. Harada. 2017. Stochastic Light Culling for VPLs on GGX Microsurfaces. *Comput. Graph. Forum* 36, 4 (2017), 55–63. https://doi.org/10.1111/cgf.13224

Y. Tokuyoshi and T. Harada. 2018. Bidirectional Path Tracing Using Backward Stochastic Light Culling. In *SIGGRAPH '18 Talks*. 70:1–70:2. https://doi.org/10.1145/3214745.3214750

T. S. Trowbridge and K. P. Reitz. 1975. Average Irregularity Representation of a Rough Surface for Ray Reflection. *J. Opt. Soc. Am.* 65, 5 (1975), 531–536. https://doi.org/10.1364/JOSA.65.000531

E. Veach. 1998. *Robust Monte Carlo Methods for Light Transport Simulation*. Ph.D. Dissertation.

E. Veach and L. Guibas. 1994. Bidirectional Estimators for Light Transport. In *EGWR '94*. 147–162.

E. Veach and L. J. Guibas. 1995. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *SIGGRAPH '95*. 419–428. https://doi.org/10.1145/218380.218498

J. Vorba, O. Karlík, M. Šik, T. Ritschel, and J. Křivánek. 2014. On-line Learning of Parametric Mixture Models for Light Transport Simulation. *ACM Trans. Graph.* 33, 4 (2014), 101:1–101:11. https://doi.org/10.1145/2601097.2601203

B. Walter, A. Arbree, K. Bala, and D. P. Greenberg. 2006. Multidimensional Lightcuts. *ACM Trans. Graph.* 25, 3 (2006), 1081–1088. https://doi.org/10.1145/1141911.1141997

B. Walter, S. Fernandez, A. Arbree, K. Bala, M. Donikian, and D. P. Greenberg. 2005. Lightcuts: A Scalable Approach to Illumination. *ACM Trans. Graph.* 24, 3 (2005), 1098–1107. https://doi.org/10.1145/1073204.1073318

B. Walter, P. Khungurn, and K. Bala. 2012. Bidirectional Lightcuts. *ACM Trans. Graph.* 31, 4 (2012), 59:1–59:11. https://doi.org/10.1145/2185520.2185555

B. Walter, S. R. Marschner, H. Li, and K. E. Torrance. 2007. Microfacet Models for Refraction Through Rough Surfaces. In *EGSR '07*. 195–206. https://doi.org/10.2312/EGWR/EGSR07/195-206

J. Wang, P. Ren, M. Gong, J. Snyder, and B. Guo. 2009. All-Frequency Rendering of Dynamic, Spatially-Varying Reflectance. *ACM Trans. Graph.* 28, 5 (2009), 133:1–133:10. https://doi.org/10.1145/1618452.1618479

K. Xu, W.-L. Sun, Z. Dong, D.-Y. Zhao, R.-D. Wu, and S.-M. Hu. 2013. Anisotropic Spherical Gaussians. *ACM Trans. Graph.* 32, 6 (2013), 209:1–209:11. https://doi.org/10.1145/2508363.2508386

T. Zirr, J. Hanika, and C. Dachsbacher. 2018. Reweighting Firefly Samples for Improved Finite-Sample Monte Carlo Estimates. *Comput. Graph. Forum* 37, 6 (2018), 410–421. https://doi.org/10.1111/cgf.13335

A SPHERICAL WARPING FOR ANISOTROPIC LOBES

Since the halfvector $\boldsymbol{\omega}_h$ depends on the outgoing direction $\boldsymbol{\omega}$, the GGX NDF is rewritten as:

$$D\left(\boldsymbol{\omega}_h(\boldsymbol{\omega}); \mathbf{T}, \alpha_x, \alpha_y\right) = \frac{\chi^+\left(\boldsymbol{\omega}_h(\boldsymbol{\omega}) \cdot \mathbf{n}\right)}{\pi \alpha_x \alpha_y \left(g(\boldsymbol{\omega}) + 1\right)^2},$$

$$g(\boldsymbol{\omega}) = \lambda_x (\boldsymbol{\omega}_h(\boldsymbol{\omega}) \cdot \mathbf{t}_x)^2 + \lambda_y (\boldsymbol{\omega}_h(\boldsymbol{\omega}) \cdot \mathbf{t}_y)^2, \tag{13}$$

where $\lambda_x = \frac{1}{\alpha_x^2} - 1$, and $\lambda_y = \frac{1}{\alpha_y^2} - 1$. Spherical warping [Xu et al. 2013] approximates Eq. (13) using a second order Taylor expansion:

$$g(\boldsymbol{\omega}) \approx \begin{bmatrix} \boldsymbol{\omega} \cdot \boldsymbol{\omega}'_x & \boldsymbol{\omega} \cdot \boldsymbol{\omega}'_y \end{bmatrix} \mathbf{H} \begin{bmatrix} \boldsymbol{\omega} \cdot \boldsymbol{\omega}'_x & \boldsymbol{\omega} \cdot \boldsymbol{\omega}'_y \end{bmatrix}^{\mathrm{T}},$$

where $\boldsymbol{\omega}'_x = \frac{\boldsymbol{\omega}' \times \mathbf{n}}{\|\boldsymbol{\omega}' \times \mathbf{n}\|}$, $\boldsymbol{\omega}'_y = \frac{\boldsymbol{\omega}_z \times \boldsymbol{\omega}'_x}{\|\boldsymbol{\omega}_z \times \boldsymbol{\omega}'_x\|}$, and $\mathbf{H}$ is the 2×2 Hessian matrix of $g(\boldsymbol{\omega})$. At the peak direction $\boldsymbol{\omega}_h(\boldsymbol{\omega}) = \mathbf{n}$ for the NDF, $\mathbf{H}$ is given by

$$\mathbf{H} = \begin{bmatrix} \frac{\lambda_x (\boldsymbol{\omega}'_x \cdot \mathbf{t}_x)^2 + \lambda_y (\boldsymbol{\omega}'_x \cdot \mathbf{t}_y)^2}{4(\boldsymbol{\omega}' \cdot \mathbf{n})^2} & \frac{(\lambda_y - \lambda_x)(\boldsymbol{\omega}'_x \cdot \mathbf{t}_x)(\boldsymbol{\omega}'_x \cdot \mathbf{t}_y)}{4(\boldsymbol{\omega}' \cdot \mathbf{n})} \\ \frac{(\lambda_y - \lambda_x)(\boldsymbol{\omega}'_x \cdot \mathbf{t}_x)(\boldsymbol{\omega}'_x \cdot \mathbf{t}_y)}{4(\boldsymbol{\omega}' \cdot \mathbf{n})} & \frac{\lambda_y (\boldsymbol{\omega}'_x \cdot \mathbf{t}_x)^2 + \lambda_x (\boldsymbol{\omega}'_x \cdot \mathbf{t}_y)^2}{4} \end{bmatrix}.$$

Eigendecomposition for $\mathbf{H}$ yields

$$g(\boldsymbol{\omega}) \approx \lambda'_x (\boldsymbol{\omega} \cdot \boldsymbol{\omega}_x)^2 + \lambda'_y (\boldsymbol{\omega} \cdot \boldsymbol{\omega}_y)^2,$$

where λ'_x and λ'_y are the eigenvalues of $\mathbf{H}$, and

$$\begin{bmatrix} \boldsymbol{\omega}_x \\ \boldsymbol{\omega}_y \end{bmatrix} = \begin{bmatrix} \mathbf{e}_x \\ \mathbf{e}_y \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega}'_x \\ \boldsymbol{\omega}'_y \end{bmatrix},$$

where $\mathbf{e}_x$ and $\mathbf{e}_y$ are the eigenvectors of $\mathbf{H}$. Therefore, the NDF is approximated with a GGX scattering lobe centered at the perfect specular reflection vector $\boldsymbol{\omega}_z$ as follows:

$$D\left(\boldsymbol{\omega}_h; \mathbf{T}, \alpha_x, \alpha_y\right) \approx D\left(\boldsymbol{\omega}; \mathbf{Q}, \alpha'_x, \alpha'_y\right),$$

where $\mathbf{Q} = \begin{bmatrix} \boldsymbol{\omega}_x^{\mathrm{T}} & \boldsymbol{\omega}_y^{\mathrm{T}} & \boldsymbol{\omega}_z^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, $\alpha'_x = \frac{1}{\sqrt{\lambda'_x + 1}}$, and $\alpha'_y = \frac{1}{\sqrt{\lambda'_y + 1}}$.