Bidirectional Path Tracing Using Backward Stochastic Light Culling

Yusuke Tokuyoshi
SQUARE ENIX CO., LTD.
tokuyosh@square-enix.com

Takahiro Harada
Advanced Micro Devices, Inc.
Takahiro.Harada@amd.com

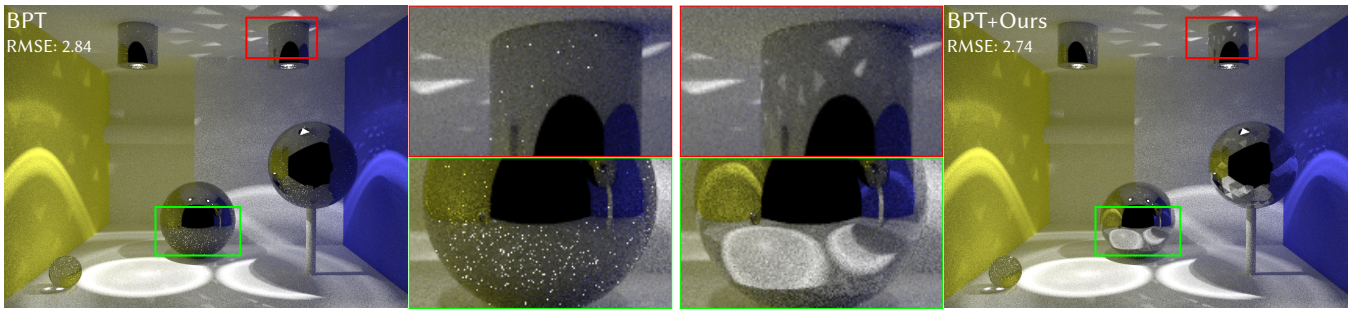

Figure 1: Equal-time (180 s) comparison of light vertex cache bidirectional path tracing (LVC-BPT) [Davidovič et al. 2014] with and without our method. BPT (left) produces noticeable noise for specular-diffuse-glossy paths. Our method (right) significantly reduces this noise on highly glossy objects (GGX roughness: 0.001). For this scene, 32M light subpaths are traced and 73936784 subpath vertices are stored in the LVC.

ABSTRACT

Bidirectional path tracing (BPT) produces noticeable variance for specular-diffuse-specular reflections even if they are not perfectly specular. This is because sampling of the connection between a light vertex and eye vertex does not take bidirectional reflectance distribution functions (BRDFs) into account. This paper presents a novel unbiased sampling method referred to as *backward stochastic light culling* which addresses the problem of specular-diffuse-glossy reflections. Our method efficiently performs *Russian roulette* for many light vertices according to the glossy BRDF at a given eye vertex using a hierarchical culling algorithm. We combine our method with light vertex cache-based BPT using multiple importance sampling to significantly reduce variance when rendering caustics reflected on highly glossy surfaces.

CCS CONCEPTS

• **Computing methodologies** → **Ray tracing**;

KEYWORDS

global illumination, bidirectional path tracing, Russian roulette

ACM Reference Format:
Yusuke Tokuyoshi and Takahiro Harada. 2018. Bidirectional Path Tracing Using Backward Stochastic Light Culling. In *Proceedings of SIGGRAPH '18 Talks.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3214745.3214750

SIGGRAPH '18 Talks, August 12-16, 2018, Vancouver, BC, Canada

ACM ISBN 978-1-4503-5820-0/18/08.
https://doi.org/10.1145/3214745.3214750

1 INTRODUCTION

Bidirectional path tracing (BPT) is a well-established light transport algorithm which combines two subpaths traced from a light source and eye using multiple importance sampling (MIS) [Veach and Guibas 1995]. However, BPT produces noticeable variance for specular-diffuse-specular paths even if they are not perfectly specular (Fig. 1). This is because sampling of the connection between two subpath vertices (i.e., light vertex and eye vertex) does not take BRDFs into account. In this paper, we present a *backward stochastic light culling* method to sample a few light vertices from a light vertex cache (LVC) [Davidovič et al. 2014] using *Russian roulette* according to the specular BRDF at a given eye vertex. To accelerate this Russian roulette for many light vertices, we introduce a hierarchical algorithm to cull them efficiently. In addition, we also present a random number assignment technique to avoid the correlation of variance for our stochastic light culling. Since our method can be easily combined with LVC-BPT using MIS, we are able to reduce the variance for caustics reflected on highly glossy surfaces in an unbiased fashion.

2 OUR METHOD

Our method is based on stochastic light culling [Tokuyoshi and Harada 2017] which was developed for real-time rendering. This technique first restricts the range of influence for each light vertex based on Russian roulette, and then culls light vertices before shading using the bounding ellipsoid of this light range. The previous work ignored the BRDFs of eye vertices, and a single random number was assigned to each light vertex to reuse real-time culling

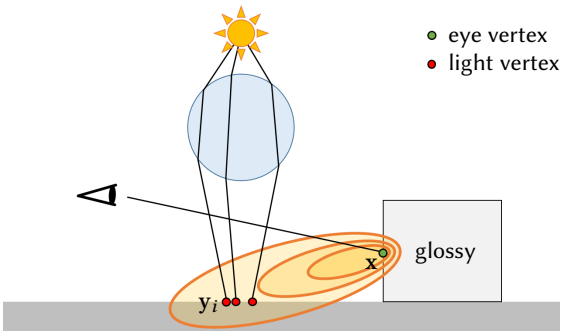

Figure 2: Backward stochastic light culling gathers light vertices within the ellipsoid whose shape is determined using the BRDF at an eye vertex. The ellipsoid size is randomly changed for each light vertex.

algorithms. Unlike this real-time method, our method determines the shape of the ellipsoid using the specular BRDF at an eye vertex (Fig. 2), and assigns a different random number for each pair of light and eye vertices.

2.1 Probability of Russian Roulette

For the connection between an eye vertex $\mathbf{x}$ and the ith light vertex $\mathbf{y}_i$, we use Russian roulette with the following probability:

$$p(\mathbf{x}, \mathbf{y}_i) = \min\left(\frac{c_i f(\mathbf{x}, \boldsymbol{\omega}', \boldsymbol{\omega}_i) \max(\boldsymbol{\omega}_i \cdot \mathbf{n}, 0)}{\|\mathbf{x} - \mathbf{y}_i\|^2}, 1\right), \tag{1}$$

where $f(\cdot)$ is the specular BRDF, $\boldsymbol{\omega}'$ is the direction from $\mathbf{x}$ to the previous vertex in the eye subpath, $\boldsymbol{\omega}_i$ is the direction from $\mathbf{x}$ to $\mathbf{y}_i$, $\mathbf{n}$ is the surface normal at $\mathbf{x}$, and c_i is a constant term to control the variance (for details, please see the supplementary material). Unlike resampling using all the light vertices (e.g., lightcuts [Walter et al. 2005]), this Russian roulette uses only the local information of two vertices. Hence, our method is easily combined with BPT using MIS. The light range for direction $\boldsymbol{\omega}$ is the inverse of the above probability as follows:

$$l(\boldsymbol{\omega}) = \sqrt{\frac{c_i f(\mathbf{x}, \boldsymbol{\omega}', \boldsymbol{\omega}) \max(\boldsymbol{\omega} \cdot \mathbf{n}, 0)}{\xi_i}}, \tag{2}$$

where $\xi_i \in [0, 1)$ is the uniform random number generated for the ith vertex. Similar to the previous work, the bounding ellipsoid of this light range is used for simplicity (for details, please refer to the supplementary material). To accelerate Russian roulette, only light vertices within this ellipsoid are evaluated.

2.2 Hierarchical Stochastic Light Culling

To search light vertices within the ellipsoid efficiently, we build a bounding volume hierarchy (BVH) of light vertices. Using this BVH, the intersection between the ellipsoid and bounding box of light vertices is tested hierarchically in a top-down fashion. However, while the shape of the ellipsoid depends only on the given eye vertex $\mathbf{x}$, the size of the ellipsoid depends on $\sqrt{\frac{c_i}{\xi_i}}$ of each light vertex. Therefore, to conservatively perform the intersection test for each BVH node, the maximum size of the ellipsoid is used at an internal node. This size is obtained by $\sqrt{\frac{\max_{i\in L} c_i}{\min_{i\in L} \xi_i}}$ where L is the set of light vertices (i.e., leaf nodes) covered by the node. For our method, $\max_{i\in L} c_i$ is stored into each node in preprocessing, while $\min_{i\in L} \xi_i$ is computed on-the-fly in tree traversal.

2.3 On-the-fly Semi-Stratified Random Number Assignment

Although the minimum of ξ_i can also be stored into each node by precomputing a single ξ_i for each light vertex, this approach produces banding artifacts due to the correlation of variance between eye vertices (please see the supplementary material). This correlation induces inefficiency when eye vertices are densely sampled (e.g., super sample antialiasing). Therefore, we generate the minimum of ξ_i for each node on-the-fly during the top-down tree traversal. Assuming 1D stratified sampling, the minimum of stratified random numbers is within the lowest stratum. Therefore, the minimum random number for a node is obtained by generating a single random number $\xi \in [0, 1)$ as follows:

$$\min_{i\in L} \xi_i = s + (1 - s)\frac{\xi}{|L|}, \tag{3}$$

where $|L|$ is the number of leaves covered by the node, s is the lower bound of the stratum given from the parent node, and $s = 0$ for the root node. When generating this minimum random number, the lower bound is updated simultaneously for the next random number as follows:

$$s' = s + (1 - s)\frac{1}{|L|}. \tag{4}$$

Since the minimum random number for a parent node is the minimum of two child nodes' values, the parent's minimum random number and lower bound are transferred into either one child node. This child is randomly selected according to the number of leaves covered by the child node. For the other child, a new minimum random number and lower bound are generated using Eqs. (3) and (4). Finally, this algorithm produces uniform and partially stratified random numbers for leaves. Using this algorithm, we assign a different random number for each pair of light and eye vertices without executing a full bottom-up traversing of the entire tree.

3 RESULTS AND FUTURE WORK

Fig. 1 shows our experimental results. For this experiment, backward stochastic light culling is applied to the first non-perfectly specular eye vertex in eye subpath tracing. Although our method is applicable only to rough specular reflections, noise is significantly reduced for caustics reflected on highly-glossy surfaces without bias and correlation. Extension to refractions will be a part of a future work.

REFERENCES

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, Article 29 (2014), 19 pages.

Y. Tokuyoshi and T. Harada. 2017. Stochastic Light Culling for VPLs on GGX Microsurfaces. *Comput. Graph. Forum* 36, 4 (2017), 55–63.

E. Veach and L. J. Guibas. 1995. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *SIGGRAPH '95*. 419–428.

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.