Guiding Light Trees for Many-Light Direct Illumination

Eric Hamann, Alisa Jung and Carsten Dachsbacher

Karlsruhe Institute of Technology, Department of Informatics, Germany

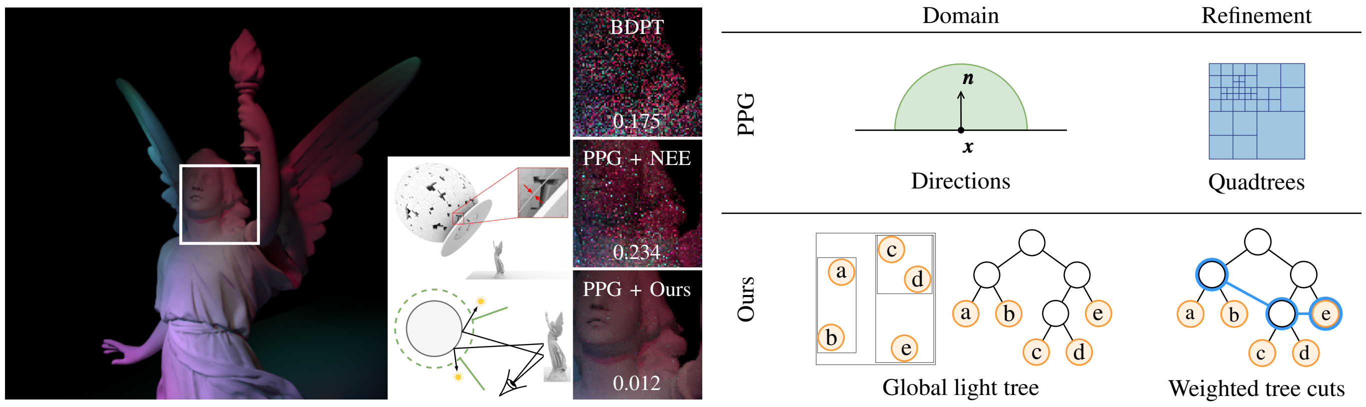

Figure 1: ***Left***: *Equal time comparison (40 min) and RMSE (insets) of bidirectional path tracing (BDPT), Practical Path Guiding (PPG) with NEE, and our sampling technique, in a scene with 24.6k lights along a ring (red inset) and difficult visibility. Our technique guides the last path segment to relevant light sources, with little overhead but significant improvements over PPG.* ***Right***: *PPG approximates the total incident radiance by subdividing the directional domain with a quadtree, using the quadtree entries to sample directions. We approximate the direct incident radiance by subdividing a light BVH with a tree cut (bottom right, blue), using the node weights to sample emitters.*

Abstract

Path guiding techniques reduce the variance in path tracing by reusing knowledge from previous samples to build adaptive sampling distributions. The Practical Path Guiding (PPG) approach stores and iteratively refines an approximation of the incident radiance field in a spatio-directional data structure that allows sampling the incident radiance. However, due to the limited resolution in both spatial and directional dimensions, this discrete approximation is not able to accurately capture a large number of very small lights. We present an emitter sampling technique to guide next event estimation (NEE) with a global light tree and adaptive tree cuts that integrates into the PPG framework. In scenes with many lights our technique significantly reduces the RMSE compared to PPG with uniform NEE, while adding close to no overhead in scenes with few light sources. The results show that our technique can also aid the incident radiance learning of PPG in scenes with difficult visibility.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Ray tracing;***

1. Introduction

Modern scenes in photorealistic rendering often contain thousands or even millions of light sources. The direct scattered radiance $L_o(x,\omega_o)$ at a surface point x due to the direct illumination by these light sources can be modeled as the sum over the surfaces $\mathcal{S}_l$ of the $l = 1,\dots,N$ light sources:

$$L_o(x,\omega_o) = \sum_{l=1}^{N} \int_{\mathcal{S}_l} L_e(y \to x)\, f_r(y \to x,\omega_o)\, G(x,y)\, \mathrm{d}A(y). \qquad (1)$$

Next Event Estimation (NEE) approximates this equation numerically with Monte Carlo integration by sampling a vertex y on a light source from a probability density function (PDF) $p(y)$:

$$L_o(x,\omega_o) \approx \frac{L_e(y \to x)\, f_r(y \to x,\omega_o)\, G(y,x)}{p(y)}.$$

The PDF $p(y)$ is usually a product of a probability mass function (PMF) $P(l)$ for first sampling a light source l, and a PDF p_l on that light source for sampling the vertex itself.

DOI: 10.2312/egs.20231004

The variance of this estimate can be reduced if lights can be sampled according to a PDF as proportional to the integrand as possible. In scenes with only a handful of light sources, selecting light sources proportional to their overall power usually works well. However, the contribution of a light vertex y to a scene vertex x also depends on the distance and orientation of the light and the surface, the visibility between x and y, and the reflectance behaviour at x. In scenes with thousands of light sources, this simple strategy often results in stronger mismatches between the PDF and the integrand, and thus exhibits higher variance. To improve this we need the PDF to depend on x, ω_o, or both.

Previous Work achieves this by considering distance and orientation in a pre-build light hierarchy [EK18] that tailors the PDF to the scene vertex x, or by using spatio-temporal reservoir sampling [BWP*20] to reuse and share light samples between pixels. Lightcuts [WFA*05] on the other hand approximate the cumulative influence of less contributing lights by clustering lights in a tree which lowers variance but introduces bias. Vevoda et al. [VKK18] learn cluster selection probabilities using Bayesian regression at run time for improved emitter sampling, but rely on light clusterings that are constructed up front, based on conservative contribution estimates. Pantaleoni [Pan19] proposes adaptive, weighted cuts through a light tree to approximate direct illumination and sample light sources. Our technique reuses the idea of adaptive tree cuts but differs in the way the cuts are refined and how lights are sampled *inside* a selected light cluster. Aiming at realtime scenarios, Pantaleoni [Pan19] splits and merges only two nodes of the adaptive cut in each refinement iteration while our technique, aimed at offline rendering, generates a roughly equi-energy tree cut with a configurable relative energy threshold in each iteration. The technique by Pantaleoni selects a light uniformly in a light cluster while our technique takes global energy estimates into account.

Practical Path Guiding Guiding approaches learn the entire (direct and indirect) incident radiance during rendering and use this information to guide future paths in directions with high incident energy. Practical Path Guiding (PPG) [MGN17, Mü19] stores a discrete approximation of the incident radiance field in a 5D data structure, where the leaf nodes of a 3D spatial binary tree store 2D quadtrees subdividing the directional domain (Figure 1 top right). This SD-tree is used to importance sample directions for path construction. The approximation is iteratively improved during rendering, with twice as many samples in each iteration. During rendering, radiance estimates are splatted into the SD-tree at each path vertex. After each iteration, the quadtrees are refined via a simple subdivision scheme: If the relative flux Φ_n/Φ collected by a quadtree node exceeds a threshold ρ (default = 1%), the node is subdivided.

This can improve variance over unguided NEE, but the finite resolution of the learned radiance field usually leaves some remaining variance since not all directions near a high-contributing light source actually hit the light source. To learn useful sampling distributions for guiding, a sufficient number of paths with high contribution needs to be generated by simple BSDF sampling or uniform NEE, which can be a limiting factor in scenes with thousands of very small, high-power light sources.

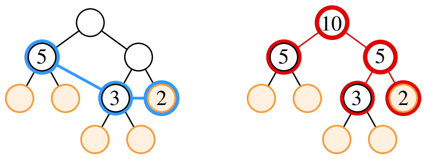

Figure 2: ***Left****: Global light tree with one leaf per emitter (orange), and a weighted tree cut (blue).* ***Right****: In practice we store a weighted sub-tree (red) from the root to the cut.*

Contribution We learn a light selection PMF $P(l|x)$ that depends on where a point x is in the scene, and adapts to power, visibility, orientation, and direction of the light source. We adapt this probability in the cells of a spatial subdivision hierarchy, in particular the 3D binary tree employed by PPG [MGN17].

Akin to Lightcuts we first build a binary tree over all light sources. We represent the direct radiance in a spatial cell as a weighted tree cut (Figure 2 left), with each weight corresponding to the estimated contribution of lights in the cluster. During rendering we accumulate radiance estimates in the cut nodes, and use the stored weights to importance sample nodes in the light tree to guide NEE. The supplemental document gives a pseudocode overview of our approach.

In scenes with many light sources our technique consistently improves upon PPG while requiring at most a similar amount of additional storage, and has little to no memory and performance overhead in scenes with only a handful of lights. The learned tree cut depth automatically adapts to the direct radiance in the scene, and the degree of its refinement can be controlled by a user parameter.

2. Guided Emitter Sampling with Adaptive Tree Cuts

2.1. Direct Radiance Representation

We approximate the incident direct light field discretized over individual light sources. Before rendering starts, we build a binary light subdivision tree with one emitter per leaf, and store a tree cut [VKK18, WFA*05] through the light tree in each cell of the 3D spatial tree. A tree cut is a weighted set of nodes in the binary light tree, where each path from the root to a leaf contains exactly one node in the cut (Figure 2 left). Each cut partitions lights into disjoint clusters, with each node in the cut representing a cluster containing all lights in leaves below it. The weight of a node in the cut approximates the relative total contribution of lights below that node to the spatial cell. These weights guide the sampling of lights or light clusters, similar to how PPG uses values in the directional quadtree to guide direction sampling (Figure 1 right). In order to simplify the sampling and PDF evaluation routines, we store a weighted light sub-tree from the root to the cut (Figure 2 right) instead of only the cut nodes. We call this data structure *Guiding Light Tree* (GLT).

2.2. Learning and Refinement

In order to refine a GLT over rendering iterations, we store two cuts in each cell, using one for guiding and the other to accumulate the contribution of new paths, and swap them after each refinement iteration. We initialize the GLT with the two children of the light tree root node with equal weights, and start the first rendering iteration with 1 sample per pixel. Whenever we construct a complete path $(x_0, \dots, x_L)$ with x_0 on a camera and x_L on a light during rendering, we accumulate the direct radiance estimate $L_e(x_L \to x_{L-1})G(x_L, x_{L-1})/p(x_L)$ at x_{L-1} in the spatial cell containing x_{L-1} in the tree cut node containing the light source from x_L. We apply spatial filtering similar to [Mü19] by jittering the position x_{L-1} within a filter volume (proportional to the size of the spatial cell that holds x_{L-1}) centered around x_{L-1} before locating the cut for depositing the radiance estimate.

After rendering each iteration we refine the tree cuts similar to PPG's quadtree approach. A node is subdivided if its relative collected flux Φ_n/Φ is larger than a threshold σ, and both children receive half the original flux. We recursively apply the subdivision until reaching the threshold or a leaf node of the light tree. While the continuous directional domain can be subdivided indefinitely, the discrete tree cut subdivision stops if a light cluster contains a single light only, even if its relative flux exceeds the threshold σ. Similarly, we recursively merge siblings with a total flux below the threshold. This makes the memory footprint of the cuts depend not only on the value of σ, but also on the number of lights and illumination characteristics of a given scene. If a spatial cell is subdivided by the PPG refinement we copy its cut to both children. After the refinement we discard the current image and start a new iteration with twice as many samples. In the end, we combine the results from the final four iterations as in [MGN17, Mü19].

2.3. Sampling

For sampling at a given path vertex x (Supplemental Algorithm 3), we traverse the spatial tree to find the tree cut approximating the local direct illumination at x. Using a single random number, we stochastically descend the GLT to select a light cluster with a probability proportional to its estimated contribution. From this light cluster, we traverse the global light tree using an overall energy estimate (surface area · radiosity) until we reach an individual emitter. The PMF $P(l|x)$ of a light l is the product of all selection probabilities on the path from the light tree's root node to the light's leaf node. We replace PPG's uniform NEE with our sampling strategy. Since we allow the PMF of light clusters to be zero, our strategy needs to be combined with at least one unbiased sampling strategy, e.g. BSDF sampling.

Each light stores its location in the binary tree as a binary sequence in a 32-bit integer to accelerate locating the light source in the light tree for PMF calculation and radiance depositing. This introduces a memory overhead of 4 bytes per light source, but significantly increases the performance of these operations.

3. Results

We integrated our sampling strategy into the PPG framework [MGN17], replacing uniform NEE with our guided direct illumination sampling (Source code: github.com/nessux/path-guiding-many-lights). All images were rendered on an Intel® Core™ i7 12700K CPU with 32GB of DDR4 RAM.

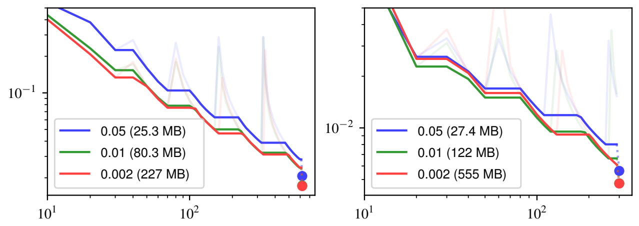

Figure 3: *RMSE over time in the* CITY *(left) and* TORUS LIGHT *(right) scene for different subdivision thresholds σ. The plots show the minimum RMSE of the current and all previous images, with the current RMSE in lower opacity. Peaks/plateaus in the plots result from starting new rendering iterations. The filled circle represents the combined final result. In both plots the final result of $\sigma = 0.01$ is almost identical to $\sigma = 0.002$. The legend also shows the memory footprint of the adaptive tree cuts after the last rendering iteration.*

3.1. Analysis

Subdivision threshold Figure 3 shows how the subdivision threshold σ impacts convergence and memory footprint. Reducing the threshold from 5% to 1% noticeably decreases variance, with no further improvements from a reduction to 0.2%. As the subdivision threshold controls the degree of refinement, the memory footprint of the GLTs increases with smaller σ. At 1% the GLTs require similar memory as PPG's directional quadtrees with the recommended parameters in both scenes. With lower thresholds the memory footprint keeps growing with no further variance reduction.

We thus recommend a default value of $\sigma = 0.01$. An increase of the threshold can be considered to significantly reduce memory usage while moderately increasing variance. Incidentally, Müller et al. [MGN17] also recommend a default value of $\rho = 0.01$ as the threshold for subdividing their directional quadtrees. The threshold barely affects the number of rendered samples per pixel in a given time budget, as the light tree is traversed from the root node to a leaf node for each sampled light either way. A deeper cut only causes more child selection decisions to be based on learned cluster contributions instead of overall energy estimates.

Light Tree We construct the global light tree as a BVH with one leaf per emitter and tested three construction heuristics: a balanced heuristic with equal weight for each emitter, the surface area heuristic (SAH) proportional to the emitter surface area, and a surface area radiance heuristic (SARH) proportional to the SAH multiplied to the emitter's radiosity. We used the SARH for all results in this paper, as its RMSE and memory overhead were a few percent lower than for the balanced or SAH tree, likely due to mostly flatter tree cuts.

3.2. Comparison

In Figure 4 we show equal time comparisons of PPG with our sampling strategy to standard path tracing with uniform NEE (PT+NEE) or bidirectional path tracing (BDPT) depending on which technique performed better in a given scene. We also compare to PPG with uniform NEE or no NEE at all, depending on which technique performed better.

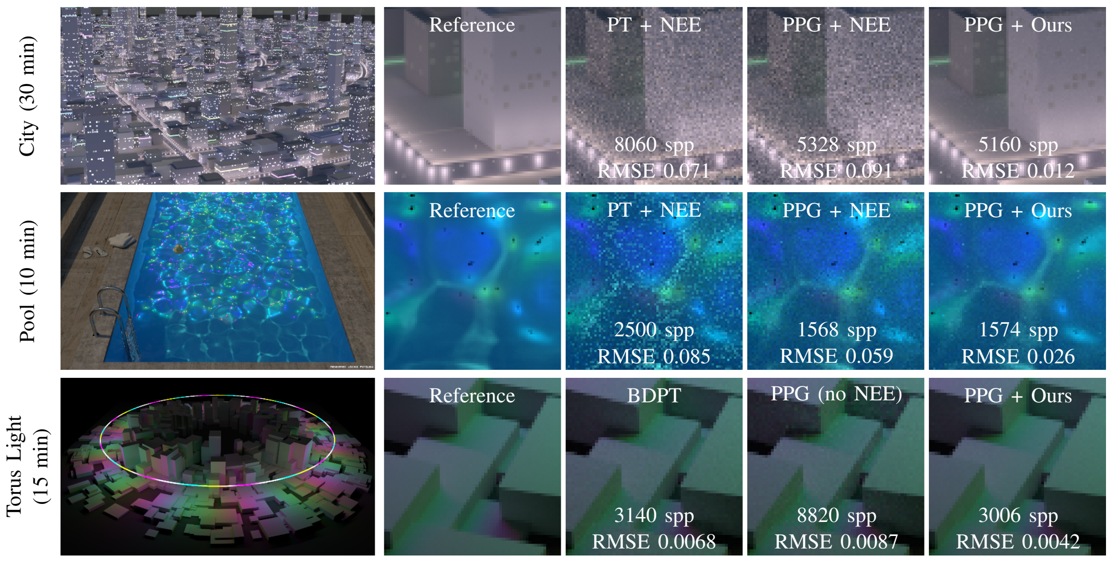

Figure 4: *Equal time comparisons, with the insets and RMSE rendered without directly visible lights to highlight direct/indirect illumination noise. Our technique significantly reduces the error over basic PPG, and outperforms BDPT even without specular interactions.*

The CITY scene contains a simple model of a city under a glass dome with 283924 individual quad and spherical emitters. The overhead of PPG visibly increases noise compared to PT with uniform NEE, especially regarding direct illumination. Our technique reduces the RMSE over PPG with uniform NEE by a factor of 7.6.

The POOL scene contains 1700 small spherical lights near the bottom of a pool with a bumpy, specular water surface. PPG with our adaptive tree cuts performs better than PPG with uniform NEE without introducing noticeable overhead, rendering the same number of samples in the given time budget.

The TORUS LIGHT scene features a mesh light with 24576 primitives, but no SDS light transport. PPG without NEE uses its directional quadtrees to sample direct illumination, but cannot achieve lower error than default BDPT. PPG with our sampling technique performs better than BDPT while remaining unidirectional.

The OCCLUSION scene (Figure 1 left) combines the mesh light from the TORUS LIGHT scene with highly occluded indirect illumination: Light is reflected from a glossy surface to reach a diffuse statue which is visible to the camera. The glossy surface is surrounded by hundreds of small, dark objects occluding both the light and the statue. PPG struggles to learn directional distributions at the statue, resulting in visible noise and fireflies. BDPT also struggles with the occluded indirect illumination but reaches a lower RMSE than PPG. PPG combined with our sampling strategy achieves a 14.6 times lower RMSE than BDPT, indicating that guiding direct illumination can also aid the learning of PPG's directional distributions. Assuming a standard Monte Carlo convergence of $O(1/\sqrt{N})$, a 14.6 times lower RMSE represents a 213 times speedup.

In the SPACESHIP scene (no figure) by [MGN17] with only 5 emitters our technique performs similar to PPG: At a memory overhead of 3.5MB, we achieve 2408spp and an RMSE of 0.0101 in 5 minutes, compared to PPG with 2632spp and an RMSE of 0.0106.

4. Conclusion

We presented an unbiased light selection strategy to guide direct illumination that significantly decreased the variance of PPG in scenes with a large number of light sources and difficult visibility. Memory consumption is similar to PPG's quadtrees and can be controlled by a user parameter. A default value of $\sigma = 0.01$ performed well in all our tests. Depending only on an iterative rendering process and an existing spatial subdivision structure, our adaptive light clustering could also be applied in rendering frameworks other than PPG. Performance could be further improved by incorporating results of previous work regarding the construction of light hierarchies [EK18] or optimal light selection probabilities [VKK18].

References

[BWP*20] BITTERLI B., WYMAN C., PHARR M., SHIRLEY P., LEFOHN A., JAROSZ W.: Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Transactions on Graphics 39* (7 2020). 2

[EK18] ESTEVEZ A. C., KULLA C.: Importance sampling of many lights with adaptive tree splitting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 1* (8 2018), 1–17. 2, 4

[MGN17] MÜLLER T., GROSS M., NOVÁK J.: Practical path guiding for efficient light-transport simulation. *Computer Graphics Forum 36* (6 2017), 91–100. 2, 3, 4

[Mü19] MÜLLER T.: "Practical Path Guiding" in production. ACM, p. 18:35–18:48. 2, 3

[Pan19] PANTALEONI J.: Importance sampling of many lights with reinforcement lightcuts learning, 2019. 2

[VKK18] VÉVODA P., KONDAPANENI I., KŘIVÁNEK J.: Bayesian online regression for adaptive direct illumination sampling. *ACM Transactions on Graphics 37* (2018). 2, 4

[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* (7 2005), 1098–1107. 2