Probabilistic Connections for Bidirectional Path Tracing

Stefan Popov, Ravi Ramamoorthi, Fredo Durand, George Drettakis

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

Volume 34 (2015), Number 4

Probabilistic Connections for Bidirectional Path Tracing

Stefan Popov [1] Ravi Ramamoorthi [2] Fredo Durand [3] George Drettakis [1]

[1] Inria [2] UC San Diego [3] MIT CSAIL

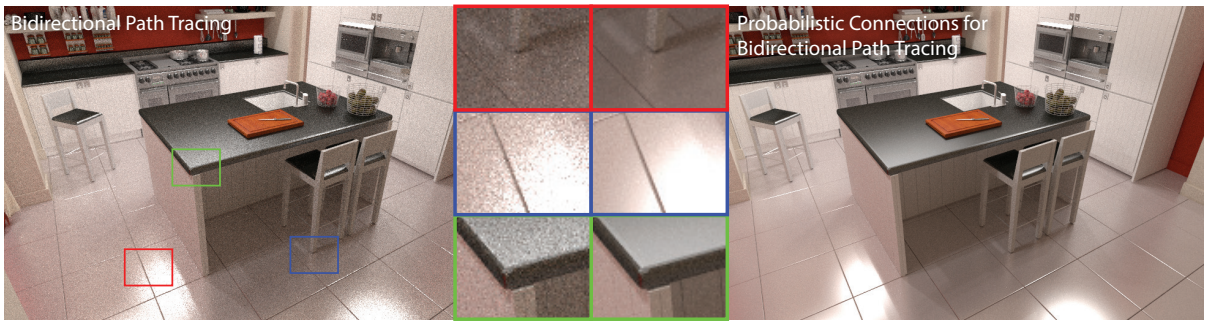


Figure 1: *Our Probabilistic Connections for Bidirectional Path Tracing approach importance samples connections to an eye sub-path, and greatly reduces variance, by considering and reusing multiple light sub-paths at once. Our approach (right) achieves much higher quality than bidirectional path-tracing on the left for the same computation time (~8.4 min).*

.

Abstract

Bidirectional path tracing (BDPT) with Multiple Importance Sampling is one of the most versatile unbiased rendering algorithms today. BDPT repeatedly generates sub-paths from the eye and the lights, which are connected for each pixel and then discarded. Unfortunately, many such bidirectional connections turn out to have low contribution to the solution. Our key observation is that we can importance sample connections to an eye sub-path by considering multiple light sub-paths at once and creating connections probabilistically. We do this by storing light paths, and estimating probability mass functions of the discrete set of possible connections to all light paths. This has two key advantages: we efficiently create connections with low variance by Monte Carlo sampling, and we reuse light paths across different eye paths. We also introduce a caching scheme by deriving an approximation to sub-path contribution which avoids high-dimensional path distance computations. Our approach builds on caching methods developed in the different context of VPLs. Our Probabilistic Connections for Bidirectional Path Tracing approach raises a major challenge, since reuse results in high variance due to correlation between paths. We analyze the problem of path correlation and derive a conservative upper bound of the variance, with computationally tractable sample weights. We present results of our method which shows significant improvement over previous unbiased global illumination methods, and evaluate our algorithmic choices.

1. Introduction

Bidirectional path tracing (BDPT) [VG95a, LW93] with Multiple Importance Sampling (MIS) [VG95b] is one of the most versatile rendering algorithms today. BDPT repeatedly builds an eye and a light sub-path, connects their vertices, computes MIS weights to estimate the contribution to the pixel and then discards the paths. In all path tracing algorithms, as the probability of the path gets closer to being proportional to its throughput, variance is reduced.

In bidirectional path tracing, while the eye and light sub-paths are usually well importance sampled, the connection itself unfortunately is not. For example, the connection segment has a direction that has no special reason to be well aligned with the lobes of the BSDFs at the two connected vertices.

Our key observation is that we can importance sample connections to an eye sub-path by considering multiple light sub-paths at once and creating connections probabilistically. We do this by estimating probability mass functions (PMF) of the discrete set of possible connections to all light paths. This has two key advantages: we create connections

with contributions that reduce variance more quickly, and we reuse light paths across different eye paths. However, it raises a major challenge, since reuse makes paths correlated, resulting in high variance if multiple importance sampling is applied directly. Note that when combining paths in BDPT, connections are deterministic for two special path types: paths traced from the camera where the last vertex of the path lies directly on a light source (unidirectional path tracing (UPT) [Kaj86]) and paths traced from the light with the last vertex directly connecting to a pinhole camera (light tracing (LT) [DW95]). We call all other types *inner paths*: our approach only improves these.

A large body of previous work exists on estimating probability density functions (PDFs) for importance sampling (e.g., [Jen95, HP02, VKv*14]) which intuitively guides the choice of the direction of the next ray at a path vertex. Our work is complementary to this, since we importance sample the *connections*, improving the overall result. We demonstrate this complementarity by using the method of [HP02] during the generation of initial paths for our algorithm in Sect. 7.

Even though connections are not importance sampled in BDPT, Veach [Vea98] did sample visibility tests between one eye path and one light path using Russian roulette to optimize performance. In contrast, we introduce the first approach that importance-samples connections to a set of presampled light paths in the context of unbiased BDPT algorithms, and thus probabilistically connects to all of them. To sample sub-path connections and compute corresponding PMFs, we consider the set of contributions from all M light paths to a given eye vertex. Specifically, we define a Monte Carlo estimator of the sum of all these contributions, since directly computing it would add the cost of M additional connections per eye vertex. To importance sample this estimator, we define a PMF over the discrete set of light paths.

Computing this PMF at every vertex would also incur M additional connection calculations. We introduce an efficient caching scheme instead. At a first glance, it may seem that this requires the computation of distances between paths, which is a high-dimensional problem. We show that it is possible to introduce a low-dimensional approximation, resulting in a fast solution.

We present three main contributions:

- The Probabilistic Connections for Bidirectional Path Tracing (PCBPT) algorithm, which extends Georgiev et al's [GKPS12] idea of importance sampling connections to unbiased bidirectional rendering (Sect. 4).
- An approximation to sub-path contributions avoiding high-dimensional path comparisons, allowing the use of an efficient PMF caching scheme (Sect. 4.2).
- An analysis of the effect of path correlation on the MIS variance reduction technique. Based on our analysis, we derive a conservative upper bound of the variance, with computationally tractable sample weights (Sect. 5).

Our approach can drastically reduce the noise in inner paths: compared to BDPT our method reduces this error up to 6.4 times, with an average of 3x on our scenes. For the images we show in our results (Fig. 1,10,11,12,13) PCBPT is up to 3.4 times faster than BDPT (average 2.5) for the computation of *all paths*, for the same quality.

2. Related Work

Our method builds on BDPT [VG95a, LW93, Vea98]. We generalize the way paths are built and we introduce importance sampling of vertex connections. We also derive new MIS weights that work in the presence of significant correlation, inspired by the methodology in Veach's work [Vea98, p.288], [VG95b].

Connecting each pair of eye and light vertices in BDPT has been attempted before: namely in Combinatorial BDPT [PBPP11] and Bidirectional Light Cuts (BDLC) [WKB12]. In the former case, the connections were off-loaded to the GPU; the issue of path correlation was not treated. This resulted in faster execution, but did not improve the efficiency of the algorithm itself. Similar to our approach, the method of Walter et al. [WKB12] reduces the number of computed connections, by employing a spatial data structure over the last vertices of the sub-paths and by computing conservative bounds on the contribution of the tree's nodes. The resulting solution is biased, since as noted by Walter et al. [2012], the weights for bidirectional lightcuts are explicitly chosen to control the tradeoff between bias vs. cost. GPU acceleration was also used to accelerate connections in [VA11]. The efficiency optimized Russian roulette (EORR) approach in Sect. 10.4 of Veach's Ph.D. thesis [Vea98], implicitly varies the number of connections actually established by pruning the number of rays based on statistical efficiency of each sample, reducing the number of visibility tests. This is no longer the main source of computational cost in modern Monte Carlo ray-tracers; we show this in Sect. 7. Multi-light or Virtual Point Light (VPL) methods, such as Instant Radiosity (IR) [Kel97], or row-column methods [HPB07] connect each eye vertex to a subset of light vertices. However, they are inherently less powerful than BDPT and hence our approach, since they build each path in only one way, have weak singularities [KK06], have typical difficulties in handling glossy reflections, and no support for caustics.

In the different context of volumetric rendering, sequences of vertices [GKH*13] are sampled from a joint distribution and continuous family of VPLs [NNDJ12] are used to simulate indirect illumination in participating media. These approaches are related to ours, in that they stochastically select the right shadow connection over a continuous family of VPL or path vertex locations.

In recent work Vorba et al. [VKv*14] use Gaussian Mixture Models (GMM) to learn distributions based on light and

	Path-related quantities	
Symbol	Quantity	Sec/Eq
M	# light paths in 1 iteration	1,3
K	# connections/eye path	4.1
$\bar{x} = x_0\ x_1\ x_2 \ldots x_{l-1}$	Subpath of length l	3
$\bar{z} = z_0 \rightarrow \ldots \rightarrow z_{t-1}$	Eye subpath of length t	3.1
$\bar{y} = y_0 \rightarrow \ldots \rightarrow y_{s-1}$	Light subpath of length s	3.1
$p(\bar{x})$	PDF for path sampling	3 (5)
$\bar{X}_j$	Path sample drawn from $p(\bar{x})$	3 (5)
i	strategy i; path with $t = i$	3.1
$\bar{X}_{ij}$	jth sample of strategy $i \in 0 \ldots l$	3.1
$p_i^{conn}(j)$	Conn. PMF for strategy i for light path j	4.1
z^c	eye vertex with imp. record	6

	Radiance Estimates	
Symbol	Quantity	Sec/Eq
I	Radiance at a pixel	3
I_l	$l-1$'st term Eq 1; radiance for path len. l	3
$\tilde{I}_l$	PCBPT MC estimate of I_l	3.1 (7)
$I(i,j)$	Contrib. of light path j to strategy i	3.1 (8)
I_i	Contrib. of M light paths to strategy i	4.1 (9)
$\tilde{I}_i$	PCBPT MC estimate of I_i	4.1 (10)

Table 1: *List of symbols*

importance particle tracing. This method could be combined with our PCBPT similar to Hey and Purgathofer [HP02], which we discussed in Sect. 1. We show a comparison to this approach in Sect. 7.

Several methods sample connections in Instant Radiosity (IR), including bidirectional IR [SIMP06], Lightcuts [WFA*05], and Importance Caching [GKPS12]. In all cases, they rely on intrinsic properties of IR, such as a constant eye sub-path length of 2 (e.g. to compute the camera ray footprint in [GKPS12]) and only a single strategy that samples paths.

Caching of sampling distributions has been used before, for importance sampling [BRDC12] and in the context of VPLs [GKPS12]. Our caching strategy was inspired by the latter and has similar implementation, but operates in a BDPT context requiring care in its justification.

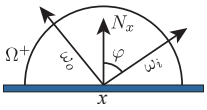
Finally, the strengths of Photon mapping [Jen01] (PM) have been successfully combined with BDPT in the Vertex Connection Merging (VCM) framework [GKDS12] and in the path space extension algorithm (PEA) [HPJ12], showing that BDPT and PM can be merged successfully, albeit in a consistently biased algorithm. Even though we focus on unbiased results here, PCBPT could be merged with either approach in the future.

3. Background on BDPT and MIS

Global illumination methods compute the measurement equation, given by Veach [Vea98, Chap. 3], for the radiance I:

$$I = \langle W, L_e \rangle + \langle W, TL_e \rangle + \left\langle W, T^2 L_e \right\rangle + \ldots \qquad (1)$$

where x is as shown in the inset, $W(x \leftarrow \omega_i)$ is the importance function (in the sense of Veach [Vea98, Chap. 3]), and $L_e(x \rightarrow \omega_o)$ is the emittance. If we denote with $h(x,\omega)$ the first intersection of the ray (x,ω) with the scene, with N_x the normal at surface point x, and with ρ the bidirectional scattering distribution function (BSDF), then the transport operator T over a function ϕ is defined as:



$$(T\phi)(x \rightarrow \omega_o) = \int_\Omega \rho(\omega_i \rightarrow x \rightarrow \omega_o)\phi(h(x,\omega_i) \rightarrow -\omega_i)\langle N_x, \omega_i \rangle d\omega_i$$

To solve Eq. (1), we rewrite it using the surface area form. For convenience, we denote its $(l-1)$th term with I_l and its contribution is given by Veach [Vea98, p.223]:

$$I_l = \left\langle W, T^{l-2} L_e \right\rangle = \int_{\mathcal{M}^l} f(\bar{x})\, d\mu(\bar{x}) \qquad (2)$$

$$f(\bar{x}) = L_e(x_0 \rightarrow x_1)\ \Pi_l(\bar{x})\ GV(x_{l-2} \leftrightarrow x_{l-1}) W(x_{l-2} \rightarrow x_{l-1}) \qquad (3)$$

$$\Pi_l(\bar{x}) = \prod_{i=0}^{l-3} GV(x_i \leftrightarrow x_{i+1}) \rho(x_i \rightarrow x_{i+1} \rightarrow x_{i+2}) \qquad (4)$$

where $\mathcal{M}$ denotes the surface of the scene (and $\mathcal{M}^l$ is the l-dimensional Cartesian product over surfaces), $\bar{x} = x_0\ x_1\ \ldots\ x_{l-1}$, $GV(x \leftrightarrow y)$ is the combined geometry and visibility terms between x and y, and $d\mu(\bar{x}) = dx_0 \ldots dx_{l-1}$.

We can estimate the value of each I_l using Monte Carlo integration in an unbiased way, by generating N samples of dimension l (i.e., paths of length l) $\{\bar{X}_j\}, j \in \{1 \ldots N\}$, from some distribution $p(\bar{x})$:

$$\frac{1}{N} \sum_{j=1}^{N} \frac{f(\bar{X}_j)}{p(\bar{X}_j)} \approx I_l \qquad (5)$$

We include a list of symbols in Table 1.

3.1. Multiple Importance Sampling

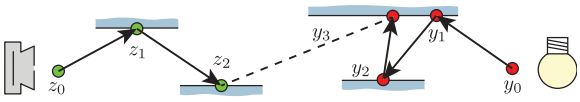


Figure 2: *A path with eye-sub-path length $t = 3$ and light sub-path length $s = 4$, built by connecting the eye sub-path $z_0 \rightarrow \ldots \rightarrow z_2$ to the light sub-path $y_0 \rightarrow \ldots \rightarrow y_3$.*

The $l-$dimensional samples for Eq. (5) are typically generated through local path sampling: First generate a sub-path $\bar{z} = z_0 \rightarrow \ldots \rightarrow z_{t-1}$ from the camera with length t, generate a sub-path $\bar{y} = y_0 \rightarrow \ldots \rightarrow y_{s-1}$ from the light source with length s, and connect z_{t-1} to y_{s-1} (see Fig. 2). This way, we obtain a path $\bar{x}$ with $l = s + t$ vertices and probability $p(\bar{x}) = p(\bar{y})p(\bar{z})$.

There are $l+1$ different ways, or *strategies*, to generate

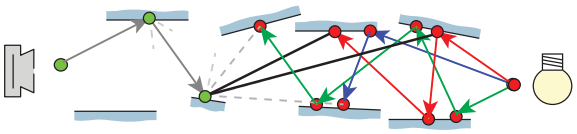


Figure 3: *We probabilistically sample a small number of contributions (solid black lines) from the set of all light paths (dashed gray lines). Some connections are incomplete for clarity of illustration.*

the same path: $(s=0,t=l),\ldots,(s=l,t=0)$. The case $s=0$ corresponds to unidirectional path tracing, $s=1$ to path tracing with next event estimation, and $t=1$ to light tracing. The different strategies have different PDFs and sample any small region $\mathcal{B} \in \mathcal{M}^l$ with different densities. Intuitively, the more samples a strategy generates in $\mathcal{B}$, the better it "explores" it and the more we can trust its Monte Carlo estimate there. Thus, we need a mechanism to locally "favor" strategies that are denser. Unless noted otherwise, our analysis is presented for a fixed path length of l for clarity.

The power of BDPT lies in its ability to reduce variance by combining samples from different strategies. To do so, it relies on a linear multi-sample model: Assume $l+1$ strategies are used to estimate a pixel's value I_l. n_i samples $\{\bar{X}_{ij}\}, j \in 1\ldots n_i$ are taken from each strategy $i \in 0,\ldots,l$, distributed according to the strategy's PDF $p_i(\bar{x})$. Weights $w_i(\bar{X}_{ij})$ can be assigned to the samples. As long as the weights for a particular sample sum up to 1 (i.e., $\sum_i w_i(\bar{x}) = 1$), and $w_i(\bar{x}) = 0$ whenever $p_i(\bar{x}) = 0$, $\tilde{I}_l$ defined below can be used as an unbiased estimate of I_l:

$$\tilde{I}_l = \sum_{i=0}^{l} \frac{1}{n_i} \sum_{j=1}^{n_i} \frac{w_i(\bar{X}_{ij}) f(\bar{X}_{ij})}{p_i(\bar{X}_{ij})} \tag{6}$$

In the above i denotes the strategy and we choose i to be equal to t in the context of BDPT; for example, for paths of length $l=7$, strategy $i=3$ will be the strategy with eye sub-paths of length 3, or $t=3$ and $s=4$ (see Fig. 2).

The advantage of introducing weights is that variance can be reduced by combining the samples (or paths) *after* they have been sampled. In general, weights that minimize the error should be chosen. Since BDPT is unbiased, the error is only due to variance. If the samples are independent, the variance can be reasonably reduced by using constrained vector minimization [Vea98, p.288], resulting in the balance heuristic.

4. Probabilistic Connections

The main motivation of our approach is to importance sample connections and amortize the cost of tracing sub-paths. BDPT discards each pair of sub-paths immediately after connecting them. In contrast, we first generate and store a set of light sub-paths and we probabilistically connect each eye sub-path to all of them (Fig. 3).

Algorithm 1 PCBPT Algorithm

```
1:  // Initialize
2:  Generate light paths; generate eye-paths
3:  Evaluate light-tracing and path-tracing contributions
4:
5:  // PMF caching
6:  Pick a set of eye paths uniformly distributed in the image
7:  for each selected eye path z̄ do
8:      for each vertex z_i^c of the chosen path do
9:          create PMF
10:         Cache importance record at vertex z_i^c
11:
12: // Do actual PCBPT
13: for each eye-path z̄ (pixel) do
14:     for each vertex z_i of z̄ do
15:         find closest importance records
16:         interpolate PMF
17:         sample interpolated PMF
18:         add resulting path to contribution
```

For each pixel we generate an eye sub-path (in solid gray in Fig. 3). At each vertex of this eye sub-path, we probabilistically create a small number of connections (compared to the total number of light paths) through importance sampling as shown schematically in the figure. Our probabilistic sampling is over a discrete set of light path contributions; its *discrete* nature, and the reuse of light paths are key elements in making our algorithm efficient.

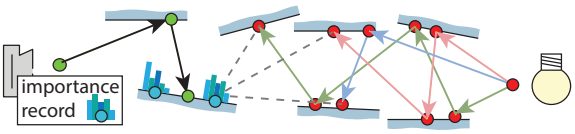


Figure 4: *We store PMFs for a small subset of eye sub-paths in* importance records. *Every other eye sub-path interpolates its PMF from the closest records.*

Computing the probability mass functions (PMFs) to sample light subpaths at each eye vertex incurs M additional computations per eye vertex. To avoid this additional cost, we use a caching scheme similar to [GKPS12], storing accurate PMFs in a small set of *importance records*, shown as cyan circles in Fig. 4. In the figure the final green vertex of the eye sub-path uses the closest importance records to create a PMF which it then uses to sample the connections to the light sub-paths in the scene.

The PCBPT algorithm iteratively generates paths which are then connected probabilistically. We generate a new set of eye (and light) paths at each iteration. We summarize one iteration of the PCBPT algorithm in Algorithm 1.

We execute Algorithm 1 iteratively, generating one sample per pixel at each iteration. Performing n iterations is

equivalent to using n samples per pixel. Lines 1-3 are standard for BDPT, except that we store M light paths; lines 5-18 are described in Sects. 4.1 and 4.2 below.

4.1. Connection Probabilities

For a given total path length l, each strategy i corresponds to a prefix of the eye sub-path of length i (Fig. 5). This prefix gets connected to many light paths, and the contributions are then averaged over time (or, equivalently iterations).

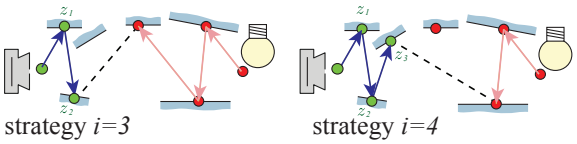


Figure 5: *For a fixed path length $l = 7$, strategy $i = 3$ corresponds to connections at eye vertex 2 (left) and similarly for $i = 4$ (right).*

We define $I(i,j)$, which is the contribution of light path j to strategy (or eye sub-path vertex) i, as:

$$I(i,j) = \frac{w_i(\bar{X}_{ij}) f(\bar{X}_{ij})}{n_i p_i(\bar{X}_{ij})} \tag{7}$$

and, for a given path length l, we rewrite the MIS estimator of the contribution $\tilde{I}_l$ (Eq. (6)), recalling that M is the number of light sub-paths:

$$\tilde{I}_l = \sum_{i=0}^{l} \sum_{j=1}^{M} I(i,j) \tag{8}$$

For convenience, we also define: $I_i = \sum_{j=1}^{M} I(i,j)$, which is the contribution of all light paths for a given strategy/vertex. We have $\tilde{I}_l = \sum_{i=0}^{l} I_i$. We illustrate these quantities in Fig. 6.

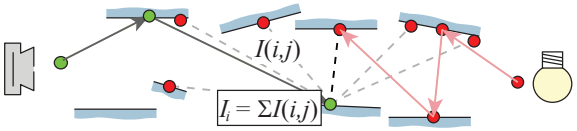


Figure 6: *The contribution of a single connection is $I(i,j)$ for strategy i (i.e., eye sub-path lengths of $i = 3$ here) and light sub-path j (red). I_i is the* sum *of all such contributions, which we estimate probabilistically.*

Computing I_i directly would require M operations (i.e., the number of stored light paths); instead we perform Monte Carlo sampling to estimate the sum I_i, similar to [GKPS12]. We do this by defining a probability mass function (PMF) $p_i^{\text{conn}}(j)$ for strategy i and light path j over the discrete set of M light paths. We draw $K \ll M$ independent and identically distributed (IID) samples $\xi_1 \ldots \xi_k \in [1 \ldots M]$ for the index j in the sum. The quantity $\tilde{I}_i$ defined below is an unbiased estimator for I_i:

$$\tilde{I}_i = \frac{1}{K} \sum_{k=1}^{K} \frac{I(i,\xi_k)}{p_i^{\text{conn}}(\xi_k)} \tag{9}$$

The fact that our estimator is unbiased can be easily demonstrated using standard Monte Carlo techniques (see supplemental material).

Since $p_i^{\text{conn}}(\bar{x})$ is discrete, we can compute each $I(i,j)$, then take $p_i^{\text{conn}}(j) = I(i,j)/\sum_j I(i,j)$, and finally use CDF inversion to sample from it. This will result in perfect importance sampling of the discrete set of paths. But doing it for each light path will be at least as expensive as computing I_i deterministically and we make it efficient by exploiting locality, in the sense of distance between eye paths.

4.2. PMF caching

To avoid the cost of computing the PMF at each eye vertex, we cache the PMFs at a small set of eye paths and then interpolate the PMF for other eye vertices. We show next that this is possible and that the PMF of a sub-path is a function of its last vertex's position and incoming direction. Consequently, we can cache and look up the PMFs based on this vertex's position, in a fashion similar to [GKPS12], but with added importance record weights that account for the proximity of incoming directions. The exact expression of the weights and the interpolation are given in supplemental material.

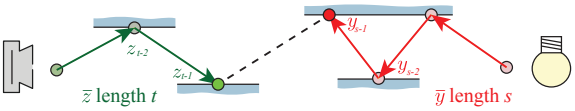


Figure 7: *Eye/light path configuration.*

Consider the configuration shown in Fig. 7. We denote with $\bar{y}$ the j-th light sub-path and with $\bar{z}$ the i-th eye sub-path, where s and t are their respective lengths in vertices. Then $I(i,j)$ becomes:

$$I(i,j) = w_i(\bar{y}\bar{z})\, \frac{f(\bar{y})}{p(\bar{y})}\, f_{\text{conn}}(\bar{y},\bar{z})\, \frac{f(\bar{z})}{p(\bar{z})} \tag{10}$$

$$\begin{aligned} f_{\text{conn}}(\bar{y},\bar{z}) = {} & \rho(y_{s-2} \to y_{s-1} \to z_{t-1}) \\ & G V(y_{s-1} \leftrightarrow z_{t-1}) \\ & \rho(y_{s-1} \to z_{t-1} \to z_{t-2}) \end{aligned} \tag{11}$$

While constructing our PMF, we ignore the $w_i(\bar{y}\bar{z})$ term; ignoring a term in importance sampling is common practice [Vea98, p. 48]; These MIS weights are of course used for the final unbiased calculation of the path's contribution (i.e., $I(i,\xi_k)$). We have M light subpaths: $\bar{y}^{(1)}, \bar{y}^{(2)}, \ldots \bar{y}^{(M)}$. For each eye subpath we have a corresponding set of potential full paths $\bar{x}^{(1)}, \bar{x}^{(2)}, \ldots \bar{x}^{(M)}$. After ignoring the MIS

weights, the perfect PMF for the rest of the terms for a given path $\bar{x}^{(j)}$ is given by:

$$p^{\text{conn}}(j) = \frac{\frac{f(\bar{x}^{(j)})}{p(\bar{x}^{(j)})}}{\sum_{k=0}^{M} \frac{f(\bar{x}^{(k)})}{p(\bar{x}^{(k)})}} = \frac{\frac{f(\bar{z})\rho GV\rho f(\bar{y}^{(j)})}{p(\bar{z})p(\bar{y}^{(j)})}}{\frac{f(\bar{z})}{p(\bar{z})}\sum_{k=0}^{M} \frac{\rho GV\rho f(\bar{y}^{(k)})}{p(\bar{y}^{(k)})}} \tag{12}$$

by substituting from Eq. (10) above. We see that $f(\bar{z})/p(\bar{z})$ cancels out, and the remaining expression is a function of the position $p_z = z_{t-1}$ of the last eye path vertex, the incoming direction $\omega_z = z_{t-2} \to z_{t-1}$ at p_z, and the set $\mathcal{S}_{\mathcal{L}}$ of all light path vertices $\mathcal{S}_{\mathcal{L}} = \{y_{s-1}^{(1)}, \ldots y_0^{(1)} \ldots y_{s-1}^{(M)}, \ldots y_0^{(M)}\}$. However, the light paths are the same for all eye vertices, and the only variables that the PMF depends on are the position and incoming direction of the last eye path vertex. This shows that it is possible to avoid the high-dimensional path distance computation, and simply interpolate based on the much lower dimensional position and direction. Note that this demonstration holds both for BSDF sampling, and for more sophisticated solutions such as that of Hey and Purgathofer [HP02].

5. Multiple Importance Sampling with Correlations

5.1. Minimizing Variance

Since samples in PCBPT are not independent, their covariance is not zero, which leads to an additional term in the variance of the estimate. The standard MIS weights therefore no longer provide the fine error guarantees from [Vea98, Chap. 9]. The variance of the pixel estimator in Eq. (6) is given by:

$$\begin{aligned} V[\tilde{I}_l] =& V\left[\sum_i \frac{1}{n_i}\sum_j F_{ij}\right] \\ =& \sum_{i_1}\sum_{i_2}\sum_{j_1}\sum_{j_2} \frac{1}{n_{i_1} n_{i_2}} \text{COV}[F_{i_1 j_1}, F_{i_2 j_2}] \end{aligned} \tag{13}$$

where $F_{ij} = w_i(\bar{X}_{ij}) f(\bar{X}_{ij}) / p_i(\bar{X}_{ij})$

We can assume that two eye paths with different lengths are independent. While this is not necessarily true in practice, the effect of the introduced correlation on the variance is negligible [Vea98, p.307] and $\text{COV}[F_{i_1 j_1}, F_{i_2 j_2}] \approx 0$ whenever $i_1 \neq i_2$. Also, for a fixed i the variables $\{F_{i,j} | j = 1 \ldots n_i\}$ are identically distributed but not independent. Thus $\text{COV}[F_{i j_1}, F_{i j_2}] = COV[F_{i1}, F_{i2}]$ and $V[F_{ij}] = V[F_{i1}]$ for two given paths $i1$ and $i2$ (e.g., the first two). Substituting into Eq. (13) we obtain:

$$V[\tilde{I}_l] = \sum_{i=0}^{l} \left(\frac{1}{n_i} V[F_{i1}] + \frac{n_i - 1}{n_i} \text{COV}[F_{i1}, F_{i2}] \right) \tag{14}$$

To maintain the benefits of MIS, we need to find functions w_i that minimize the functional $V[\tilde{I}_l]$. This involves solving a constrained minimization problem in function space, and bears similarity to how MIS was derived [Vea98]. The full derivation is provided in supplemental material. The final equations for the weights contain rational expressions over integrals recursively nested in integrals of other rational expressions. Given this complexity, it would be less expensive to compute global illumination than the solution to these equations; we instead minimize an upper bound of the variance.

5.2. Upper Bound

We adopt a conservative solution, which while not optimal has computationally tractable weights. As a consequence of the Cauchy-Schwarz inequality $\text{COV}[F_{i1}, F_{i2}]^2 \leq V[F_{i1}]^2$ and thus $\text{COV}[F_{i1}, F_{i2}] \leq |\text{COV}[F_{i1}, F_{i2}]| \leq V[F_{i1}]$. We use this fact to derive an upper bound for Eq. (14):

$$V[\tilde{I}_l] \leq \sum_{i \in S_u} \frac{1}{n_i} V[F_{i1}] + \sum_{i \in S_c} V[F_{i1}] \tag{15}$$

where S_u is the set of uncorrelated strategies, that is light tracing with $t = 1$ and unidirectional path tracing with $s = 0$, and S_c is the set of correlated ones, i.e., inner paths.

The variance of the estimator in Eq. (6) is equal to the right hand side of Eq. (15), guaranteeing that the error will not exceed that of BDPT including MIS.

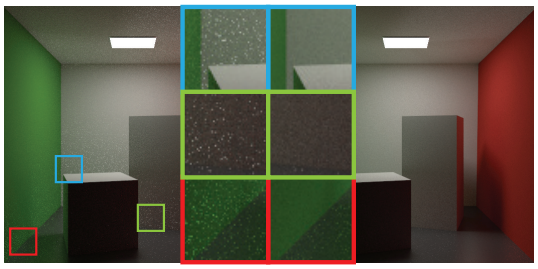


Figure 8: *Evaluation of MIS upper bound weights. Left: results with standard MIS weights; Right: same scene computed with our upper bound MIS weights for correlated paths as proposed in Sect. 5. Noise is reduced significantly with our improved weights.*

To minimize Eq. (15), we use the same technique as Veach [Vea98, Appendix 9.A]. Specifically, the same proof can be used to demonstrate that the upper bound is minimized if we use the following weights:

$$w_i(\bar{x}) = \frac{n_i p_i(\bar{x})}{\sum_{k \in S_u} n_k p_k(\bar{x}) + \sum_{k \in S_c} p_k(\bar{x})} \quad , i \in S_u \tag{16}$$

$$w_i(\bar{x}) = \frac{p_i(\bar{x})}{\sum_{k \in S_u} n_k p_k(\bar{x}) + \sum_{k \in S_c} p_k(\bar{x})} \quad , i \in S_c \tag{17}$$

where $p_i(\bar{x})$ is the probability of path $\bar{x}$ for strategy i, n_i is the number of samples taken from strategy i, and $\bar{z}$ is the eye sub-path. For the uncorrelated samples S_u, the weights are the same as for traditional MIS weights [Vea98], while

for the correlated samples S_c, the factor n_i is not present in the numerator. In practice this means that lower weight is given to correlated samples. Furthermore, we also use the same arguments as in Veach [Vea98] to use the power/max heuristics instead of the balance heuristic. The use of the upper bound can reduce noise significantly, as we can see in Fig. 8, and also in Fig. 17.

6. Implementation

At each iteration we first cast $\mathrm{W} \times \mathrm{H}$ eye paths and light paths (where W, H are the width and height of the image) for the light and unidirectional path tracing (line 1 in Algorithm 1). We then follow standard practice for BDPT, and sample the light sources with eye paths (unidirectional path tracing) and connect light paths to the camera (light tracing) (line 2). These paths are uncorrelated, and are treated with standard methods.

We store the first M light sub-paths, where $M = 100$ in all our tests. These paths include the visibility computation. We then choose K, i.e., the number of connections actually made depending on the scene. We discuss and evaluate the trade-offs of these parameters in Sect. 7.2.

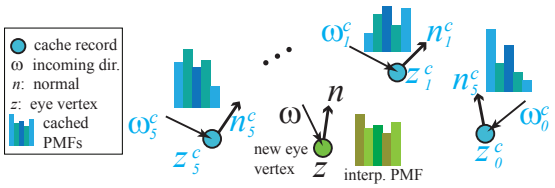


Figure 9: *PMF interpolation: The PMF at the new eye vertex z (green) is interpolated from the cached records (cyan), weighted by the cosines between incoming directions* ω *and normals n.*

We create $0.004 \times \mathrm{W} \times \mathrm{H}$ eye paths for caching, using low discrepancy sampling to uniformly cover the screen. At each eye vertex of these paths we create an importance record and store the complete PMF and the cumulative mass function (CMF) for all M light paths (lines 5-10). Furthermore, we consider the sum of contributions of all light path lengths and compute the PMF over the members of this sum. The PMF and CMF are floating point arrays of size V – the total number of vertices in the M paths. For each eye vertex z_i^c which holds an importance record, we store $\mathrm{PMF}(z_i^c, v)$ and $\mathrm{CMF}(z_i^c, v)$, $v = 0 \ldots V-1$. During PCBPT, for an eye vertex z_i we lookup the 6 nearest cache points z_q^c, $q = 0 \ldots 5$ using a KD-tree on vertex positions and perform a weighted interpolation of the cached mass functions (lines 15-18). We blend with a uniform distribution to avoid bias or very low probability samples which would occur if all importance records contain a light vertex with a very small or zero probability [GKPS12]. The weights are the product of the cosines of the angles of incoming directions and the normals at z_i and z_q^c, times the distance between z_i and z_q^c, normalized over the distances to all selected cached points. While theoretically the entire 2D angular domain could be considered, using only the cosine proved sufficient in our tests, even for the highly glossy Plants scene (see below). We perform inversion sampling of the CMF by binary search, using lazy evaluation of the interpolated CMF, and then interpolate the value of the PMF to find p_i^{conn}. The process is illustrated in the Fig. 9, and details are given in supplemental material. We implemented our approach in our in-house rendering software in C++, using the Intel Embree [WFWB13] ray-tracer. All timings are reported on a 12-core PC with the Intel Xeon E5-2630 at 2.3Ghz.

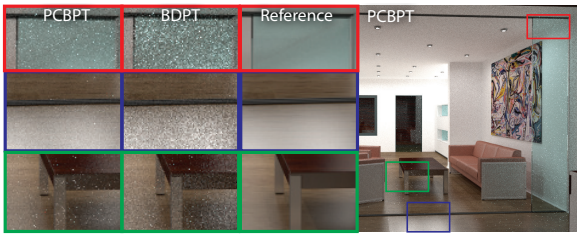


Figure 10: *Same time comparisons for Apartment scene. Right: PCBPT image. Inset results for PCBPT, BDPT, and reference. Rendering time is 12min; PCBPT and BDPT performed 200 iterations each.*

7. Results and Comparisons

In this section we present results of our algorithm, comparison to other methods and evaluation of different parameters. We show results of our method on four scenes: Apartment (Fig. 10), Living Room from Vorba et al. [VKv*14] (Fig. 11), Kitchen (Fig. 1) and Plants (Fig. 12). All images in this section are also provided in the supplemental material. We chose to present images where noise levels have dropped significantly, to show the comparative benefit of our approach before convergence.

For Kitchen the number of actual connections is $K = 2$, for Apartment $K = 1$, for Living Room $K = 10$ and for Plants $K = 2$. We discuss the effect of the choice of K in Sect. 7.2. The image resolutions are $\mathrm{W} = 820, \mathrm{H} = 512$ in the Apartment, Kitchen and Plant scenes, $\mathrm{W} = 910, \mathrm{H} = 512$ in Living Room. Image resolution determines the number of cached records as explained in the implementation (Sect. 6). The memory consumption of the PMFs does not exceed 50 MB for any of these scenes. All reference images can be found in supplemental material; we include insets from the reference in our figures of results. The breakdown of computation for a given iteration is shown in Table 2. BDPT performs the same first two steps for path generation and light/path tracing (lines 1-2 in Algorithm 1); the connections for all other paths are shown in the last column of Table 2.

The percentage of time spent in the inner paths for BDPT

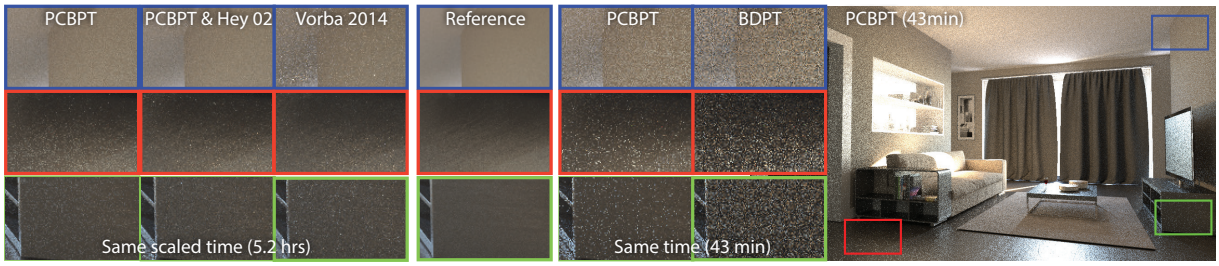


Figure 11: *Same time comparisons for Living Room scene. Inset results left to right: PCBPT (7879 iterations), PCBPT combined with [Hey and Purgathofer 2002] (5838 iters), [Vorba et al. 2014] for 5.2 hours of computation (wall-clock (unscaled) time for [Vorba et al. 2014] 1h); reference; PCBPT, BDPT after 43 minutes and 1100 iterations. Right: PCBPT image.*

	Common to Both		PCBPT only		BDPT
Scene	P. Gen.	PT/LT	PMF	S/C	inner
Kitchen	1314	648/202	786	2267	3161
Apartment	868	383/141	425	1708	2068
Living Rm.	671	799/169	92	472	552
Plants	550	355/53	149	646	764

Table 2: *Computation time breakdown in milliseconds per iteration. P. Gen: generation of light and eye paths (line 1 in Algorithm 1) , LT/UPT: light and path tracing (l. 2); the first two steps are common to BDPT and PCBPT. The next two columns concern PCBPT only; PMF: creation and storage of importance records (l. 5-11), S/C: sampling and connecting all other paths in PCBPT, which includes KNN lookup and PMF interpolation (l. 13-20). BDPT is the time required for BDPT to compute all inner paths.*

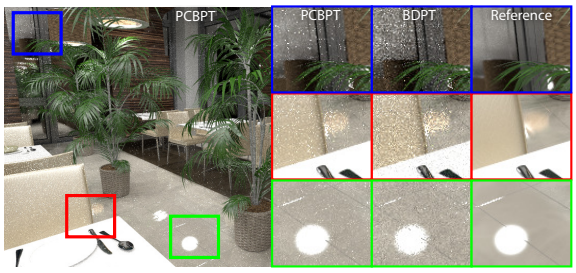


Figure 12: *Same time comparison (5min) for Plant scene. Left: PCBPT image. Insets: PCBPT, BDPT, reference. PCBPT and BDPT performed 167 and 163 iterations respectively.*

is 59%, 60%, 25% and 44% for Kitchen, Apartment, Living Room and Plant respectively. The potential for gain from our algorithm is higher for the scenes where more time is spent on inner paths.

7.1. Comparisons

For the same-time comparison with Vorba et al. [VKv*14], we did not have access to the software, so we matched the convergence graph numbers for BDPT for the Living Room scene from the publication [VKv*14], which provided the scale factor between the different programs: our BDPT is 5.2 times slower than the commercial Corona renderer used in that work. We report the measured and scaled computation times where relevant. Full images of all comparison renderings are included as supplemental material.

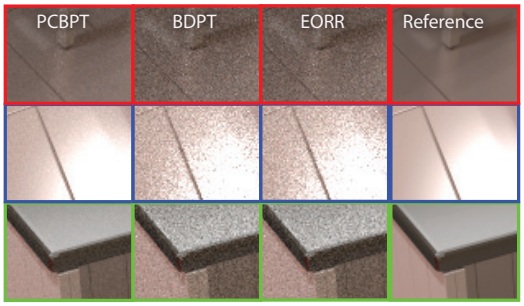


Figure 13: *Same time comparison for 8.4 min for Kitchen scene, insets correspond to Fig. 1. Left to right: PCBPT (97 iterations), BDPT (92 iterations), EORR (102 iterations), and reference.*

We present comparisons to BDPT [VG95a] as a baseline, in Fig. 1 and Figs. 10, 11, 12, and 13. We see that in all cases PCBPT significantly reduces noise levels compared to BDPT. Noise reduction is particularly evident in regions with glossy materials.

We implemented the method of Hey and Purgathofer [HP02], and combined it with PCBPT, which improves the unidirectional path tracing step. We compare this combined algorithm with Vorba et al. [VKv*14] in Fig. 11. Our approach has significantly lower noise at the back of the room, where light is due to non LT/UPT paths, but the method of Vorba et al. improves highlights on the floor, and converges faster than our method in the L_1 sense. As mentioned previously, the methods are complementary: PCBPT importance samples connections, and can only benefit from a method such as that of Vorba et al. [VKv*14] to improve importance sampling of the paths. The methods can be combined in the future in the same way as we combined PCBPT with the approach of Hey and Purghatofer [HP02].

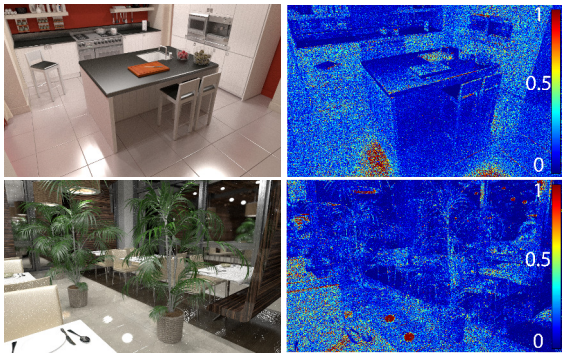


Figure 14: *Left: Images where we compute the exact PMF at every eye vertex, for the same number of iterations as the images of Figs. 1 and 10. Right: Difference multiplied by 10 for clarity, which is low even in the Plants scene with low spatial coherence.*

We implemented EORR [Vea98], and demonstrate that this strategy to reduce ray-intersections is no longer very beneficial for modern renderers. In particular, profiling showed that visibility computations no longer dominate for our scenes and renderer. Visibility tests are now so efficient compared to evaluating the unoccluded contribution that it is faster to evaluate visibility first, and then evaluate the contribution. We use this strategy for PCBPT. EORR however first evaluates the unoccluded contribution and only evaluates visibility if required; this may result in *higher* computation time. EORR helps marginally for the Kitchen scene, decreasing the rendering time of BDPT by 4.5% at essentially constant error (EORR L_1 error 0.0412 compared to 0.0417 for BDPT); visually the results are equivalent (Fig. 13). In the Living Room scene however, where many paths don't contribute to the image, EORR increases the rendering time (by 17%) and with similar error (0.1384 to 0.1383), again with visually comparable results.

A comparison to more recent Metropolis global illumination methods, which also use non-local information to explore path space, can be found in supplemental material.

VCM [GKDS12] and the path extension algorithm [HPJ12] are designed to treat SDS paths that are not handled by BDPT; a comparison would probably be uninformative. Inconsistent biased algorithms do not converge to the same reference image; we thus do not include comparisons to VPL methods, or bidirectional light cuts.

7.2. Evaluation

We evaluate the three key components of our approach: 1) the choice of M and K, 2) the effect of caching and interpolation of PMFs and 3) the effect of our upper-bound MIS weights compared to traditional MIS weights. We also show the improvement on *inner paths* (i.e., only for paths with connections) and present analysis of convergence.

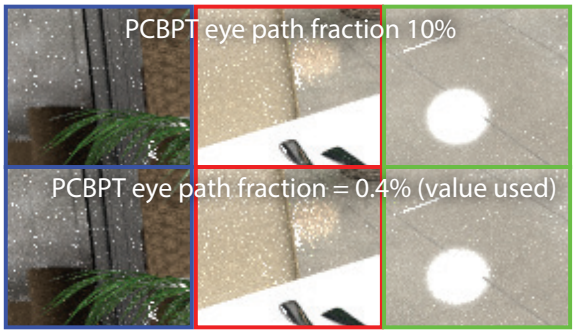


Figure 15: *Increasing the number of eye-paths stored for caching does not have a large influence on results, in most regions of the image (insets same as Fig. 12). Computation times are respectively 5 min (below) and 28 min (above), for 167 iterations.*

Choice of M and K. To evaluate the effect of M, we fixed computation time to 20 and 12min (Kitchen and Apartment respectively), and computed results with 5 different values of M: 50, 100, 200, 400, 800. For values lower than 50 gains were too low to compensate for the additional overhead of our method. As we can see from Table 3, there is no gain beyond M=200. While this may initially seem counterintuitive, a large M incurs higher costs in cache creation and connection sampling, thus decreasing the overall gain possible; larger M may actually *increase* error (see Table 3) since PCBPT can do fewer iterations. To choose K for a given scene, we perform an iteration of BDPT (e.g., on a very small resolution image), to determine the value of K which ensures that this part is not more expensive than BDPT.

Scene \ M	50	100	200	400	800
Kitchen	0.0235	0.0233	0.0250	0.0289	0.0344
Apartment	0.1204	0.1208	0.1217	0.1250	0.1282

Table 3: *Effect of changing M: L_1 error from reference.*

Effect of Caching and Interpolation of PMFs. We computed the images of Fig. 1 and Fig. 12 using the exact PMF at each eye vertex instead of interpolating from the cache. We show the result and the difference images (x10) in Fig. 14. As we can see, our caching scheme shows very low error in smooth regions, and even for the Plant scene with detailed illumination features, error is low. We currently use $0.004 \times W \times H$ eye paths for caching; increasing this fraction does improve quality, but only marginally. In Fig. 15, we see that even when increasing to 10% instead of 0.4%, the gain is marginal in most places and most notable for glossy reflections; however time increases from 5 to 28min. Pushing to 20% increased running time from 28 to 44 min. with little improvement. Caching works well for scenes with many highly glossy surfaces (e.g., Plants): MIS gives low weights to paths connecting from highly glossy surfaces, so errors

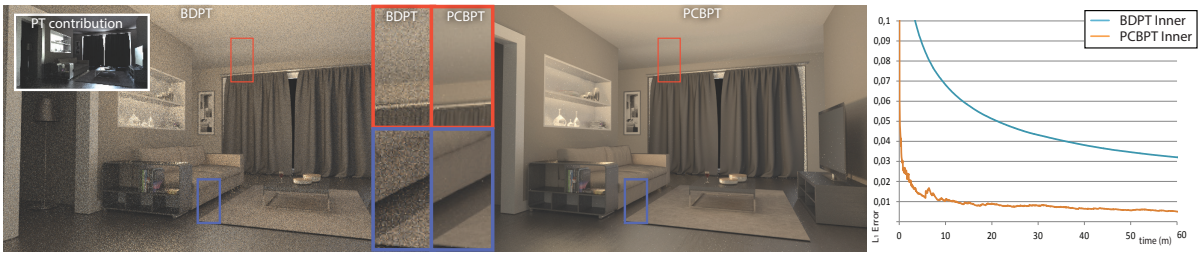


Figure 16: *Left: contribution only from inner paths for BDPT. Center: same for PCBPT. Noise levels are reduced dramatically (inset). Right: L_1 error with iterations. The error drops sharply in the first few iterations. See text for further discussion.*

from the cache have little effect. The cache size for the images shown is 50Mb. For 1080p it is 200Mb, which is still reasonable. Higher resolution for the same viewpoint improves the result of the cache since there is more coherency across pixels: For the kitchen scene at 1080p and same number of paths/pixel, average L_1 error for PCBPT dropped by 12%.

Improvement for inner paths only. PCBPT will only improve the performance for inner paths. In Fig. 16 we show the improvement only for these paths. On the right we show the decrease in error for these paths with iterations. We see that the error decreases dramatically in a very small number of iterations, greatly improving the quality of the overall images even after a few seconds. The ratio between the error for BDPT and PCBPT for these paths is 6.4 on average. For other scenes, this ratio is lower however: 1.6, 2.4 and 1.7 for Kitchen, Apartment and Plant respectively (see supplemental for graphs). This is due to the fact that the PMFs for light paths for the Living Room image have a few small peaks, and thus the blending with the uniform distribution is less conservative than in the other scenes. Note that because of higher correlation, for a single run the error in inner paths for PCBPT is noisy (even sometimes non-monotonic); however the convergence of the full result is smooth as can be seen in Fig. 18.

Evaluation of MIS weight approximation. We also show images in Fig. 8 and Fig. 17 illustrating the difference between using standard MIS weights and our upper bound. We disable the effect of PMF interpolation for this test to highlight the sole effect of weighting. The use of standard MIS weights with correlated paths clearly increases noise, especially in the presence of glossy or specular materials. In the Apartment scene we see spiky noise on glass materials in Fig. 17, which persists even with very large numbers of iterations.

Note that correlation concerns eye paths to all M paths, and has a significant effect on image quality (see Fig. 8).

Convergence. We next discuss the convergence properties of the complete solutions, i.e., including all paths. We compute reference solutions for all scenes and show the convergence graphs in Fig. 18, plotting L_1 error. PCBPT always converges faster than BDPT. As we can see in Fig. 18,

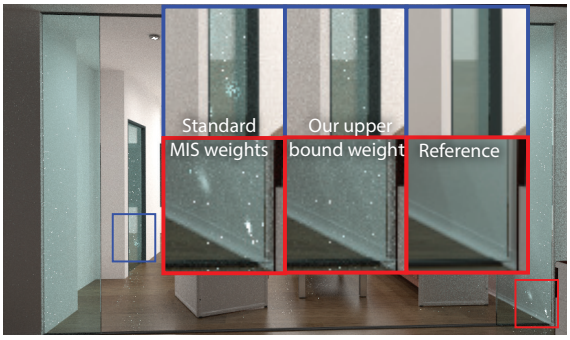


Figure 17: *Evaluation of MIS upper bound weights. We show the apartment scene computed with standard MIS weights. Left inset: result with standard MIS weights. Middle inset: result with our upper bound. Right: reference. Standard weights result in high intensity noise in regions with glass due to path correlation.*

PCBPT in Kitchen and Apartment converges faster than for Living Room and Plants, compared to BDPT. To achieve the same image quality as that shown in our results Figs 1, 10-12, BDPT requires 3.4, 3.1, 1.6 and 1.9 times more computation time, which is consistent with the respective rates of convergence. The lower number for Living Room is due to the fact it is the only scene where UPT contribution is high. This is shown in the top left inset of Fig. 16; for all other scenes UPT paths contribute little, and the corresponding images are mostly black; please see supplemental material. This is consistent with the computation breakdown in Table 2. The Plants scene contains complex glossy materials, such as the highly glossy mirror on the entire wall, which also reduces the gain from PCBPT. We discuss these issues further in Sect. 7.3.

7.3. Discussion and Limitations

Our method inherits the limitations of BDPT and related methods, for example in the treatment of SDS paths. This can be seen in the higher levels of noise in the glass in Fig. 13 top row inset. To solve these, our method could be combined with VCM [GKDS12] or PEA [HPJ12] at the cost of providing a biased but consistent result.

PCBPT incurs overhead for creating and querying the

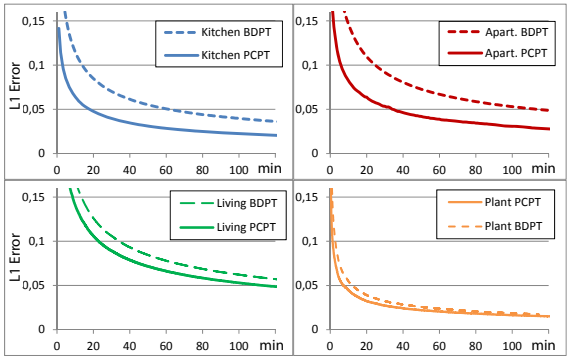


Figure 18: *Convergence graph for all scenes using L_1 distance to reference. Dashed lines give convergence for BDPT and solid lines for our method.*

Figure 19: *Detail of Living Room scene: In this region PCBPT and BDPT are equivalent.*

PMFs. As a result, BDPT can perform more iterations for the same time in some specific configurations. In such cases, BDPT can do as well as our method or even slightly better in some image regions. An example is a scene with a light source outside the scene, in regions where unidirectional path tracing ($s = 0$) contributes most to the result; since BDPT can trace more of these, the gain of PCBPT is lost and the results are equivalent (Fig. 19).

As can be seen in the results, our method achieves best performance for scenes where inner paths dominate the computation (Kitchen and Apartment see Table 2). Gains are more moderate when path-tracing contributes to many pixels in the image, as is the case for Living Room (see UPT inset in Fig. 16), despite the good performance for inner paths. In scenes with multiple highly glossy/specular paths, e.g., those from the highly glossy wall in Plants, PCBPT reduces noise in many places, but overall error can remain high for many pixels in the image over iterations, affecting performance.

8. Conclusions and Future Work

We presented Probabilistic Connections for Bidirectional Path Tracing, in which connections are importance sampled for bidirectional path tracing. We obtain superior results over previous unbiased methods for the same computation time.

Currently we ignore weights when computing the PMFs; incorporating this without incurring a high computational overhead is non trivial, but could theoretically improve results. Doing this implies increasing the dimension of the cache; our tests showed that the current approach provides the best tradeoff between improved performance and additional overhead. The problem still remains low-dimensional since the MIS term $w_i(\bar{y}\bar{z})$ in Eq. (10), can be represented in a recursive manner [Geo12] and computed from only two single floating point values, stored in the last eye and light path vertices respectively.

Currently we use simple Monte Carlo sampling of connections; it could be possible to develop a Metropolis sampling method instead, although care must be taken to retain the efficiency of our method.

Our method can be used in combination with other recent methods which improve sampling (e.g., [VKv*14,HKD14]); since we importance sample connections as well, the combined result should be significantly better. Even though we focused on unbiased rendering here, combining this approach with VCM [GKDS12] or PEA [HPJ12] would provide a practical solution for treating hard illumination paths (such as *SDS*).

As demonstrated in our results, PCBPT can dramatically reduce the variance from inner paths, by probabilistically sampling stored light path information. The error for these paths can be up to 6.4 times lower compared to BDPT, resulting in faster convergence than BDPT for all scenes. For the full solution (all paths), and the images shown in our results, we achieve speedups of up to 3.4 compared to BDPT to achieve the same quality.

9. Acknowledgments

This work was funded by the ANR ALTA (project ID: ANR-11-BS02-0006). R. Ramamoorthi acknowledges support by NSF grants 1115242, 1451830, and the UC San Diego Center for Visual Computing. F. Durant acknowledges support by NSF grant IIS-1420122. Thanks to J. Vorba for help with the comparisons.

References

[BRDC12] BASHFORD-ROGERS T., DEBATTISTA K., CHALMERS A.: A significance cache for accelerating global illumination. In *Comp. Graph. Forum* (2012), vol. 31, Wiley. 3

[DW95] DUTRE P., WILLEMS Y. D.: Importance-driven Monte Carlo light tracing. In *EGWR*. 1995, pp. 188–197. 2

[Geo12] GEORGIEV I.: *Implementing Vertex Connection and Merging*. Tech. rep., Saarland University, 2012. 11

[GKDS12] GEORGIEV I., KŘIVÁNEK J., DAVIDOVIČ T., SLUSALLEK P.: Light transport simulation with vertex connection and merging. *ACM Trans. Graph. 31*, 6 (2012). 3, 9, 10, 11

[GKH*13] GEORGIEV I., KŘIVÁNEK J., HACHISUKA T., NOWROUZEZAHRAI D., JAROSZ W.: Joint importance sampling of low-order volumetric scattering. *ACM Trans. on Graph. 32*, 6 (2013). 2

[GKPS12] GEORGIEV I., KŘIVÁNEK J., POPOV S., SLUSALLEK P.: Importance caching for complex illumination. *Comp. Graph. Forum 31*, 2 (2012). EUROGRAPHICS 2012. 2, 3, 4, 5, 7

[HKD14] HACHISUKA T., KAPLANYAN A. S., DACHSBACHER C.: Multiplexed metropolis light transport. *ACM Trans. on Graph. 33*, 4 (2014), 100. 11

[HP02] HEY H., PURGATHOFER W.: Importance sampling with hemispherical particle footprints. In *SCCG* (2002). 2, 3, 6, 8

[HPB07] HAŠAN M., PELLACINI F., BALA K.: Matrix row-column sampling for the many-light problem. In *ACM Trans. on Graph.* (2007), vol. 26, ACM, p. 26. 2

[HPJ12] HACHISUKA T., PANTALEONI J., JENSEN H. W.: A path space extension for robust light transport simulation. *ACM Trans. on Graph. 31*, 6 (2012), 191. 3, 9, 10, 11

[Jen95] JENSEN H. W.: Importance driven path tracing using the photon map. In *EGWR*. 1995. 2

[Jen01] JENSEN H. W.: *Realistic image synthesis using photon mapping*. A. K. Peters, Ltd., 2001. 3

[Kaj86] KAJIYA J. T.: The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques* (1986), SIGGRAPH '86. 2

[Kel97] KELLER A.: Instant radiosity. In *SIGGRAPH'97* (1997). 2

[KK06] KOLLIG T., KELLER A.: Illumination in the presence of weak singularities. In *Monte Carlo and Quasi-Monte Carlo Methods 2004*, Niederreiter H., Talay D., (Eds.). 2006. 2

[LW93] LAFORTUNE E. P., WILLEMS Y. D.: Bi-directional path tracing. In *CompuGraphics* (1993), vol. 93. 1, 2

[NNDJ12] NOVÁK J., NOWROUZEZAHRAI D., DACHSBACHER C., JAROSZ W.: Virtual ray lights for rendering scenes with participating media. *ACM Trans. on Graph. 31*, 4 (2012), 60. 2

[PBPP11] PAJOT A., BARTHE L., PAULIN M., POULIN P.: Combinatorial bidirectional path-tracing for efficient hybrid CPU/GPU rendering. *Comp. Graph. Forum 30*, 2 (2011). 2

[SIMP06] SEGOVIA B., IEHL J. C., MITANCHEY R., PÉROCHE B.: Bidirectional instant radiosity. In *EGSR* (2006). 3

[VA11] VAN ANTWERPEN D.: Improving SIMD efficiency for parallel Monte Carlo light transport on the GPU. In *High Performance Graph.* (2011). 2

[Vea98] VEACH E.: *Robust monte carlo methods for light transport simulation*. PhD thesis, Stanford University, 1998. 2, 3, 4, 5, 6, 7, 9

[VG95a] VEACH E., GUIBAS L.: Bidirectional estimators for light transport. In *EGWR*. 1995. 1, 2, 8

[VG95b] VEACH E., GUIBAS L. J.: Optimally combining sampling techniques for Monte Carlo rendering. In *SIGGRAPH'95* (1995). 1, 2

[VKv*14] VORBA J., KARLÍK O., ŠIK M., RITSCHEL T., KŘIVÁNEK J.: On-line learning of parametric mixture models for light transport simulation. *ACM Trans. on Graph. 33*, 4 (2014). 2, 7, 8, 11

[WFA*05] WALTER B., FERNANDEZ S., ARBREE A., BALA K., DONIKIAN M., GREENBERG D. P.: Lightcuts: a scalable approach to illumination. In *SIGGRAPH '05* (2005). 3

[WFWB13] WOOP S., FENG L., WALD I., BENTHIN C.: Embree ray tracing kernels for CPUs and the Xeon Phi architecture. In *ACM SIGGRAPH 2013 Talks* (2013), p. 44. 7

[WKB12] WALTER B., KHUNGURN P., BALA K.: Bidirectional lightcuts. *ACM Trans. Graph. 31*, 4 (2012). 2