Pacific Graphics 2015
N. J. Mitra, J. Stam, and K. Xu
(Guest Editors)

Volume 34 (2015), Number 7

Skeleton based Vertex Connection Resampling for Bidirectional Path Tracing

Laurent Noël and Venceslas Biri

Université Paris-Est, LIGM, France

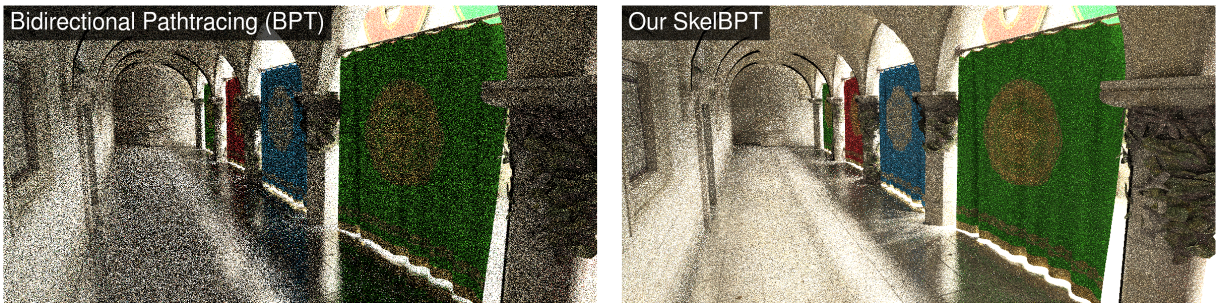

Figure 1: *A comparison between standard BPT and BPT improved with our connection resampling strategy (after 60 seconds of rendering). In this setting, the draperies occlude most of the light vertices from the eye vertices, resulting in many null contributions. Our resampling favors connections that are more likely to produce a high contribution.*

Abstract

Bidirectional path tracing is known to perform poorly for the rendering of highly occluded scenes. Indeed, the connection strategy between light and eye subpaths does not take into account the visibility factor, presenting no contribution for many sampled paths. To improve the efficiency of bidirectional path tracing, we propose a new method for adaptive resampling of connections between light and eye subpaths. Aiming for this objective, we build discrete probability distributions of light subpaths based on a skeleton of the empty space of the scene. In order to demonstrate the efficiency of our algorithm, we compare our method to both standard bidirectional path tracing and a recent important caching method.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Raytracing

1. Introduction

Bidirectional path tracing (BPT) [VG94] is known to be a robust rendering algorithm thanks to its ability to mix various sampling strategies using multiple importance sampling [VG95]. It has been recently combined with photon mapping [HPJ12, GKDS12] to address its principal weakness: the rendering of caustics. The most expensive operation performed by BPT is the connection between a large number of pairs of vertices, requiring many visibility tests. In highly occluded scenes, most tests fail and present no contribution for the visibility tests cost. For such configurations, a caching and resampling scheme similar to the one proposed by Georgiev et al. [GKPS12] can significantly improve the performance of BPT by favoring connections that are likely to produce a high contribution to the final image. These kinds of strategies generally use surface points as importance caches. For robust resampling, a high cache density is required, inducing a high overhead, both in terms of computation and memory.

In this paper, we propose a new solution for resampling connections, based on a centered skeleton of the empty space of the scene, represented by a sparse set of points. We use this skeleton to quickly pre-compute discrete probability distributions of light subpaths based on their contribution around the different parts of the skeleton. These distributions are then used while tracing eye subpaths to resample efficiently the connections with light subpaths. Note that, in this paper, the term "distribution" always refer to a discrete cumulative distribution function for resampling. Our method can be implemented on top of any bidirectional path tracer, potentially improving any other method based on BPT for sampling paths [GKDS12, VG97, VKŠ*14]. The source code will be made freely available on the authors' website after publication. The main contributions of this paper are:

- A new unbiased strategy for vertex connection resampling, based on a pre-computed skeleton of the empty space of the scene.
- A comparison between our method, standard BPT and an adaptation of the method from Georgiev et al. [GKPS12] to BPT. While the latter algorithm performs better for the same number of rendering iterations, our method gives better results for the same rendering time due to a smaller overhead.

2. Related work

Path tracing based methods. Many state-of-the-art rendering algorithms are based on Monte Carlo integration and path tracing. Such methods use various sampling strategies to select random paths and accumulate their contributions for estimating each pixel intensity.

The path tracing algorithm [Kaj86] samples paths by starting from the camera lens and extends a path at each intersection by shooting in a random direction. Bidirectional path tracing (BPT) [VG94] builds a set of paths by sampling simultaneously an eye subpath starting from the camera and a light subpath starting from the light sources. Complete paths are obtained by connecting each vertex of the eye subpath with each vertex of the light subpath. Multiple importance sampling [VG95] (MIS) improves the robustness of BPT by weighting the contribution of a path by considering all possible ways of sampling it. This strategy reduces the contribution of paths carrying high energy but sampled with low probability. Despite the use of MIS, BPT performs poorly in presence of specular materials due to the limited number of strategies able to sample a path containing specular reflections. The photon mapping [Jen01] (PM) algorithm, and its variants [HOJ08, KD13] are able to render efficiently such kind of effects. However, those algorithms are less effective for glossy materials. BPT and PM have been simultaneously combined by Georgiev et al. [GKDS12], with their vertex connection and merging algorithm, and Hachisuka et al. [HPJ12], with unified path sampling. These two methods are able to render efficiently scenes that contain rich material combination and are equivalent in terms of implementation. Many-light rendering (MLR) methods [Kel97, DKH*13] sample a large number of light subpaths and use their vertices as *virtual point lights* (VPL) to illuminate points visible from the camera. These methods have recently received attention for their simplicity and ability to generate noise-free images. They work well in diffuse environments but a large number of VPLs is generally required for glossy materials. Thus, scalable evaluation algorithms [HPB07, WABG06] must be used to achieve reasonable computation cost. Metropolis Light Transport [VG97] is a Markov Chain Monte Carlo algorithm that mutates existing light transport paths to build new path samples. This strategy takes advantage of correlations between pixel measurements and focuses the estimation on paths that carry high energy. The set of initial light transport paths is generally sampled using BPT, before mutations are performed on them.

All of these methods are based on two key components for building complete paths connecting light sources with the camera: ray sampling, for extending an existing path, and vertex connection, for connecting a light subpath with an eye subpath.

Ray sampling. In presence of many occlusions, illumination becomes strongly indirect and the ray sampling strategy must be adapted to importance sample the incident radiance (resp. importance) function for eye (resp. light) subpaths.

Particle tracing is often used for this purpose. Jensen [Jen95] used a pre-computed photon map to approximate a probability density function (PDF) proportional to incident radiance at any surface point. More recently, Vorba et al. [VKŠ*14] followed the same idea, however they use parametric mixture model estimation to learn PDFs from streams of particles. By doing so, they avoid storing particles while still able to sample rays according to incident radiance/importance.

Biri et al. [BC12] take advantage of a topological curvilinear skeleton of the empty space of the scene to sample rays more efficiently toward light sources. Their method uses a shortest-path algorithm to compute importance points along the skeleton which are used to sample rays around preferred directions. Despite a noticeable reduction of noise in their results, the underlying estimator is biased and hard to generalize to multiple light sources. Note that, in this work, we use the same kind of skeleton, however we apply it to the vertex connection component of path sampling.

Vertex connection. Connecting two vertices is an expensive operation because it involves a visibility test. For scenes with complex visibility, it can be very inefficient since many such tests result in no contribution. Indeed, blindly connecting random vertices does not importance sample the visibility function and can introduce high variance in the estimation. Resampling is a general method that applies Monte

Carlo estimation to the sum of contributions already sampled for Monte Carlo integration. It can be used to reduce the number of connections that must be evaluated and to increase the probability of connecting vertices that are likely to produce a high contribution.

Talbot et al. [TCE05] investigate the use of resampling for Monte Carlo rendering. Their method can be applied to the connection problem in BPT and many-light methods by building a discrete probability distribution over the light vertices for each eye vertex. This distribution is built using the contribution of each light vertex without the visibility factor. This operation is expensive when the number of light vertices is high. Moreover, ignoring visibility reduces the robustness of the method in highly occluded scenes since arbitrary high contributions can be reduced to zero when multiplied by the visibility factor.

Georgiev et al. [GKPS12] introduce the importance caching (IC) method and apply the same idea to many-light rendering. However, they build discrete resampling distributions that include the visibility factor on a sparse set of surface points called importance records. For each point to illuminate, they gather nearest importance records using a Kd-tree and resample VPLs according to the pre-computed distributions for these records. By combining these distributions with more conservative ones, they propose a robust estimator that is able to deal with complex scenes. However, gathering nearest importance records is expensive and increases the rendering time dramatically when applied to BPT. Indeed, the number of eye vertices is higher for BPT due to multiple reflections along eye subpaths, and thus increases the number of nearest neighbor queries. Moreover, more resampling distributions have to be pre-computed before each rendering iteration since importance records must be spread on more surfaces than only those visible from the camera. Finally, BPT importance samples efficiently the directional component of the BSDF through local eye subpath sampling, which reduces the interest of introducing it in resampling distributions.

As opposed to established methods, our algorithm caches resampling distributions on a sparse set of points centered in the empty space of the scene, obtained from a curvilinear skeleton. This approach significantly reduces the number of distributions that must be pre-computed for each rendering iteration, while still exploiting visibility and geometric information for efficient resampling.

3. Background

In this section, we review the path integral formulation of light transport and the BPT algorithm. We also present the thinning algorithm to compute the skeleton used in our method.

Path integral framework. The path integral framework [Vea97] is well suited to express the BPT algorithm. In this framework, the intensity I of a pixel is expressed by the *measurement integral* over the space of paths:

$$I = \int_{\bar{\mathbf{x}} \in \Omega} f(\bar{\mathbf{x}}) d\mu(\bar{\mathbf{x}}) \tag{1}$$

A path $\bar{\mathbf{x}} = \mathbf{x}_0 \ldots \mathbf{x}_{k-1}$ is a sequence of vertices such that $\mathbf{x}_0$ is on a light source and $\mathbf{x}_{k-1}$ is on the camera lens. $k \geq 2$ is the *length* of the path, i.e. its number of vertices. The contribution of a path is expressed by the product $f(\bar{\mathbf{x}}) = L_e(\mathbf{x}_0, \mathbf{x}_1) T(\bar{\mathbf{x}}) W_e(\mathbf{x}_{k-2}, \mathbf{x}_{k-1})$ where $L_e(\mathbf{x}_0, \mathbf{x}_1)$ is the radiance emitted at $\mathbf{x}_0$, $W_e(\mathbf{x}_{k-2}, \mathbf{x}_{k-1})$ is the sensor response at $\mathbf{x}_{k-1}$, and $T(\bar{\mathbf{x}})$ is the throughput of the path. The throughput is itself a product $T(\bar{\mathbf{x}}) = \prod_{i=0}^{k-2} G(\mathbf{x}_i, \mathbf{x}_{i+1}) \prod_{i=0}^{k-3} f_s(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{x}_{i+2})$ where $G(\mathbf{x}_i, \mathbf{x}_{i+1})$ is the geometric factor between two sequential vertices and $f_s(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{x}_{i+2})$ is the BSDF evaluated at $\mathbf{x}_{i+1}$ for the incident direction $\overrightarrow{\mathbf{x}_{i+1}\mathbf{x}_i}$ and the outgoing direction $\overrightarrow{\mathbf{x}_{i+1}\mathbf{x}_{i+2}}$. Equation (1) cannot be computed analytically and is generally estimated using a Monte Carlo method, which approximates the integral by a weighted sum of contributions obtained from random path samples.

Bidirectional path tracing. The bidirectional path tracing algorithm [VG94] samples paths by connecting *eye subpaths*, starting at the camera lens, with *light subpaths*, starting at light sources.

More precisely, a sequence of correlated paths $\bar{\mathbf{x}}_{s,t} = \mathbf{y}_0 \ldots \mathbf{y}_{s-1} \mathbf{z}_{t-1} \ldots \mathbf{z}_1$ is obtained by connecting all vertices of a light subpath $\bar{\mathbf{y}}_S = \mathbf{y}_0 \ldots \mathbf{y}_{S-1}$ of length S with all vertices of an eye subpath $\bar{\mathbf{z}}_T = \mathbf{z}_0 \ldots \mathbf{z}_{T-1}$ of length T. We refer to the vertices of a light subpath (resp. eye subpath) as *light vertices* (resp. *eye vertices*).

This sequence of paths is used to estimate the integral (1) conjointly with multiple importance sampling [VG95] (MIS), resulting in the following Monte Carlo estimator:

$$\hat{I} := \sum_{t=0}^{T} \sum_{\substack{s=0 \\ s+t>1}}^{S} w_{s,t}(\bar{\mathbf{x}}_{s,t}) \frac{f(\bar{\mathbf{x}}_{s,t})}{p_{s,t}(\bar{\mathbf{x}}_{s,t})} = \sum_{t=0}^{T} \sum_{\substack{s=0 \\ s+t>1}}^{S} C(\bar{\mathbf{x}}_{s,t}) \tag{2}$$

Each MIS weight $w_{s,t}(\bar{\mathbf{x}}_{s,t})$ associated to a path ensures that its contribution is correctly weighted to take into account each possible way of sampling the path. The *balance heuristic* introduced by Veach et al. [VG95] provides near optimal MIS weights. We use it in our implementation of BPT.

Scene thinning. Our method is based on a curvilinear skeleton of the empty space of the scene, computed by a *thinning algorithm*. Such algorithms generally take a voxelization as input. In our case the scene is first voxelized then complemented to get a voxelization of the empty space. One can choose any thinning algorithm available but we decided to use the topological thinning algorithm described by Chaussard et al. [CNBC13]. To obtain a centered skeleton, this algorithm removes voxels from the voxelized object one layer at a time, starting at the border and in parallel for each layer.

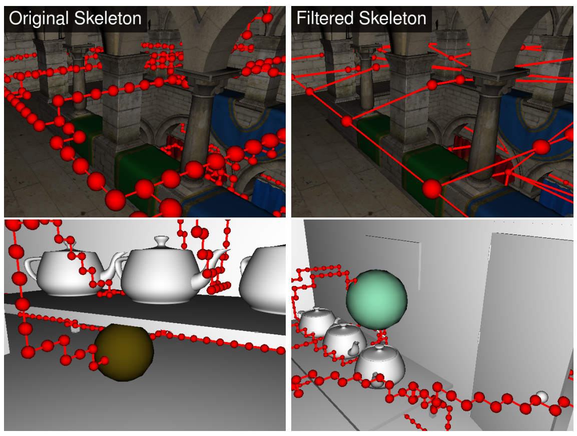

Figure 2: *Top-left: the skeleton of sponza. Top-right: the filtered skeleton. Bottom: two maximal balls of the door scene illustrated as spheres. We use them to compute the filtered skeleton as described in Section 4.4.*

The deletion of voxels is carefully designed so that the topology of the original object is preserved at each step, to ensure that the skeleton does not miss any hole or tunnel in the object. By having a centered skeleton we get a better coverage of the scene and the associated segmentation is sparser. Furthermore, this algorithm can be implemented in parallel to achieve better performance. The resulting curvilinear skeleton is represented by a discrete graph embedded in empty space (see top-left image of Figure 2).

In conjunction with the skeleton, we pre-compute a 3D-grid that maps each voxel to a node of the graph. This grid is computed by propagating the index of each node in the voxelization of the empty space. It results in a grid containing for each voxel the nearest node with respect to geodesic distance. We use this grid to map any 3D point to a node of the skeleton in constant time. We refer to it as the *node mapping grid*.

4. Skeleton based Vertex Connection Resampling for BPT

4.1. Motivation and overview

As discussed in Section 2, connection between random vertices performed by bidirectional path tracing is likely to be inefficient because of occlusions, erasing the potential contribution of many connections. A strategy to solve this problem is to initially sample more than one light subpath and to resample only one of each length at each eye vertex based on visibility information extracted from all light vertices.

For a sequence of N light subpaths $(\bar{\mathbf{y}}^i_S = \mathbf{y}^i_1 \dots \mathbf{y}^i_{S-1})_{i=1,\dots,N}$ and a single eye subpath $\bar{\mathbf{z}}_T$, the bidirectional estimator (2) can be rewritten as:

$$\hat{I} := \sum_{t=0}^{T} \sum_{\substack{s=0 \\ s+t>1}}^{S} \frac{1}{N} \sum_{i=1}^{N} C(\bar{\mathbf{x}}^i_{s,t}) \tag{3}$$

Connections are performed for $s,t \neq 0$, thus the inner sum can be resampled to give the new estimator:

$$\hat{I}_R := \sum_{t=1}^{T} \sum_{s=1}^{S} \frac{1}{N} \frac{C(\bar{\mathbf{X}}_{s,t})}{p^R_{s,t}(\bar{\mathbf{Y}}_{s,t})} \tag{4}$$

Where $\bar{\mathbf{X}}_{s,t}$ is formed by connecting a resampled light subpath $\bar{\mathbf{Y}}_{s,t}$ with the eye subpath $\bar{\mathbf{z}}_t$. The light subpath $\bar{\mathbf{Y}}_{s,t}$ is obtained according to a discrete probability distribution $p^R_{s,t}$ associated to the eye vertex $\mathbf{z}_{t-1}$.

Finding good resampling distributions is challenging because a tradeoff must be made between resampling quality and computation speed. Contrary to previous resampling approaches, our distributions are not cached at surface points but at skeleton nodes, which are points centered in the empty space of the scene. We build our distributions from visibility between nodes and light vertices, but also distance and partial contribution associated to each light vertex. Caching this kind of information at points of empty space instead of surface points is motivated by several arguments:

- Our skeleton sparsely covers the entire scene, therefore few resampling distributions must be pre-computed compared to the number of vertices involved in the estimation.
- The similarity between the visibility function of a node and the visibility function of eye vertices mapped to the node is enough for coarse but cheap resampling.
- Each resampling distribution is shared by a high number of eye vertices, inducing more coherent resampling between these vertices.
- Thanks to our node mapping grid, we obtain in constant time the distributions stored at the nearest node of each eye vertex.

As shown in Section 5, our method improves the convergence speed of bidirectional path tracing at a negligible cost. Moreover, the algorithm is simple to implement on top of any BPT implementation.

Our algorithm progressively renders the image by performing the following steps at each iteration:

1. Sample N light subpaths, resulting in $N \times S$ light vertices where S is the maximal length of a light subpath.
2. Build the resampling distributions of light vertices for each node of the skeleton.
3. Sample an eye subpath for each pixel and estimate the measurement integral by resampling light vertices according to the resampling distributions.

The first step is identical to light subpath tracing performed for bidirectional path tracing, but we store light vertices for resampling them later. We detail the second and third step in Section 4.2 and Section 4.3 respectively.

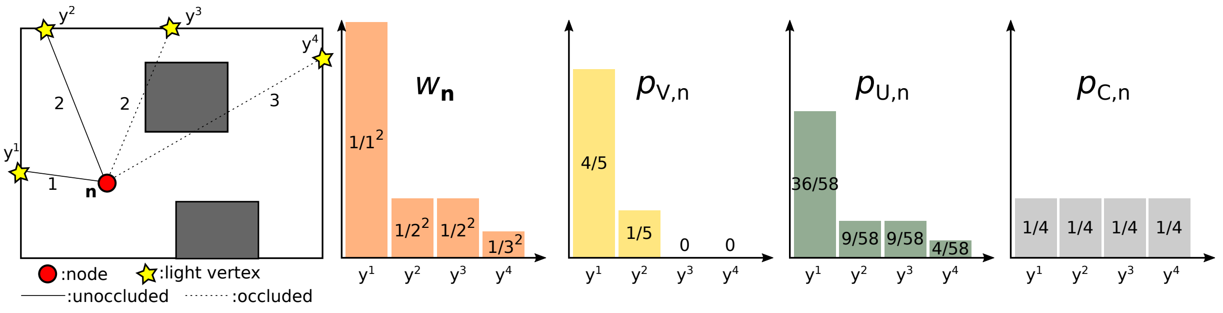

Figure 3: *An illustration of the node weighting function* $w_{\mathbf{n}}$ *and the three resampling probability distributions* $p_{X,\mathbf{n}}$, $X \in \{V,U,C\}$ *for a node* $\mathbf{n}$ *and four light vertices* $\mathbf{y}^i$. *For simplicity, we suppose here that* $L(\mathbf{y}^i) = 1, i = 1...4$, *thus we have* $w_{\mathbf{n}}(\mathbf{y}^i) = \frac{1}{||P_{\mathbf{n}} - \mathbf{y}^i||^2}$. *Distances from the node to light vertices are shown on the scene illustration.*

4.2. Skeleton node resampling distributions

Each node stores $3 \times S$ discrete cumulative distribution functions (CDFs) of size N to perform robust and unbiased resampling. Each CDF is restricted to light subpaths having the same length, in order to connect each eye vertex to one light subpath of each length. The three distributions for a given length are combined using multiple importance sampling at the next step. This approach is similar to importance caching [GKPS12] in which they use four distributions at each importance record.

Our first and second distributions both depend on a common weighting function. Let $\mathbf{n}$ be a node of the skeleton, with position $P_{\mathbf{n}}$. For each light vertex $\mathbf{y}_s^i$, we define the following weight:

$$w_{\mathbf{n}}(\mathbf{y}_s^i) := \frac{L(\mathbf{y}_s^i)}{||P_{\mathbf{n}} - \mathbf{y}_s^i||^2} \tag{5}$$

where $L(\mathbf{y}_s^i)$ is the partial contribution of the subpath $\bar{\mathbf{y}}_{s+1}^i$, that is, the product of emitted radiance, geometric factors and scattering factors divided by the probability density of sampling the subpath.

Our weights are proportional to the inverse squared distance between the node and light vertices to favor those located close to the node. This choice is driven by the fact that such light vertices are also located close to eye vertices mapped to the node. Since the geometric factor between vertices also depends on the inverse squared distance, our weights tend to favor light vertices that give a high geometric factor when connected with an eye vertex.

Our first distribution combines the weighting function (5) with the visibility function of the node:

$$p_{V,\mathbf{n}}(\mathbf{y}_s^i) = \frac{V(\mathbf{y}_s^i, P_{\mathbf{n}}).w_{\mathbf{n}}(\mathbf{y}_s^i)}{\sum_{j=1}^N V(\mathbf{y}_s^j, P_{\mathbf{n}}).w_{\mathbf{n}}(\mathbf{y}_s^j)} \tag{6}$$

By favoring light vertices visible from the node, we ensure a good resampling for all eye vertices that share many visible points with their associated node, which is the case for most of them. However, this distribution is biased since it gives zero probability to some light vertices that actually contribute to eye vertices mapped to the node.

Our second distribution is similar to the first but does not take into account visibility:

$$p_{U,\mathbf{n}}(\mathbf{y}_s^i) = \frac{w_{\mathbf{n}}(\mathbf{y}_s^i)}{\sum_{j=1}^N w_{\mathbf{n}}(\mathbf{y}_s^j)} \tag{7}$$

This one is important for eye vertices mapped to a node that does not approximate well their visibility function. It also ensures that our estimator is unbiased.

Sampling a distant light vertex with distribution $p_{U,\mathbf{n}}$ would give it low probability and possibly introduce additional variance if the light vertex contributes to the illumination of some eye vertices mapped to the node. To avoid this, we use a third resampling distribution, uniform among light subpaths of the same length:

$$p_{C,\mathbf{n}}(\mathbf{y}_s^i) = \frac{1}{N} \tag{8}$$

Figure 3 illustrates both the weighting function and the three resampling probability distributions on a simple example scene.

4.3. Eye subpath sampling

For this last step we sample an eye subpath $\bar{\mathbf{z}}_T = \mathbf{z}_0...\mathbf{z}_{T-1}$ through each pixel to estimate the measurement integral using our resampling distributions. We note $\mathbf{n}_t$ the node associated to the eye vertex $\mathbf{z}_t$. It is obtained by looking at the voxel containing $\mathbf{z}_t$ in the 3D-grid that map each voxel to a skeleton node.

The estimator is then:

$$\hat{I}_{Skel} := \sum_{t=1}^{T} \sum_{s=1}^{S} \frac{1}{N} w_{X,\mathbf{n}_{t-1}}(\bar{\mathbf{Y}}_{X,s,t}) \frac{C(\bar{\mathbf{X}}_{X,s,t})}{p_X \times p_{X,\mathbf{n}_{t-1}}(\bar{\mathbf{Y}}_{X,s,t})} \tag{9}$$

where X is sampled from $\{V,U,C\}$ with uniform probability

$p_X = \frac{1}{3}$ and $\bar{\mathbf{Y}}_{X,s,t}$ is the light subpath resampled from the discrete distribution $p_{X,\mathbf{n}_{t-1}}$. Since we use multiple discrete distributions per eye vertex, we must weight the samples using MIS, which is expressed by the factor $w_{X,\mathbf{n}_{t-1}}(\bar{\mathbf{Y}}_{X,s,t})$ weighting the contribution.

In our implementation, we use the max heuristic [VG95], which assigns a weight of one to a light subpath only if it is resampled with the distribution giving it the highest probability, and zero otherwise. We optimize our distributions by pre-multiplying every light vertex probability by its weight and we re-normalize the distributions according to the new probabilities. Thus, we obtain a reduction of variance of the estimator and weights do not need to be evaluated anymore during eye subpath sampling.

4.4. Optimizations

Despite the sparsity of our skeleton nodes, their number depends on the voxelization resolution used to compute the skeleton. When a high resolution is chosen, the skeleton can be composed of too many nodes in large empty areas. It has a negative impact on our method since we pre-compute and store resampling distributions for each node. To reduce this dependency, we use two optimizations:

Reducing the skeleton size. We propose here a simple solution for reducing the number of nodes of the skeleton based on a geometric information for each node: the radius of the maximal ball centered in the node and contained in empty space. The thinning algorithm that we use provides this information. See bottom images of Figure 2 for an illustration of maximal balls.

The filtering algorithm to simplify our skeleton works as follow. Let $(\mathbf{n}_i)_{i=1...N}$ be the sequence of nodes, $(\mathbf{P}_i)_{i=1...N}$ their positions and $(\mathbf{r}_i)_{i=1...N}$ the radius of their maximal balls. We first sort the nodes according to their radius (largest first). Then for each node $\mathbf{n}_k$ in this order, we remove the nodes $\mathbf{n}_l$ such that $||\mathbf{P}_l - \mathbf{P}_k|| < \mathbf{r}_k$ (meaning that the node $\mathbf{n}_l$ is contained in the maximal ball of $\mathbf{n}_k$). As a result we get a curvilinear skeleton with the same topology but with a density of nodes that depends on the local thickness of the empty space (i.e. less nodes in large empty volumes). Top-right image of Figure 2 demonstrates the result of our simplification.

Reducing the number of distributions. A given view configuration does not require the usage of all nodes of the skeleton. Indeed, many nodes can be located in a part of the scene that is not reachable by eye subpaths. The importance of a node $\mathbf{n}_i$ can be described by the number $N_{mapped}(\mathbf{n}_i)$ of eye vertices mapped to it during the rendering simulation. When this value is small compared to the mean $\bar{N}_{mapped}$ over all nodes, computing the distributions for the node $\mathbf{n}_i$ is not worth the pre-computation time. On our test scenes, more than 50% of the nodes map to less than 1% of the eye vertices.

Config.	Number of nodes	Per iteration
Door	74	9
Sponza	666	45
Sibenik	1420	85

Figure 4: *Number of nodes used per rendering iteration for* $\alpha_{accept} = 1$.

Based on this observation, we introduce the node acceptance parameter $\alpha_{accept} \in [0,1]$ that is used to ignore nodes that are not mapped to enough eye vertices. The values $\left(N_{mapped}(\mathbf{n}_i)\right)_{i=1...N}$ are accumulated at each iteration and for each node. Before pre-computing the resampling distributions for a given iteration, all nodes $\mathbf{n}_i$ such that $N_{mapped}(\mathbf{n}_i) \leq \alpha_{accept}.\bar{N}_{mapped}$ are discarded for this iteration. We fall back to a uniform resampling distribution for eye vertices that are mapped to these nodes. This optimization increases significantly pre-computation time while keeping the same rendering quality. For our results, we simply set $\alpha_{accept} = 1$, so every node mapping to fewer eye vertices than the mean among all nodes is discarded for the iteration. Table 4 records the number of nodes used per iteration compared to the number of nodes of the filtered skeleton.

5. Results

All of our results were rendered on a PC with two CPUs Intel Xeon E5-2650 hexa-core at 2.0 Ghz.

Rendering configurations. We compare our algorithm (SkelBPT) against standard BPT and importance caching [GKPS12] adapted to BPT (ICBPT) on three scenes with different view points, lighting configurations and materials. All algorithms sample complete paths having a maximal length of 7 and trace one eye subpath per pixel per iteration. We use the balance heuristic to compute multiple importance sampling weights for BPT path weights. The max heuristic is used for combining resampling strategies for both our method and ICBPT. The images have a resolution of 1024 x 512 pixels but were cropped to fit in the article. We sample a number of light subpaths equal to the number of pixels at each iteration but we perform connection resampling only on a subset of $N = 1024$ light subpaths randomly chosen in order to keep the overhead of resampling reasonable (both for SkelBPT and ICBPT). We do not apply resampling on paths sampled with the camera projection strategy ($t = 1$) since this strategy requires many light subpaths to be effective. Our implementation of the balance heuristic takes into account the number of paths sampled by each strategy. Our reference images were computed using standard BPT.

ICBPT. We adapted importance caching [GKPS12] to BPT in order to compare our algorithm to a recent similar method.

At each iteration, we sample a sparse set of eye subpaths and we use their vertices as importance records (IR) for resampling. The number of IR is controlled by a density parameter $d \in [0,1]$ such that $d\times$ the number of pixels is the number of eye subpaths traced to position importance records. We set $d = 0.001$ for the presented results, resulting in 524 paths for about 3000 importance records. We use a Kd-tree to store and access to the nearest IRs at each eye vertex.

Skeleton computation time. Our method depends on the pre-computation of our skeleton. The method we use has a $\mathcal{O}(n)$ time complexity with n being the number of voxels of the voxel grid. Table 5 records time required to compute the skeleton of each presented scene. These measures include both the voxelization time and thinning time. The voxelization is performed on GPU and the thinning algorithm is partially parallelized. The pre-processing time of our method is negligible compared to targeted rendering times.

Comparisons. Figure 7 illustrates visual comparisons of the results produced with our method against standard BPT and ICBPT. Both SkelBPT and ICBPT increase rendering quality after a fix number of rendering iterations, as demonstrated by Figure 9. However, ICBPT has a higher cost than our method and the quality gain is not enough to compensate the slower rendering time. Indeed, pre-computing the high number of resampling distributions of ICBPT and accessing them through the Kd-tree is too expensive and the results are worse than expected. Figure 8 shows the convergence curves for the L_1 error. We observe that our method gives a lower error than BPT except for the sibenik scene for which the curves overlap. Table 6 records the time required to reach a given error and the speedup relative to BPT. The **Door** scene features difficult visibility due to the narrow opening of the door. In that case, many light vertices are located behind the door and are connected by BPT with eye vertices of the main room, resulting in many null contributions. Figure 10 illustrates the individual contribution of each of our resampling distributions for this configuration. The scene is provided by Miika Aittala, Samuli Laine, and Jaakko Lehtinen.

The **Sponza** scene is illuminated by a strong directional light source and lighting is only indirect in this configuration. Our strategy reduces significantly the noise generated by standard BPT by choosing light vertices that are likely to contribute to the final image. Despite significant noise reduction on the ground performed by ICBPT, variance is still high as demonstrated by bright spots, especially on the ceiling and draperies. Our reference image has been rendered for 85 hours and still exposes small bright spots on the ground.

Scene	**Resolution**	**Time (seconds)**	**Memory**
Door	(73, 119, 29)	0.32	1 Mo
Sponza	(119, 51, 73)	1.59	1.7 Mo
Sibenik	(119, 91, 51)	0.98	2.1 Mo

Figure 5: *Time required to compute the skeleton and memory overhead to store the node mapping grid.*

Configurations		L_1 error	Time (sec)	Speedup
Door	BPT	0.01	3677.44	$\times$ 1
	ICBPT		7095.64	$\times$ 0.51
	SkelBPT		2341.36	$\times$ 1.57
Sponza	BPT	0.15	1910.39	$\times$ 1
	ICBPT		4731.15	$\times$ 0.40
	SkelBPT		675.98	$\times$ 2.82
Sibenik	BPT	0.025	1334.48	$\times$ 1
	ICBPT		3114.65	$\times$ 0.42
	SkelBPT		1298.69	$\times$ 1.02

Figure 6: *Rendering time required to achieve a given error value and speed up relative to BPT.*

The **Sibenik** scene has less occlusions than the previous ones. The scene is lit by two small area light sources located in corners of the scene. In that configuration, our strategy produces similar results compared to BPT, but additionally slightly reducing visible noise on some parts of the image.

6. Conclusion and future works

We presented a new method to improve the efficiency of bidirectional path tracing by using a skeleton of the empty parts of the scene. We demonstrated experimentally that taking advantage of this skeleton leads to a simple and efficient resampling strategy for algorithms based on vertex connection. We discuss here some limitations and future works.

Combining our method with ICBPT. Importance caching is extremely efficient when applied to many-light rendering. Our first experiments were actually performed on MLR and demonstrated that IC outperforms our method on this algorithm. The reason for this is that MLR only connects points visible from the camera to light vertices (VPLs). This set of visible points can be covered with a limited number of importance records for achieving precise resampling. For BPT, it would be interesting to use IC at eye vertices visible from the camera and our skeleton based resampling at remaining eye vertices. The method would then benefit from a costly but robust resampling for the directly visible eye vertex of each path and a cheaper but coarse resampling for others.

GPU Implementation. Our method can take advantage of a GPU implementation since our skeleton is static and sparse. A shadow map can be pre-computed for each node and used to compute our resampling distributions that require many visibility tests. Using this strategy would be more efficient than tracing shadow rays since the visibility test could thus be performed in constant time.

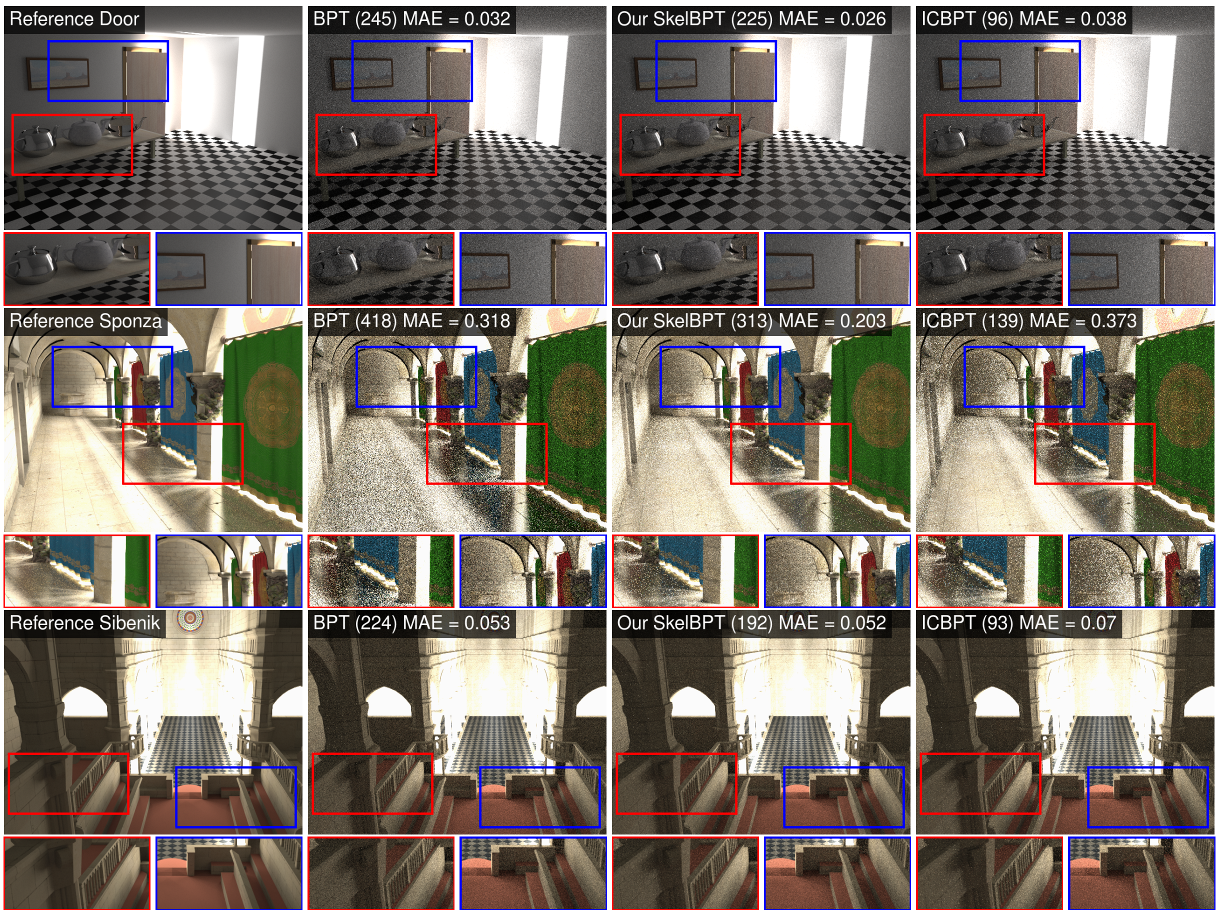

Figure 7: *Visual comparison of BPT, SkelBPT and ICBPT for a rendering time of 300 seconds. The number of iterations performed by each method is indicated as well as the L_1 error (MAE). Our SkelBPT provides a good noise reduction over BPT and outperforms ICBPT for each configuration.*

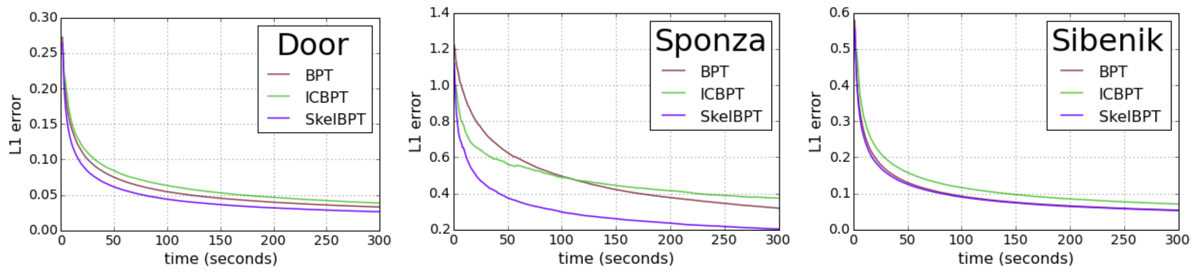

Figure 8: *L1-error curves for a rendering time of 300 seconds.*

References

[BC12] Biri V., Chaussard J.: Skeleton based importance sampling for path tracing. In *proceedings of Eurographics 2012* (mar 2012), pp. 1–4. short papers. 2

[CNBC13] Chaussard J., Noël L., Biri V., Couprie M.: A 3d curvilinear skeletonization algorithm with application to path tracing. In *Discrete Geometry for Computer Imagery - 17th IAPR International Conference, DGCI 2013, Seville, Spain, March 20-22, 2013. Proceedings* (2013), pp. 119–130. 3

[DKH*13] Dachsbacher C., Křivánek J., Hašan M., Arbree A., Walter B., Novák J.: Scalable realistic rendering with many-light methods. In *Computer Graphics Forum* (2013), vol. 33, pp. 88–104. 2

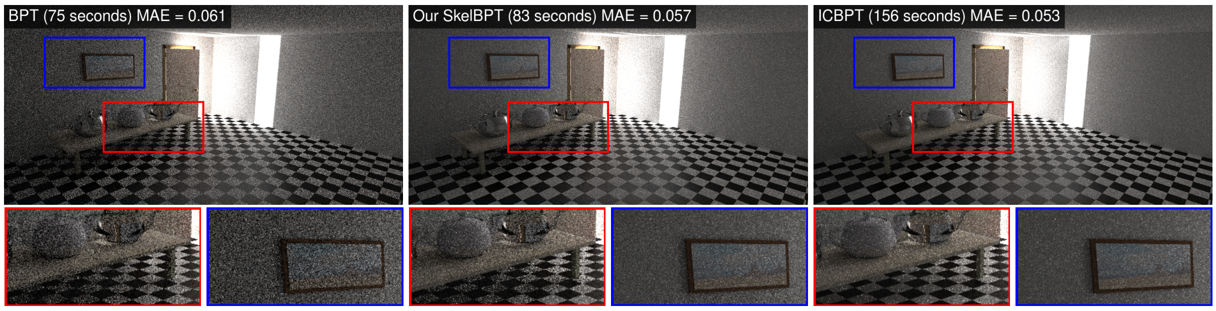
BPT (75 seconds) MAE = 0.061
Our SkelBPT (83 seconds) MAE = 0.057
ICBPT (156 seconds) MAE = 0.053

Figure 9: *Comparison of the three methods after 64 rendering iterations. Given the same number of rendering iterations, ICBPT gives a better result on this configuration. However, the time required by ICBPT is approximately twice the rendering time of BPT. The quality gain is not enough to compensate the overhead introduced by ICBPT, as opposed to our SkelBPT.*

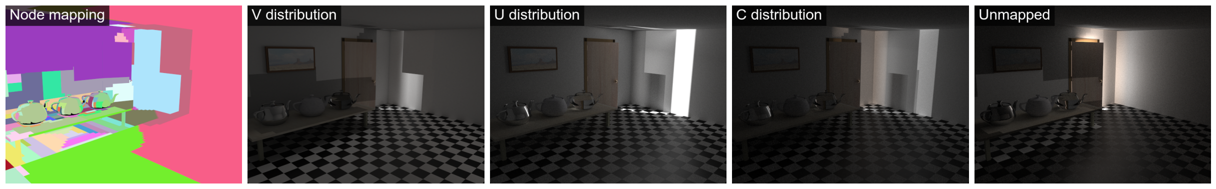

Figure 10: *Individual contribution of each of our distributions on the Door scene after two hours of rendering. Leftmost image illustrates the segmentation of the scene according to our skeleton node mapping. Rightmost image is the contribution associated to the default uniform resampling distribution, used when the node mapped to a surface point has been discarded by our optimization at a given iteration. Multiple importance sampling blends the contributions such that the discontinuities produced by node mapping are not visible in the final result.*

[GKDS12] Georgiev I., Křivánek J., Davidovič T., Slusallek P.: Light transport simulation with vertex connection and merging. *ACM Trans. Graph. 31* (Nov. 2012), 192:1–192:10. 1, 2

[GKPS12] Georgiev I., Křivánek J., Popov S., Slusallek P.: Importance caching for complex illumination. In *Computer Graphics Forum* (2012), vol. 31, pp. 701–710. 1, 2, 3, 5, 6

[HOJ08] Hachisuka T., Ogaki S., Jensen H. W.: Progressive photon mapping. In *ACM Transactions on Graphics (TOG)* (2008), vol. 27, p. 130. 2

[HPB07] Hašan M., Pellacini F., Bala K.: Matrix Row-Column Sampling for the Many-Light Problem. In *ACM SIGGRAPH 2007 papers* (2007). 2

[HPJ12] Hachisuka T., Pantaleoni J., Jensen H. W.: A path space extension for robust light transport simulation. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 191. 1, 2

[Jen95] Jensen H. W.: Importance driven path tracing using the photon map. In *Eurographics Rendering Workshop* (1995), pp. 326–335. 2

[Jen01] Jensen H. W.: *Realistic Image Synthesis Using Photon Mapping*. Natick, MA, USA, 2001. 2

[Kaj86] Kajiya J. T.: The rendering equation. In *ACM Siggraph Computer Graphics* (1986), vol. 20, pp. 143–150. 2

[KD13] Kaplanyan A. S., Dachsbacher C.: Adaptive progressive photon mapping. *ACM Transactions on Graphics 32*, 2 (Apr. 2013), 1–13. 2

[Kel97] Keller A.: Instant radiosity. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (1997), pp. 49–56. 2

[TCE05] Talbot J. F., Cline D., Egbert P.: Importance resampling for global illumination. In *Proceedings of the Sixteenth Eurographics conference on Rendering Techniques* (2005), pp. 139–146. 3

[Vea97] Veach E.: *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, 1997. 3

[VG94] Veach E., Guibas L.: Bidirectional estimators for light transport. In *Proceedings of Eurographics Rendering Workshop* (Berlin, Heidelberg, 1994), Sakas G., Müller S., Shirley P., (Eds.), pp. 145–167. 1, 2, 3

[VG95] Veach E., Guibas L. J.: Optimally combining sampling techniques for monte carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), pp. 419–428. 1, 2, 3, 6

[VG97] Veach E., Guibas L. J.: Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (1997), pp. 65–76. 2

[VKŠ*14] Vorba J., Karlík O., Šik M., Ritschel T., Křivánek J.: On-line learning of parametric mixture models for light transport simulation. *ACM Transactions on Graphics 33*, 4 (July 2014), 1–11. 2

[WABG06] Walter B., Arbree A., Bala K., Greenberg D. P.: Multidimensional lightcuts. In *ACM Transactions on Graphics (TOG)* (2006), vol. 25, pp. 1081–1088. 2