Real-Time Bidirectional Path Tracing via Rasterization

Yusuke Tokuyoshi*
Square Enix Co., Ltd.

Shinji Ogaki†
Square Enix Co., Ltd.

Figure 1: *Completely dynamic scenes are rendered with 1920×1088 resolution and 2×2 supersampling. Both images are computed two-bounce indirect illumination from a point light source on AMD Radeon HD 6990 830MHz.*

Abstract

Global illumination drastically improves visual realism of interactive applications. Although many interactive techniques are available, they have some limitations or employ coarse approximations. For example, general instant radiosity often has numerical error, because the sampling strategy fails in some cases. This problem can be reduced by a bidirectional sampling strategy that is often used in off-line rendering. However, it has been complicated to implement in real-time applications. This paper presents a simple real-time global illumination system based on bidirectional path tracing. The proposed system approximates bidirectional path tracing by using rasterization on a commodity DirectX®11 capable GPU. Moreover, for glossy surfaces, a simple and efficient artifact suppression technique is also introduced.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture;

Keywords: interactive global illumination, global ray-bundles, instant radiosity

1 Introduction

Indirect illumination effects are perceptually important for interactive applications such as video games, lighting design or virtual reality systems. However, they are often omitted or coarsely approximated due to high rendering cost. High-quality global illumination at real-time frame rates is a challenge with high industrial impact. Static or semi-static approaches such as pre-baking can produce high-quality images. However, many applications require completely dynamic approaches to evaluate indirect illumination because scenes may change dramatically.

Nevertheless, most dynamic approaches often have large numerical error. For example, general real-time instant radiosity is unsuitable for large light sources such as environment maps, and the failure of the sampling strategy induces spike artifacts. A bidirectional sampling strategy such as bidirectional path tracing [Lafortune and Willems 1993; Veach and Guibas 1994] is a robust technique to reduce the error. However, this method is hard to compute in real-time. For interactive applications, a hybrid algorithm such as bidirectional instant radiosity [Segovia. et al. 2006a] was proposed. This algorithm does not only require rasterization, but also fast ray tracing. Many techniques including shaders have been developed for rasterization pipeline. Therefore, if all visibility tests are done by rasterization, we can reuse all of them. In particular, hardware tessellation is very attractive as it enables to easily render highly detailed objects with arbitrary displacements.

This paper presents a high-quality real-time rendering system for completely dynamic scenes. Our system approximates bidirectional path tracing by using only GPU rasterization. This is easy to implement and intuitive. The paths from a light source are traced by reflective shadow maps as general real-time instant radiosity [Dachsbacher and Stamminger 2005], and the paths from a camera are traced by global ray-bundles [Sbert 1996]. Global ray-bundles are often used for off-line rendering. This paper demonstrates that global ray-bundles can work in real-time frame rates on a modern commodity GPU. Furthermore, to avoid unpleasant artifacts, our system provides a novel artifact suppression technique for glossy surfaces.

2 Related Work

There are well established off-line global illumination methods such as path tracing [Kajiya 1986]. However, the sampling strategy of tracing paths from only a camera fails to capture a highly intense spot of light. Bidirectional path tracing was introduced by [Lafortune and Willems 1993; Veach and Guibas 1994] to compute two independent paths. One is generated from a camera while the

*e-mail: tokuyosh@square-enix.com
†e-mail: ogaki@square-enix.com

other starts at a light source. These paths are connected by shadow rays. This algorithm is known to be robust but slow.

Instant radiosity [Keller 1997] is a major interactive technique for completely dynamic scenes. It generates a number of virtual point lights (VPLs) to approximate indirect illumination, and shade visible surfaces from these lights. This is a hardware-friendly method because the visibility can be resolved by computing a shadow map for each VPL. The shadow map computation is the main bottleneck of the algorithm, as it requires sampling the scene many times for a reasonable number of VPLs. To address this problem, many methods are developed. Dachsbacher and Stamminger proposed reflective shadow maps [2005] to handle dynamic scenes, but neglected visibility. Segovia et al. [2006b] employ interleaved sampling using G-buffer splitting. This method reduces rendering cost and avoids alias artifacts. Incremental instant radiosity [Laine et al. 2007] was proposed for an animated sequence of static scenes, where only a subset of shadow maps is recomputed at each frame. Ritschel et al. [2008; 2011] introduced imperfect shadow maps to approximate shadow maps of VPLs by coarse point-based rendering. However, a number of VPLs are necessary to remove undersampling artifacts such as flickering. To avoid these temporal artifacts, Knecht [2009] combined the imperfect shadow maps and reverse reprojection caching [Nehab et al. 2007]. Herzog et al. [2010] proposed an upsampling method using reverse reprojection.

In theory, VPL based rendering provides a correct solution to the rendering equation, however, spike artifacts can be produced since the bidirectional reflectance distribution function (BRDF) and the geometry term can have very large values compared to the light density. This is normally handled by *clamping*, which often removes interesting glossy illumination effects [Křivánek et al. 2010]. Virtual spherical lights (VSLs) [Hašan et al. 2009] were introduced to address this issue. However, VSL is insufficient to reproduce sharp details of glossy reflection. Moreover, the contribution of a VSL needs to be calculated by an iterative algorithm. Kollig and Keller [2006] compensated for the clamped illumination in instant radiosity by a recursive path tracing. Davidovič et al. [2010] approximated indirect illumination by using local virtual lights to compensate for energy loss due to clamping. Unfortunately, these techniques are unsuitable for real-time rendering. Novák et al. [2011] proposed screen-space compensation for interactive rendering. However, this method does not take into account the illumination from off-screen objects.

In addition, the sampling strategy of traditional instant radiosity fails in other cases. The reason is that the VPLs are only traced from light sources. Bidirectional instant radiosity proposed by Segovia et al. [2006a] is an extension of instant radiosity. This method can reduce error in some difficult scenes by sampling VPLs also from a camera. However, tracing VPLs from a camera is done by real-time ray tracing. Therefore, in addition to rasterization pipeline, a fast ray tracing system is required. Bidirectional reflective shadow maps [Ritschel et al. 2011] is a technique to easily compute view-adaptive sampling by using only rasterization, but the distribution of VPLs is less optimal. Ritschel et al. [2009] introduced final gathering using micro-rendering. To perform interactive frame rates, scene geometry are represented by a point hierarchy.

This paper uses global ray-bundles [Sbert 1996] to trace camera paths. This can be simply implemented with a modern GPU rasterization. Global ray-bundles were often used for off-line rendering to accelerate visibility test. Szirmay-Kalos and Purgathofer [1998] proposed global ray-bundles with rasterization. Computing global ray-bundles on the GPU is a similar problem to order independent transparency. Hachisuka [2005] proposed global ray-bundles for final gathering. To obtain each depth fragment, depth peeling [Everitt 2001] was used. Ideally, we would need to store a list of fragments per pixel as an A-buffer [Carpenter 1984] in a single-pass. Hermes et al. [2010] used *k*-buffer [Callahan et al. 2005] and they demonstrated high-quality global illumination with multiple glossy reflections using their radiance exchange. Although *k*-buffer can be created in a single-pass, the number of fragments per pixel is fixed. Tokuyoshi et al. [2011] created global ray-bundles with per pixel linked-list on a DirectX 11 GPU [Yang et al. 2010] for baking light maps. This linked-list is faster than depth peeling, and provides an unlimited storage per pixel unlike *k*-buffer. Furthermore, using DirectX 11 rasterization pipeline, they inexpensively generate light maps of highly tessellated objects.

This paper proposes a new real-time rendering system. Our system addresses the problems in real-time rendering by an approximate bidirectional path tracing in a simplistic way.

3 Tracing Camera Paths

For bidirectional path tracing, we have to trace paths from a camera. In this section, we present a simple path tracing via global ray-bundles for real-time rendering.

3.1 Path Tracing

Path tracing is a well-established technique based on Monte Carlo integration to calculate indirect illumination. This algorithm is suitable for large light sources. It is done by evaluating a hemispherical integral via recursive ray tracing. The resulting radiance L at the point $\boldsymbol{x}$ and direction $\boldsymbol{\omega}$ can be computed as follows:

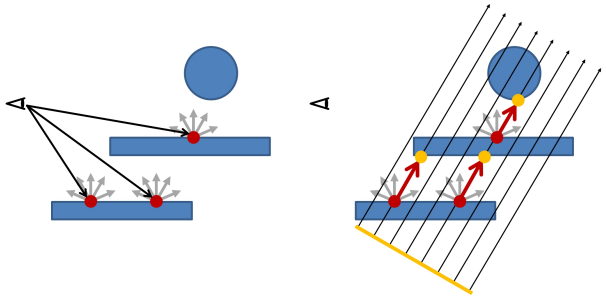
$$L(\boldsymbol{x},\boldsymbol{\omega}) \approx \frac{1}{N}\sum_{i=1}^{N}\frac{L_r(\boldsymbol{x},\boldsymbol{\omega}_i')f(\boldsymbol{x},\boldsymbol{\omega},\boldsymbol{\omega}_i')(\boldsymbol{n}\cdot\boldsymbol{\omega}_i')}{p_r(\boldsymbol{\omega}_i')}, \qquad (1)$$

where N is the number of samples, $L_r(\boldsymbol{x},\boldsymbol{\omega}_i')$ the radiance of the fragment seen from the point $\boldsymbol{x}$ and sample direction $\boldsymbol{\omega}_i'$, f the BRDF, $\boldsymbol{n}$ the surface normal at the point $\boldsymbol{x}$, and p_r the probability density function (PDF). Generally, this is solved by shooting rays in random directions on the unit hemisphere.

3.2 Global Ray-Bundles on the GPU

We approximate path tracing from a camera by using global ray-bundles. Focusing on a single global direction, the visibility for all fragments in a scene is computed in parallel as shown in Figure 2. This can be done by rendering the scene from the sample direction $\boldsymbol{\omega}_i'$ in Equation (1) using parallel projection, similar to rendering a shadow map from a directional light source. In this pass, fragment data (depth, normal, albedo and roughness of specular) is stored to a buffer. In the next pass, $L_r(\boldsymbol{x},\boldsymbol{\omega}_i')$ is calculated using the buffer for each visible point $\boldsymbol{x}$.

As shown in Figure 2 (b), a ray-bundle has multiple fragments in a single pixel. Therefore, we create global ray-bundles using per pixel linked-list construction on a DirectX 11 GPU [Yang et al. 2010]. This method is faster than depth peeling for order independent transparency. Furthermore, for global illumination, this can be done much faster because there is no need to sort the fragments in contrast to order independent transparency. We find the visible fragment from the point $\boldsymbol{x}$ with only linear search (described in the supplemental material). The performance of the global ray-bundles motivated our real-time global illumination system. In addition, this rasterization based approach enables GPU tessellation and the reuse of domain shader codes for global illumination [Tokuyoshi et al. 2011].



(a) visible points from a camera (b) global ray-bundles

Figure 2: *One-bounce path tracing via global ray-bundles. The sample direction is randomly generated in an iterative fashion. Then a ray-bundle is created in each iteration.*

3.3 Importance Sampling

Global ray-bundles transfer radiance not only in the global direction $\boldsymbol{\omega}$, but also into $-\boldsymbol{\omega}$. If $(\boldsymbol{n} \cdot \boldsymbol{\omega}) > 0$, the ray direction is $\boldsymbol{\omega}$, otherwise it is $-\boldsymbol{\omega}$. If the orientation is uniformly sampled, then its PDF is $p_r(\boldsymbol{\omega}) = 1/2\pi$. Our system uses this uniform PDF for indoor scenes or dynamic environment maps. When a scene is lit by a static environment map, importance sampling according to the light can be precomputed for variance reduction [Debevec 2005]. Our importance sampling does not only take into account forward direction, but also backward direction and indirect illumination factor. We precompute global directions using the PDF as follows:

$$p_r(\boldsymbol{\omega}) = \frac{L_e(\boldsymbol{\omega}) + L_e(-\boldsymbol{\omega}) + 2L_o}{\int_\Omega (L_e(\boldsymbol{\omega}') + L_o) d\boldsymbol{\omega}'}, \tag{2}$$

where $L_e(\boldsymbol{\omega})$ is the emitted radiance at the direction $\boldsymbol{\omega}$ of an environment map and L_o is the user-specified offset to account for indirect illumination. This importance sampling is suitable for diffuse surfaces lit by an environment map.

However, in contrast to ray tracing based methods, global ray-bundles cannot sample according to the BRDFs at each visible point. This limitation causes undesirable artifacts on glossy surfaces. In Section 5, we introduce a novel artifact suppression technique.

4 Our Bidirectional Path Tracing

If we consider indirect illumination from point light sources, this can be simply done by computing direct illumination at each intersection point of the paths from a camera. However, this algorithm can be worse than classic instant radiosity when a point light source is near objects. Path tracing from a camera fails to capture a highly intense spot of light such as the bunny ears in Figure 4 (b). On the other hand, VPLs produce undesirable spike artifacts as shown in Figure 4 (a). In traditional instant radiosity, clamping is used to suppress the artifacts. Unfortunately, clamping may change the illumination appearance of the scene.

We combine a camera path via ray-bundle and a light path via VPL using multiple importance sampling (MIS) [Veach and Guibas 1995]. This method dramatically reduces artifacts.

We only consider tracing one-bounce camera paths and light paths, and VPLs are emitted only from point light sources to alleviate computational burden. It computes two-bounce indirect illumination from point light sources. Figure 3 shows three different indirect illumination paths. In our system, $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are randomly generated in an iterative fashion.

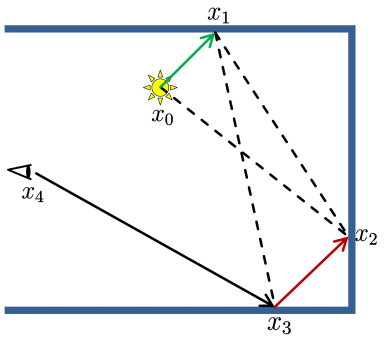


Figure 3: *The sampled indirect illumination paths by our bidirectional path tracing. Black arrow: primary ray via G-buffer. Green arrow: light subpath via reflective shadow map. Red arrow: camera subpath via ray-bundle. Dashed line: connecting edge tested by shadow maps. $\boldsymbol{x}_0$ is the position of the light source, $\boldsymbol{x}_1$ the VPL, $\boldsymbol{x}_2$ the one-bounced intersection point, and $\boldsymbol{x}_3$ the visible point from the camera $\boldsymbol{x}_4$. The path $(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_3)$ and $(\boldsymbol{x}_0, \boldsymbol{x}_2, \boldsymbol{x}_3)$ are one-bounce, and the path $(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$ two-bounce indirect illumination.*

4.1 Creating VPLs

VPL tracing is done with reflective shadow maps [Dachsbacher and Stamminger 2005]. If a light source is omnidirectional, its PDF is $p_e(\boldsymbol{\omega}) = 1/4\pi$. For a spot or textured light source, VPLs are emitted according to the PDF $p_e(\boldsymbol{\omega}) = I(\boldsymbol{\omega}) / \int_\Omega I(\boldsymbol{\omega}') d\boldsymbol{\omega}'$, where I is the intensity of a point light source. VPLs are randomly generated in an iterative fashion, and we create a shadow map for every VPL. Since it is inefficient to emit VPLs from all light sources when many light sources exist, we also use Russian roulette if desired.

4.2 One-bounce Indirect Illumination

The one-bounce indirect illumination paths via VPL and ray-bundle are combined using MIS. MIS is a strategy to combine samples generated from different PDFs using a weighting function. Let w_1 and w_2 be values of the weighting function at the paths via VPL and ray-bundle respectively. The sampled radiance of one-bounce indirect illumination is obtained by the following equation:

$$\begin{aligned} L_1(\boldsymbol{x}_3, \boldsymbol{\omega}_{3,4}) \quad = \quad & \frac{I(\boldsymbol{\omega}_{0,1}) f_{0,1,3} f_{1,3,4} (\boldsymbol{n}_3 \cdot \boldsymbol{\omega}_{3,1})}{p_1(\boldsymbol{x}_1)} w_1 + \\ & \frac{I(\boldsymbol{\omega}_{0,2}) f_{0,2,3} f_{2,3,4} (\boldsymbol{n}_3 \cdot \boldsymbol{\omega}_{3,2})}{p_2(\boldsymbol{x}_2)} w_2, \end{aligned} \tag{3}$$

where $\boldsymbol{\omega}_{i,j} = (\boldsymbol{x}_j - \boldsymbol{x}_i) / \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$, $f_{i,j,k} = f(\boldsymbol{x}_j, \boldsymbol{\omega}_{j,i}, \boldsymbol{\omega}_{j,k})$, and $\boldsymbol{n}_j$ is the surface normal at the position $\boldsymbol{x}_j$. The function p_1 and p_2 are the PDF of sampled paths via VPL and ray-bundle respectively, which are obtained by:

$$p_1(\boldsymbol{x}_j) = \frac{\|\boldsymbol{x}_j - \boldsymbol{x}_3\|^2}{(\boldsymbol{n}_j \cdot \boldsymbol{\omega}_{j,3})} p_e(\boldsymbol{\omega}_{0,j}), \tag{4}$$

$$p_2(\boldsymbol{x}_j) = \frac{\|\boldsymbol{x}_j - \boldsymbol{x}_0\|^2}{(\boldsymbol{n}_j \cdot \boldsymbol{\omega}_{j,0})} p_r(\boldsymbol{\omega}_{3,j}). \tag{5}$$

The weighting function w_j is given by:

$$w_j = \frac{p_j^\beta(\boldsymbol{x}_j)}{p_1^\beta(\boldsymbol{x}_j) + p_2^\beta(\boldsymbol{x}_j)}, \tag{6}$$

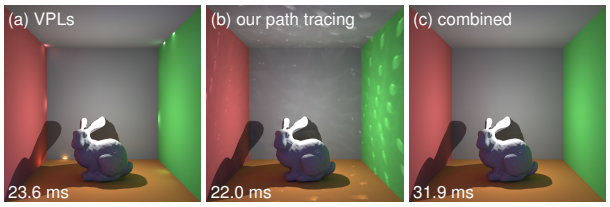


Figure 4: *Comparison of one-bounce global illumination (70 k triangles scene, 21 VPLs and 21 ray-bundles per frame). (a) VPLs produce spike artifacts in the corner of a room. (b) Path tracing is inefficient when a scene has a highly bright spot such as the bunny ears. (c) The combined method generates high-quality image.*

where β is the user-specified parameter of the power heuristic [Veach and Guibas 1995]. We use $\beta = 1.0$ for simplicity. This combination produces accurate results with fewer artifacts. Hence, clamping is no longer needed in our one-bounce indirect illumination on diffuse surfaces (Figure 4 (c)).

4.3 Two-bounce Indirect Illumination

The visibility of the edge $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is inexpensively obtained unlike classic ray tracing based methods, because the shadow map of a VPL can be shared by both one-bounce and two-bounce calculations. The radiance of path $(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$ is represented by:

$$L_2(\boldsymbol{x}_3, \boldsymbol{\omega}_{3,4}) = \frac{I(\boldsymbol{\omega}_{0,1}) f_{0,1,2} f_{1,2,3} f_{2,3,4} (\boldsymbol{n}_3 \cdot \boldsymbol{\omega}_{3,2})}{p_{1,2}}. \quad (7)$$

The PDF $p_{1,2}$ is:

$$p_{1,2} = \frac{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2 p_e(\boldsymbol{\omega}_{0,1}) p_r(\boldsymbol{\omega}_{3,2})}{(\boldsymbol{n}_1 \cdot \boldsymbol{\omega}_{1,0})(\boldsymbol{n}_2 \cdot \boldsymbol{\omega}_{2,3})}. \quad (8)$$

This calculation produces spike artifacts if $\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|$ is small. However, two-bounce indirect illumination affects illumination appearance much less than one-bounce indirect illumination. Therefore, we simply use clamping to avoid the artifacts.

5 Artifact Suppression for Glossy Surfaces

The BRDF can have a very large value compared to the light density. This produces undesirable spike artifacts, which can also be avoided by clamping. However, clamping causes overall image darkening. This in turn can negatively impact the illumination appearance, especially of glossy surfaces. Instead we suppress artifacts by smoothing. This suppression does not induce loss of illumination as VSL does. Moreover, it is simpler and faster than VSL. This is done by clamping roughness parameters of glossy materials based on kernel radius estimation.

5.1 Clamping Roughness Parameters

Our kernel radius estimation is the inverse operation of kernel density estimation. Christensen [1999] precomputes the particle radius r_i based on the k-nearest neighbor algorithm. Expressed in another way, if the density $\hat{d}(\boldsymbol{y}_i)$ is given, we can obtain the appropriate r_i.

Most glossy reflection models have a bell shaped parametric distribution function such as phong distribution [Lafortune and Willems 1994]. As shown in Figure 5, we consider the density of a halfway vector in a directional space, and treat the distribution function D as a kernel function. That is, we consider the roughness parameter

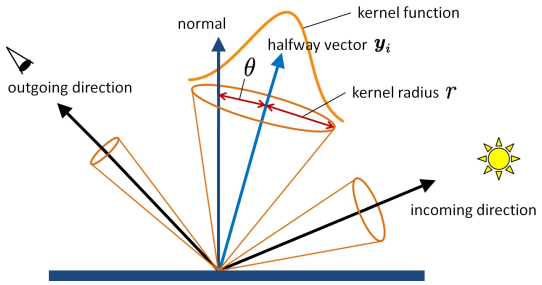


Figure 5: *The distribution function is treated as a kernel function. The roughness parameter is corresponding to the kernel radius of the sampled halfway vector. The appropriate roughness is computed based on the kernel radius estimation for each sample.*

of a material approximately corresponds to r. The maximum value of D is limited as follows:

$$\max_{\theta}(D(\theta)) \frac{m}{\hat{d}(\boldsymbol{y}_i)} \leq 1, \quad (9)$$

where m is the number of nearest neighbor particles. To satisfy Equation (9), we change the roughness of the material for each sample. For example, in the case of D being the phong distribution as: $D(\theta) = \frac{\alpha+1}{2\pi} \cos^{\alpha}(\theta)$ (α is the phong exponent to control the roughness of a surface), α is bounded as $0 \leq \alpha \leq \frac{2\pi \hat{d}(\boldsymbol{y}_i)}{m} - 1$. Therefore, we clamp α as follows:

$$\alpha_i = \max\left(0, \min\left(\alpha, \frac{2\pi \hat{d}(\boldsymbol{y}_i)}{m} - 1\right)\right). \quad (10)$$

We use α_i instead of α. If a distribution function has two roughness parameters for anisotropic reflection, we clamp each parameter in a similar fashion. For example, when D is an Ashikhmin and Shirley model [2000] defined as:

$$D(\theta) = \frac{\sqrt{(\alpha_x + 1)(\alpha_y + 1)}}{2\pi} \cos^{\alpha_x \cos^2\phi + \alpha_y \sin^2\phi}(\theta), \quad (11)$$

we obtain the clamped parameter as follows:

$$\alpha_{x,i} = \max\left(0, \min\left(\alpha_x, \frac{2\pi \hat{d}(\boldsymbol{y}_i)}{m} - 1\right)\right). \quad (12)$$

5.2 Density

We also use Equation (9) even if $D(\theta)$ is not based on a halfway vector. The density $\hat{d}(\boldsymbol{y}_i)$ is determined by both the incoming and outgoing density of the ith sample. For an arbitrary BRDF, we assume:

$$\frac{1}{\hat{d}(\boldsymbol{y}_i)} = \frac{1}{d(\boldsymbol{x}, \boldsymbol{\omega}_i)} + \frac{1}{d(\boldsymbol{x}, \boldsymbol{\omega}_i')}, \quad (13)$$

where $d(\boldsymbol{x}, \boldsymbol{\omega}_i)$ is the density of samples per unit solid angle at the position $\boldsymbol{x}$ and direction $\boldsymbol{\omega}_i$. $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_i'$ is the incoming direction and the outgoing direction, respectively. We show the case of Figure 3. For the sake of simplicity, we write $\hat{d}(\boldsymbol{y}_i)$ at $\boldsymbol{x}_j$ as $\hat{d}_j$, and $d(\boldsymbol{x}_j, \boldsymbol{\omega}_{j,k})$ as $d_{j,k}$. For the direct illumination of environment maps, $\hat{d}$ is given by:

$$\hat{d}_3 = d_{3,2} = N p_r(\boldsymbol{\omega}_{3,2}), \quad (14)$$

because $d_{3,4} = \infty$ for a pinhole camera model. This equation is also used for two-bounce indirect illumination. For path $(\boldsymbol{x}_0, \boldsymbol{x}_j, \boldsymbol{x}_3)$ where $j = \{1, 2\}$, d is obtained as follows:

$$d_{0,j} = d_{j,0} = N(p_1(\boldsymbol{x}_j) + p_2(\boldsymbol{x}_j)) \frac{(\boldsymbol{n}_j \cdot \boldsymbol{\omega}_{j,3})}{\|\boldsymbol{x}_j - \boldsymbol{x}_3\|^2}, \quad (15)$$

$$d_{3,j} = d_{j,3} = N(p_1(\boldsymbol{x}_j) + p_2(\boldsymbol{x}_j)) \frac{(\boldsymbol{n}_j \cdot \boldsymbol{\omega}_{j,0})}{\|\boldsymbol{x}_j - \boldsymbol{x}_0\|^2}. \quad (16)$$

From Equation (13), $\hat{d}$ is given as:

$$\hat{d}_j = \frac{1}{\frac{1}{d_{j,0}} + \frac{1}{d_{j,3}}}, \quad \hat{d}_3 = d_{3,j}. \quad (17)$$

An appropriate value m can be determined from the weight of reflection. Cancelling $D(\theta)$ from the BRDF, it can be obtained as $m = m' f(\boldsymbol{x}, \boldsymbol{\omega}, \boldsymbol{\omega}')(\boldsymbol{n} \cdot \boldsymbol{\omega}')/D(\theta)$, where m' is the user-specified parameter. In this paper, we use $m' = 16$.

6 Results

In the following, we show the results rendered at real-time frame rates with our system on an AMD Radeon HD 6990 830 MHz. As mentioned before, all models and light sources can be fully dynamic except environment maps. All images are rendered at 1920×1088 pixels and 2×2 supersampling. The resolution of the head pointer buffer of a ray-bundle is 256×256 pixels. We allocate 20 M bytes for the node buffer. The resolution of shadow maps for VPLs is 256×256 pixels. We compute reflective shadow maps and shadow maps of direct illumination with cube mapping. These resolutions are 256×256 and 1024×1024 pixels for each side, respectively. The direct shadow lookup is smoothed using a 16-tap percentage-closer filter. The implementation detail is described in the supplemental material.

Direct Illumination of An Environment Map Figure 6 shows the results of direct illumination of an environment map. In this scene, all objects are animated. Global ray-bundles have an advantage over shadow maps. Shadow maps can test only one direction, therefore the PDF of uniform sampling is $p_r(\boldsymbol{\omega}) = 1/4\pi$. Although ray-bundles require search cost, the same quality is achieved with a smaller number of samples compared with shadow maps. In addition, global ray-bundles can compute indirect illumination in contrast to shadow maps.

Our Bidirectional Path Tracing Figure 7 shows the rendered images of an animated scene. VPLs and our path tracing produce artifacts due to the failure of the sampling strategy. On the other hand, our bidirectional path tracing produces high-quality images. This is efficient, because we must sample a huge number of VPLs to reduce the spike artifacts if only VPLs are used. Moreover, our solution computes two-bounce indirect illumination (bottom of Figure 7 (c)(f)). This improves image quality for complex scenes.

Artifact Suppression Figure 8 shows the results of artifact suppression for Kelemen and Szirmay-Kalos's glossy reflection model [Kelemen and Szirmay-Kalos 2001] with a phong distribution function. VSLs produce disk-shaped artifacts (Figure 8 (c)). They can be removed by using a number of samples at the expense of rendering time. On the other hand, our method produces high-quality results without sacrificing time (Figure 8 (d)). In addition, the implementation is simpler. Although the distance between the VPL and the shading point is out of consideration, our technique is effective because our bidirectional path tracing does not exhibit a singularity in radiance.

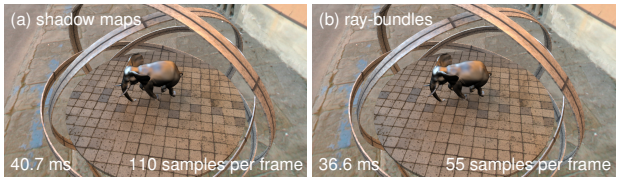


Figure 6: *Direct illumination of an environment map (87 k triangles scene). Global ray-bundles test two directions per sample, whereas shadow maps can test only one direction.*

Table 1: *Computation time*

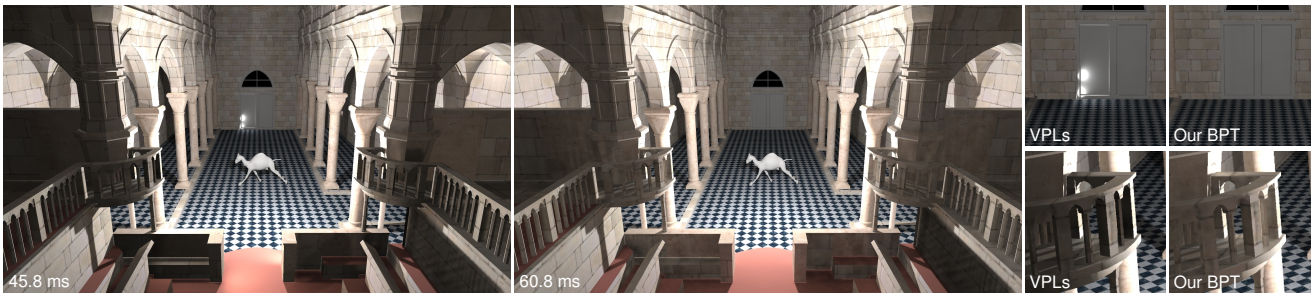
Procedure	time (ms)
Create G-Buffer	6.0
Create Reflective Shadow Maps	2.7
Create Shadow Maps	2.9
Split	1.6
Compute Bounds of Ray-bundles	0.4
Create Ray-Bundle ×21	15.4
Create VPL Shadow Map ×21	29.9
Compute Bidirectional Path Tracing ×21	5.5
Gather	1.0
Filter	8.4
Output	3.6
Others	0.4
Total	77.8

Computation Time and Quality Figure 9 shows the comparison between our result and classic path tracing solution [Kajiya 1986] of an animated scene. Our solution is over 6000 times faster than the classic solution. Furthermore, it sufficiently approximates the reference image. Instant radiosity produces detectable local error such as darkening by clamping, whereas our solution does not. Table 1 shows the computation time of each procedure (see the supplemental material) for the animated scene (Figure 9). All unlisted procedures are represented by "Others". The main bottlenecks of our system are "Create Ray-Bundle" and "Create VPL Shadow Map", i.e., visibility tests.

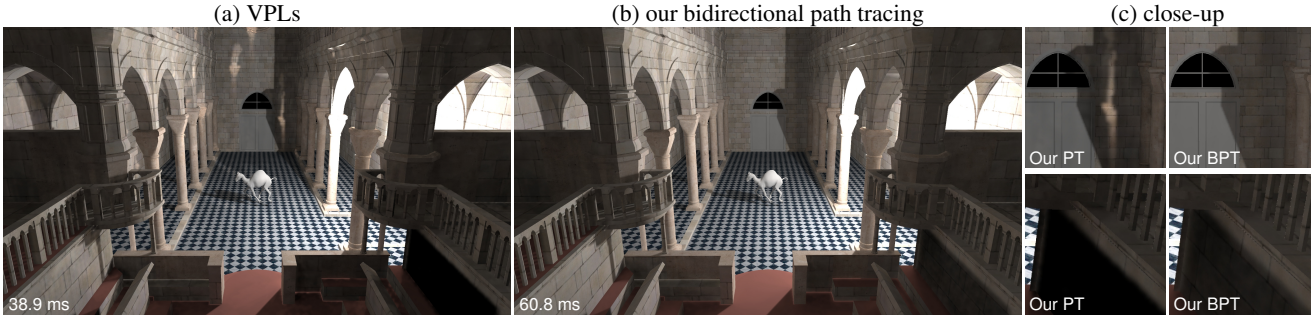
Tessellation Hardware tessellation can be used in our system because all visibility tests are done by DirectX 11 rasterization pipeline. Figure 10 shows the result of phong tessellation [Boubekeur and Alexa 2008]. The tessellation of DirectX 11 provides an arbitrary displacement by a domain shader. Our system can reuse techniques developed for a rasterization pipeline.

7 Discussion

Multi-Bounce If a scene is small, our system can be extended to multi-bounce rendering at the expense of memory usage. This is easily done by reusing global ray-bundles. For multi-bounce rendering, we create all global ray-bundles preliminary to solve the light transport problem instead of the iterative algorithm. If the ray-bundles contain an entire scene, we can randomly select a single ray-bundle and reuse it for the next bounce (Figure 11). This approach computes an arbitrary number of interreflections without additional visibility tests. As shown in Figure 12, we compute four-bounce path tracing with small overhead. However, this approach is unsuitable for large scenes, because the resolution of global ray-bundles is limited by memory and performance.

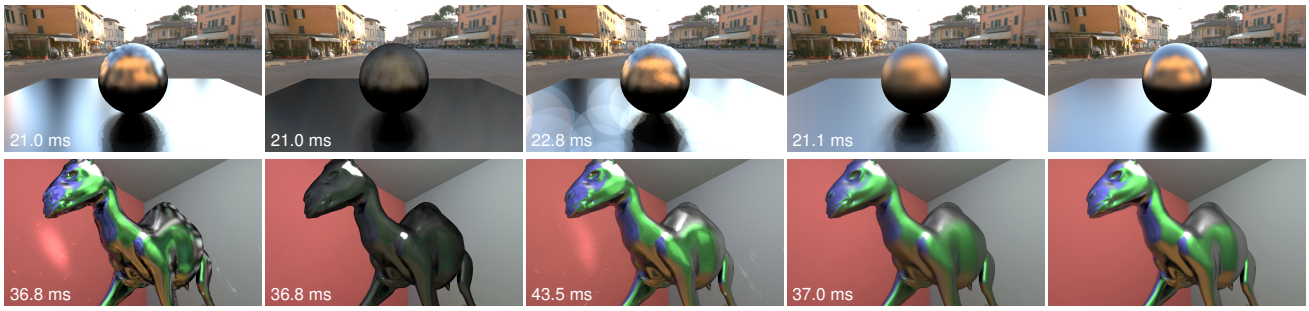


(a) VPLs (b) our bidirectional path tracing (c) close-up



(d) our path tracing (e) our bidirectional path tracing (f) close-up

Figure 7: *The comparison of (a) VPLs, (d) our path tracing, and (b)(e) our bidirectional path tracing in an animated scene (122 k triangles, 21 VPLs and 21 ray-bundles per frame). The upper and the lower are of a different frame. In this figure, VPLs and our path tracing compute one-bounce indirect illumination. On the other hand, our bidirectional path tracing computes two-bounce indirect illumination. (c) shows the close-up of (a) and (b). (f) shows the close-up of (d) and (e).*



(a) no suppression (b) clamping (c) VSLs (d) our method (e) ground truth

Figure 8: *The comparison of (a) no suppression, (b) clamping, (c) VSLs (maximum 100 samples per VSL), (d) our method, and (e) the ground truth. Upper column: a glossy sphere illuminated by an environment map (762 triangles, $\alpha = 200$, 55 ray-bundles per frame). Lower column: a glossy camel and a point light source in a Cornell box (44 k triangles, $\alpha = 255$, 21 VPLs and 21 ray-bundles per frame). The ground truth images are rendered with 75 k VPLs and 75 k ray-bundles per frame. The radius of the VSL is determined using the incoming or outgoing density described in Section 5.2 instead of using a k-nearest neighbor algorithm. Artifacts occur due to the variance of Monte Carlo integration because the same random numbers are used for all pixels in our implementation.*

(a) classic path tracing (b) our solution (c) difference

Figure 9: *The comparison of (a) classic path tracing on a CPU (2048 samples per pixels) [Kajiya 1986] and (b) our solution (174 k triangles scene, 21 VPLs and 21 ray-bundles per frame). Although our solution is several orders of magnitude faster, this is close to the reference image. The difference is due to clamping in two-bounce indirect illumination, the artifact suppression for glossy surfaces, spatio-temporal interleaved sampling, precision of textures, or number of samples. However, this error is hard to detect.*

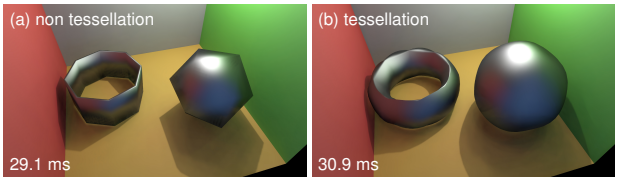


Figure 10: *Phong tessellation (96 triangles scene). The tessellation stage in the rasterization pipeline enables an arbitrary displacement with a simple implementation.*

Computational Cost There are $O(mn)$ connecting edges with m-bounce light subpaths and n-bounce camera subpaths in ray tracing based bidirectional path tracing. Therefore, for ray tracing based methods, visibility tests of all edges require $O(mn)$ rays. Since this is computationally expensive, Russian roulette is preferred. On the other hand, our visibility tests are approximated by rasterization as creating shadow maps and global ray-bundles. The shadow maps are shared by connecting edges. Hence, all tests are done by $O(m+n)$ rasterizations. This is effective even for only two bounces. For example, in Figure 3, the visibility from the light source $\boldsymbol{x}_0$ is calculated per frame, and the visibility from the VPL $\boldsymbol{x}_1$ and path $(\boldsymbol{x}_3, \boldsymbol{x}_2)$ are calculated in each iteration. Then, five visibility tests are done by only two rasterizations per iteration. Furthermore, if global ray-bundles are reused for multi-bounce rendering as previously mentioned, we can compute an arbitrary number of interreflections with $O(m)$ rasterizations. Since the visibility test is the main bottleneck of global illumination, this is an advantage of our approximate bidirectional path tracing.

8 Limitations and Future Work

BRDF The materials are limited to diffuse and rough glossy surfaces. This paper introduced a simple artifact suppression technique in Section 5, but important material cues might be lost by blurring some of sharp reflections. Many samples are needed to accurately render glossy BRDFs with a spiky lobe. This is also true for VSLs. Rendering sharp caustics is difficult for similar reasons. Another limitation of our artifact suppression is loss of illumination depending on BRDFs. For some BRDFs including Kelemen and Szirmay-Kalos's model, the albedo is determined according to the roughness parameters. It is, however, hardly detectable in practice as shown in Figure 8 (d)(e).

Memory Usage Our global ray-bundles inherit the problem of linked-list for order independent transparency. The memory usage of the linked-list is unpredictable. Hence, excessive memory has to be allocated to avoid overflow.

Resolution The limited resolution of ray-bundles causes artifacts as shadow mapping, especially for large scenes. Many anti-aliasing methods for a large environment have been developed for shadow mapping [Wimmer et al. 2004; Martin and Tan 2004; Lloyd et al. 2006; Lauritzen et al. 2011]. While the artifacts cannot be removed completely, these methods are also useful. One difference between shadow maps and global ray-bundles is that the directions of global ray-bundles are distributed omnidirectionally. Hence, methods that depend on a light direction [Wimmer et al. 2004; Martin and Tan 2004] may not work well. Another difference is that ray-bundles must contain the entire scene when multi-bounce path tracing is computed. We are currently working on this problem.

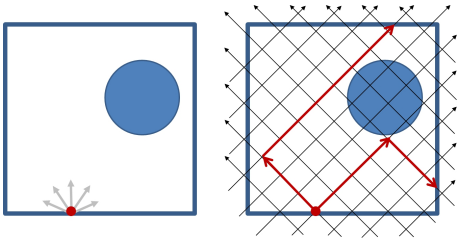


Figure 11: *Multi-bounce path tracing by reusing omnidirectional global ray-bundles.*

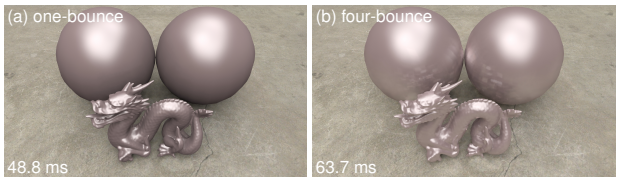


Figure 12: *Rendered images by path tracing via global ray-bundles (204 k triangles scene, 55 ray-bundles per frame). Multi-bounce is computed in a time efficient manner.*

9 Conclusion

We have presented a new robust real-time global illumination system that can handle fully dynamic scenes. Our system approximates bidirectional path tracing using VPLs and global ray-bundles with a DirectX 11 capable GPU. Additionally, we reduce spike artifacts on glossy surfaces with a small overhead. This system produces high-quality indirect illumination from dynamic point light sources. Image based lighting is also supported. Our system is easy to implement because all visibility tests are done only by the hardware rasterizer.

Acknowledgements

The light probe images and the polygon models are courtesy of Paul Debevec, Marko Dabrovic, Frank Meinl, Robert W. Sumner, Jovan Popovic, and the Stanford 3D Scanning Repository. The authors would like to thank Takashi Sugata for his textures, and Toshiya Hachisuka, Remi Driancourt, Arun Mehta, Tomohiko Mukai, Junko Asakura and Duncan Tebbs for valuable comments. We also thank anonymous reviewers for their helpful suggestions.

References

ASHIKHMIN, M., AND SHIRLEY, P. 2000. An anisotropic phong brdf model. *Journal of Graphics Tools 5*, 25–32.

BOUBEKEUR, T., AND ALEXA, M. 2008. Phong tessellation. *ACM Trans. Graph. 27*, 141:1–141:5.

CALLAHAN, S. P., IKITS, M., COMBA, J. L. D., AND SILVA, C. T. 2005. Hardware-assisted visibility ordering for unstructured volume rendering. *IEEE Transactions on Visualization and Computer Graphics 11*, 285–295.

CARPENTER, L. 1984. The a-buffer, an antialiased hidden surface method. *SIGGRAPH Comput. Graph. 18*, 103–108.

CHRISTENSEN, P. H. 1999. Faster photon map global illumination. *Journal of Graphics Tools 4*, 3, 1–10.

DACHSBACHER, C., AND STAMMINGER, M. 2005. Reflective shadow maps. In *Proc. of I3D 2005*, 203–213.

DAVIDOVIČ, T., KŘIVÁNEK, J., HAŠAN, M., SLUSALLEK, P., AND BALA, K. 2010. Combining global and local virtual lights for detailed glossy illumination. *ACM Trans. Graph. 29*, 143:1–143:8.

DEBEVEC, P. 2005. A median cut algorithm for light probe sampling. In *ACM SIGGRAPH 2005 Posters*.

EVERITT, C. 2001. Interactive order-independent transparency. Tech. rep., NVIDIA Corporation.

HACHISUKA, T. 2005. High-quality global illumination rendering using rasterization. In *GPU Gems 2*. Addison-Wesley Professional, ch. 38, 615–634.

HAŠAN, M., KŘIVÁNEK, J., WALTER, B., AND BALA, K. 2009. Virtual spherical lights for many-light rendering of glossy scenes. *ACM Trans. Graph. 28*, 143:1–143:6.

HERMES, J., HENRICH, N., GROSCH, T., AND MUELLER, S. 2010. Global illumination using parallel global ray-bundles. In *Vision, Modeling and Visualization*, 65–72.

HERZOG, R., EISEMANN, E., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2010. Spatio-temporal upsampling on the gpu. In *Proc. of I3D 2010*, 91–98.

KAJIYA, J. T. 1986. The rendering equation. *SIGGRAPH Comput. Graph. 20*, 143–150.

KELEMEN, C., AND SZIRMAY-KALOS, L. 2001. A microfacet based coupled specular-matte brdf model with importance sampling. In *Eurographics Short Presentations*, 25–34.

KELLER, A. 1997. Instant radiosity. In *Proc. of ACM SIGGRAPH '97*, 49–56.

KNECHT, M. 2009. *Real-Time Global Illumination using Temporal Coherence*. Master's thesis, Vienna University.

KOLLIG, T., AND KELLER, A. 2006. Illumination in the presence of weak singularities. In *Monte Carlo and Quasi-Monte Carlo Methods 2004*. Springer-Verlag, 245–257.

KŘIVÁNEK, J., FERWERDA, J. A., AND BALA, K. 2010. Effects of global illumination approximations on material appearance. *ACM Trans. Graph. 29*, 112:1–112:10.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1993. Bi-directional path tracing. In *Proc. of Compugraphics '93*, 145–153.

LAFORTUNE, E. P., AND WILLEMS, Y. D. 1994. Using the modified phong reflectance model for physically based rendering. Tech. rep., Report CW197, Department of Computer Science, K.U.Leuven.

LAINE, S., SARANSAARI, H., KONTKANEN, J., LEHTINEN, J., AND AILA, T. 2007. Incremental instant radiosity for real-time indirect illumination. In *Proc. of EGSR 2007*, 277–286.

LAURITZEN, A., SALVI, M., AND LEFOHN, A. 2011. Sample distribution shadow maps. In *Proc. of I3D 2011*, 97–102.

LLOYD, B., TUFT, D., YOON, S.-E., AND MANOCHA, D. 2006. Warping and partitioning for low error shadow maps. In *Proc. of EGSR 2006*, 215–226.

MARTIN, T., AND TAN, T.-S. 2004. Anti-aliasing and continuity with trapezoidal shadow maps. In *Proc. of EGSR 2004*, 153–160.

NEHAB, D., SANDER, P. V., LAWRENCE, J., TATARCHUK, N., AND ISIDORO, J. R. 2007. Accelerating real-time shading with reverse reprojection caching. In *Proc. of Graphics Hardware 2007*, 25–35.

NOVÁK, J., ENGELHARDT, T., AND DACHSBACHER, C. 2011. Screen-space bias compensation for interactive high-quality global illumination with virtual point lights. In *Proc. of I3D 2011*, 119–124.

RITSCHEL, T., GROSCH, T., KIM, M. H., SEIDEL, H.-P., DACHSBACHER, C., AND KAUTZ, J. 2008. Imperfect shadow maps for efficient computation of indirect illumination. *ACM Trans. Graph. 27*, 129:1–129:8.

RITSCHEL, T., ENGELHARDT, T., GROSCH, T., SEIDEL, H.-P., KAUTZ, J., AND DACHSBACHER, C. 2009. Micro-rendering for scalable, parallel final gathering. *ACM Trans. Graph. 28*, 132:1–132:8.

RITSCHEL, T., EISEMANN, E., HA, I., KIM, J. D., AND SEIDEL, H.-P. 2011. Making imperfect shadow maps view-adaptive: High-quality global illumination in large dynamic scenes. *Comput. Graph. Forum 30*, 2258–2269.

SBERT, M. 1996. *The Use of Global Directions to Compute Radiosity - Global Monte Carlo Techniques*. PhD thesis, Catalan Technical University.

SEGOVIA., B., IEHL., J.-C., MITANCHEY., R., AND PÉROCHE., B. 2006. Bidirectional instant radiosity. In *Proc. of the 17th Eurographics Workshop on Rendering*, 389–398.

SEGOVIA., B., IEHL., J.-C., MITANCHEY., R., AND PÉROCHE., B. 2006. Non-interleaved deferred shading of interleaved sample patterns. In *Proc. of Graphics Hardware 2006*, 53–60.

SZIRMAY-KALOS, L., AND PURGATHOFER, W. 1998. Global ray-bundle tracing with hardware acceleration. In *Rendering Techniques '98*, 247–258.

TOKUYOSHI, Y., SEKINE, T., AND OGAKI, S. 2011. Fast global illumination baking via ray-bundles. In *SIGGRAPH Asia 2011 Technical Sketches*.

VEACH, E., AND GUIBAS, L. J. 1994. Bidirectional estimators for light transport. In *Proc. of Eurographics Rendering Workshop*, 147–162.

VEACH, E., AND GUIBAS, L. J. 1995. Optimally combining sampling techniques for monte carlo rendering. In *Proc. of SIGGRAPH '95*, 419–428.

WIMMER, M., SCHERZER, D., AND PURGATHOFER, W. 2004. Light space perspective shadow maps. In *Proc. of EGSR 2004*, 143–151.

YANG, J. C., HENSLEY, J., GRÜN, H., AND THIBIEROZ, N. 2010. Real-time concurrent linked list construction on the gpu. *Comput. Graph. Forum 29*, 1297–1304.