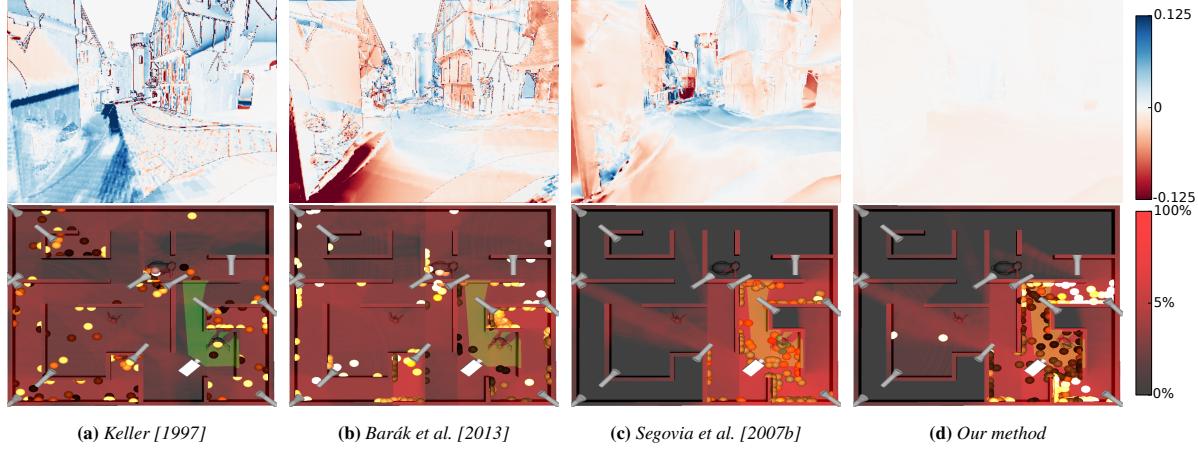


# Sequential Monte Carlo Instant Radiosity

Peter Hedman<sup>1,2,3</sup> Tero Karras<sup>1</sup> Jaakko Lehtinen<sup>1,4</sup>

<sup>1</sup>NVIDIA <sup>2</sup>University College London <sup>3</sup>University of Helsinki <sup>4</sup>Aalto University



**Figure 1:** *Top:* Temporal stability. The images show the difference between two consecutive frames in the Citadel scene (© Epic Games) where the light is moving. Our method (right) keeps the illumination stable. *Bottom:* An overhead view of a scene, the light sources, the camera and the virtual point lights (VPLs). The red tint indicates the fraction of VPLs illuminating a given point. The view frustum is denoted by the green/yellow region. Our method and Segovia et al. [2007b] only place VPLs where they illuminate surfaces visible to the camera.

## Abstract

Instant Radiosity and its derivatives are interactive methods for efficiently estimating global (indirect) illumination. They represent the last indirect bounce of illumination before the camera as the composite radiance field emitted by a set of virtual point light sources (VPLs). In complex scenes, current algorithms suffer from a difficult combination of two issues: it remains a challenge to distribute VPLs in a manner that simultaneously gives a high-quality indirect illumination solution for each frame, and does so in a temporally coherent manner. We address both issues by building, and maintaining over time, an adaptive and temporally coherent distribution of VPLs in locations where they bring indirect light to the image. We introduce a novel heuristic sampling method that strives to only move as few of the VPLs between frames as possible. The result is, to the best of our knowledge, the first interactive global illumination algorithm that works in complex, highly-occluded scenes, suffers little from temporal flickering, supports moving cameras and light sources, and is output-sensitive in the sense that it places VPLs in locations that matter most to the final result.

**Keywords:** instant radiosity, global illumination

**Concepts:** •Computing methodologies → Rendering; Ray tracing; Visibility;

## 1 Introduction

Global illumination, the infinitely complex interplay of light and the environment, is a key component in perceived image realism. Physically-based rendering algorithms simulate this process numerically. Unfortunately, most robust algorithms are currently too slow for real-time rendering of scenes of realistic extent and complexity: performance requirements severely limit the number of samples available, which results in visually unacceptable artifacts. The rich literature on interactive global illumination algorithms aims to circumvent these issues by trading variance for bias; intuitively, a smooth and temporally stable result is perceptually preferable to high variance.

We argue, that in addition to the obvious goal of being as close to ground truth as possible, the key desiderata for an interactive global illumination algorithm are

1. Low variance in space: little noise,
2. Low variance in time: temporal coherence,
3. Support for multi-bounce illumination,
4. Support for moving cameras and light sources,
5. Computing only what matters to the final image and
6. Support for moving geometry and arbitrary materials.

Desideratum 1, little noise in image space, has received an enormous amount of attention in the literature. We use the standard interleaved sampling technique as a component in our method [Keller and Heidrich 2001]. Temporal coherence (Desideratum 2) is also a well-studied problem; several image-space temporal filtering techniques share values across frames by reprojection, effectively raising the sampling rate considerably. In the interactive setting, the combination of all the desiderata have been left with significantly less attention than the individual ones.

This is an author-prepared preprint.  
The definitive version appears in the ACM Digital Library (dl.acm.org).

The key challenge in large and highly occluded scenes, such as complete building models or typical computer game levels, is that most light paths do not contribute to the image. Focusing computational resources on things that will affect the final image (Desideratum 5) is in such cases hard, because one needs to determine what parts of the scene illuminate geometry visible to the camera. Our results show that not accounting for precise visibility severely degrades quality in these settings (Figure 1). Finally, supporting full, multi-bounce global illumination (Desideratum 3) renders convenient indirect light path parametrizations based on surfaces visible to the light source, e.g. Reflective Shadow Maps [Dachsbacher and Stamminger 2005], unusable.

In this paper, we describe an algorithm that, to our best knowledge, simultaneously achieves goals 1-5 in diffuse scenes to a greater extent than previous techniques by extending the well-known Instant Radiosity algorithm [Keller 1997], which approximates the illumination in the image with *virtual point lights* (VPLs), to better adapt to challenging view-light configurations in a temporally coherent manner. Our main contribution is a VPL sampling algorithm that:

- Distributes VPLs according to the amount of light they bring the image without de-emphasizing small regions in the image (Desideratum 5),
- Minimizes temporal flickering by keeping the VPL distribution stable even if illumination changes and the camera is moving (Desiderata 2, 4),
- Supports multi-bounce indirect illumination (Desideratum 3).

Support for fully dynamic scenes and arbitrary materials (Desideratum 6) remains future work.

## 2 Previous work

Interactive global illumination is a diverse field of study too large to be fully reviewed here. The focus of this paper is to reduce the temporal variance of Instant Radiosity [Keller 1997] and place the VPLs where they are most needed for the camera view, therefore we concentrate on the most closely related work. Refer to the survey by Dachsbacher et al. [2014] for a comprehensive overview of VPL-based methods and to the survey by Ritschel et al. [2012] for a broader selection of algorithms that simulate global illumination.

**Rendering with VPLs.** The brightness  $I_p$  of a pixel  $p$  in the image  $\mathbb{P}$  is determined by the measurement equation as the integral

$$I_p = \int_{\mathcal{P}} f_p(\bar{\mathbf{z}}) d\bar{\mathbf{z}} = \int_{\mathcal{P}_l} \int_{\mathcal{P}_c} f_p(\bar{\mathbf{x}}, \bar{\mathbf{y}}) d\bar{\mathbf{x}} d\bar{\mathbf{y}} \quad (1)$$

over all light transport paths  $\bar{\mathbf{z}} \in \mathcal{P}$  [Veach and Guibas 1997]. We write each path  $\bar{\mathbf{z}}$  as the concatenation  $\bar{\mathbf{z}} = (\bar{\mathbf{x}}, \bar{\mathbf{y}})$  of a 2-vertex camera path  $\bar{\mathbf{x}} \in \mathcal{P}_c$  and a  $(N - 2)$ -vertex light path  $\bar{\mathbf{y}} \in \mathcal{P}_l$ . VPL-based methods sample one set of  $I$  light paths (or VPLs)  $(\bar{\mathbf{y}}_1, \dots, \bar{\mathbf{y}}_I)$  to estimate the value of all pixels in the image. A set of  $J$  camera paths (or *pixel samples*)  $(\bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_J)$  is then sampled for each pixel  $p$  to form the unbiased estimate

$$I_p \approx \frac{1}{I} \frac{1}{J} \sum_{i=1}^I \sum_{j=1}^J \frac{f_p(\bar{\mathbf{x}}_j, \bar{\mathbf{y}}_i)}{p(\bar{\mathbf{x}}_j)p(\bar{\mathbf{y}}_i)}. \quad (2)$$

Many-lights methods efficiently render images illuminated by a large number of point lights. Lightcuts [Walter et al. 2005] hierarchically clusters together point lights estimated to be non-essential when computing the brightness of a pixel. These methods are orthogonal to our work as they do not specify how to generate the

point lights. While we have evaluated our algorithm in an interactive setting, any many-light rendering method can be used with our method to efficiently render pictures in an offline setting.

Reflective Shadow Maps (RSMs) is a fast method to simulate single-bounce indirect illumination by placing VPLs at every texel in a shadow map [Dachsbacher and Stamminger 2005]. Approximations can be made to reduce the time spent computing visibility [Ritschel et al. 2008]. These are orthogonal to our contribution as our focus is on how to generate the VPLs. Performance can also be increased with Interleaved Sampling [Keller and Heidrich 2001; Wald et al. 2002; Segovia et al. 2006], where adjacent pixels compute the indirect illumination using disjoint subsets of VPLs. We use a similar method to compute the illumination from the VPLs.

A limitation of VPL-based methods is the singularity that occurs when VPLs and pixel samples lie close to each other or due to the narrow emission profile of VPLs on surfaces with glossy materials. We avoid this by clamping the maximum contribution of a VPL. More specialized methods, e.g. recursively sampling paths when clamping occurs [Kollig and Keller 2004] or virtual spherical lights [Hašan et al. 2009], can be used to increase quality.

**VPL sampling algorithms.** The original Instant Radiosity algorithm [Keller 1997] emits photons from the light sources and deposits a VPL at every surface interaction. This is problematic for large scenes, as it does not concentrate the VPLs in regions where they bring indirect light to the image. This issue is addressed by generating VPLs that are distributed according to the total brightness they cause in the image. Segovia et al. [2007a] achieve this by resampling the VPLs from a larger set of candidates. Georgiev and Slusallek [2010] use rejection sampling. Segovia et al. [2007b] place VPLs using a variant of Metropolis Light Transport [Veach and Guibas 1997]. Simon et al. [2015] sample VPLs from the product of two complementary photon maps [Jensen 2001], one generated from the light sources and one from the camera. Ignoring the visibility between pixel samples and VPLs, Ritschel et al. [2011] present a fast method to resample VPLs to approximately match the amount of light they bring the image. While these techniques improve the quality of the indirect illumination in individual frames, they do not provide any means to keep it temporally stable. Consequently, they cannot be used in an interactive setting as the number of VPLs needed to keep the temporal noise at acceptable levels becomes prohibitive.

**Temporal coherence** is an important aspect of interactive rendering [Scherzer et al. 2012]. In the context of VPL-based rendering, Laine et al. [2007] achieve temporally stable indirect illumination by only moving a few VPLs each frame. However, in addition to only supporting a single indirect bounce, their method is not suited for large scenes as it samples the VPL oblivious of the camera. Knecht et al. [2010] improve the stability of the indirect illumination with temporal reprojection filtering. As they do not specify how to generate the VPLs, their method is complementary to ours. Hašan et al. [2008] group point lights into clusters and reuse shaded results from the clusters over multiple frames in an animated video. As their method relies on a priori knowledge of the animation it cannot be used in an interactive setting. Wald et al. [2003] enforce temporal coherence by fixing the random number sequence used to generate the VPLs. While more VPLs are allocated to sources that bring more indirect light to the image, the distribution suffers if a single source – e.g. the sun – illuminates most of the scene. The method cannot fully ensure temporal stability with moving lights.

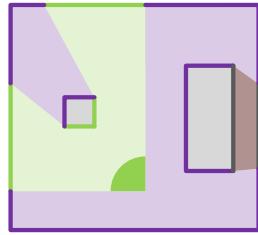
Prutkin et al. [2012] and Barák et al. [2013] devise methods to improve the temporal stability of the VPL sampling method described by Ritschel et al. [2011]. As both methods use RSMs to gener-

ate the VPLs, they only support single-bounce indirect illumination. They also ignore visibility when resampling the VPLs, which degrades the quality of the VPL distribution in heavily occluded scenes. Most importantly, as both methods enforce temporal coherence *in the RSMs* rather than in world space, they cannot ensure temporally stable illumination if a light source moves. Prutkin et al. [2012] cluster the VPLs to form area lights and achieve temporal stability by seeding the clustering algorithm with the area light centers from the preceding frame. Their method is unsuitable for walkthroughs as they only account for the camera during initialization. Barák et al. [2013] use Quasi-Monte Carlo and Metropolis-Hastings sampling to resample in a temporally stable fashion. We compare against this method in our experiments.

### 3 Method

Our two-fold goal is to distribute VPLs in the scene in a way that results in a high-quality diffuse indirect illumination field on the surfaces visible to the camera and evolve the distribution frame-to-frame in a manner that minimizes temporal flickering.

Taking inspiration from Segovia et al. [2007a], we achieve our first goal by only placing VPLs *on surfaces indirectly visible to the camera*, i.e. surfaces that can be seen from at least one of the surfaces in the image. (Naturally, only these surfaces can cast indirect light onto the image.) The inset shows directly visible surfaces in green, indirectly visible surfaces in purple and other surfaces in gray. To further improve quality, we place more VPLs on surfaces that reflect a lot of light; a type of importance sampling that improves quality in the brighter areas of the image at the expense of the darker areas. In our comparisons, we show that it is essential to consider indirect visibility this way; techniques that strive for view-adaptivity *without* accounting for visibility may use the VPL budget in a way that leaves the image quality severely degraded.



Because the camera (and potentially light sources) are moving, the VPL distribution needs to adapt to the current view-light configuration. This makes temporal coherence a potential issue. Our second goal, stability over frames, is achieved by allowing only a fixed number of VPLs to move between consecutive frames. Concretely, the majority of VPLs from the previous frame survive into the next frame, but ones that are determined the least useful — e.g. not indirectly visible any more, or in a location where VPLs are oversampled w.r.t. the indirect illumination — are discarded. New VPLs are then sampled in a manner that attempts to maintain a distribution proportional to the strength of indirect illumination (see above). This sample-evolve-resample approach is essentially a form of *particle filter*, a subset of Sequential Monte Carlo methods designed for online sampling of evolving distributions [Cappe et al. 2007].

As our sampling process is a combination of sequential resampling and discard heuristics, it is impossible to derive an exact probability density for the resulting VPLs. We address this by computing their empirical density using  $k$ -nearest neighbor density estimation. This renders our algorithm biased, but it remains consistent: increasing the number of VPLs (while decreasing their clamping) tends to the correct solution.

The remainder of the section details the specifics of these steps.

#### 3.1 Target Distribution for VPLs

We begin by expanding the measurement contribution function  $f_p$  in Equation 2. Let  $\mathbf{x}$  be the 3D position of the “primary hit”, i.e. the last vertex of the camera path  $\bar{\mathbf{x}}$ . Similarly, let  $\mathbf{y}$  be the first vertex of the light path  $\bar{\mathbf{y}}$ . The Monte Carlo estimate for pixel  $p$  is

$$I_p \approx \frac{1}{I} \frac{1}{J} \sum_{i=1}^I \sum_{j=1}^J \frac{W(\bar{\mathbf{x}}_j \leftarrow \mathbf{y}_i) G(\mathbf{x}_j \leftrightarrow \mathbf{y}_i) L(\mathbf{x}_j \leftarrow \bar{\mathbf{y}}_i)}{p(\bar{\mathbf{x}}_j)p(\bar{\mathbf{y}}_i)}, \quad (3)$$

where  $W(\bar{\mathbf{x}} \leftarrow \mathbf{y})$  is the importance of a camera path  $\bar{\mathbf{x}}$  towards the point  $\mathbf{y}$  (including the bidirectional scattering distribution function at  $\mathbf{x}$ ),  $G(\mathbf{x} \leftrightarrow \mathbf{y})$  is the geometry term between the two points (including visibility) and  $L(\mathbf{x} \leftarrow \bar{\mathbf{y}})$  is the radiance reflected from the last vertex of the light path  $\bar{\mathbf{y}}$  towards the point  $\mathbf{x}$ .

The variance of the resulting image can be reduced by using importance sampling for placing the VPLs, i.e. finding a suitable  $p(\bar{\mathbf{y}})$ . However, it is not immediately clear what the best importance sampling distribution is. It is common practice to distribute VPLs proportionally to the total power they bring to the image, as done by e.g. Segovia et al. [2007b] and Simon et al. [2015]. This is problematic for temporal coherence: small but brightly illuminated regions in the image are prone to flickering (this is evident in the accompanying videos; cf. in particular the corridor sequences in Soda Hall).

We find it better to distribute the VPLs according to a metric agnostic to the image-space size of the regions they illuminate. A good choice would be to base sampling on the maximum pixel brightness caused by the VPL in the image, i.e. have  $p(\bar{\mathbf{y}})$  be proportional to the maximum of the product of the  $W$ ,  $G$  and  $L$  terms from Equation (3) over all pixels. As this is hard to achieve in general scenes, we make two assumptions. First, we treat all indirectly visible surfaces as diffuse, so that VPLs emit uniformly in all directions, i.e.  $L(\mathbf{x} \leftarrow \bar{\mathbf{y}}) = L(\bar{\mathbf{y}})$ . Second, we replace the view importance and geometry terms with unity, i.e. we ignore the materials of the primary hits and their geometric configuration w.r.t. indirectly visible surfaces and instead treat all of them with equal importance. Together, these assumptions mean we strive to distribute VPLs according to their radiosity:

$$p(\bar{\mathbf{y}}) \propto L(\bar{\mathbf{y}}). \quad (4)$$

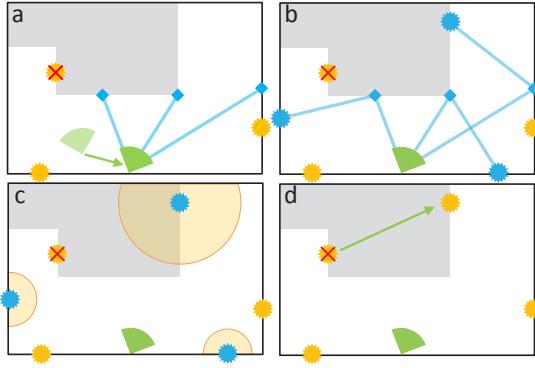
The next sections show how this is achieved in practice. While our results show this leads to high-quality solutions in diffuse scenes, we believe existing work on efficiently estimating illumination bounds can be used for tightening these for more general scenes. We leave this as future work.

**VPLs vs. Light Paths.** Until now, we have assumed, like many Instant Radiosity algorithms, that a VPL is synonymous with a single light path  $\bar{\mathbf{y}}$ . Similarly to Segovia et al. [2007a] and Simon et al. [2015], we instead marginalize over all light paths that end up at the same VPL location  $\mathbf{y}$ . That is, instead of having the VPL emit proportional to a single path from the source, we compute its total radiosity as an average over several paths. From now on, we consider a VPL just to be the point  $\mathbf{y}$ . Given unbiased estimates for the radiosities of the VPLs, the estimate

$$I_p \approx \frac{1}{I} \frac{1}{J} \sum_{i=1}^I \sum_{j=1}^J \frac{W(\bar{\mathbf{x}}_j \leftarrow \mathbf{y}_i) G(\mathbf{x}_j \leftrightarrow \mathbf{y}_i) L(\mathbf{y}_i)}{p(\bar{\mathbf{x}}_j)p(\mathbf{y}_i)}, \quad (5)$$

where  $p(\mathbf{y}_i)$  is the probability density of the VPL point  $\mathbf{y}_i$  with respect to the surface area measure, is still unbiased.<sup>1</sup> To simplify

<sup>1</sup>The details are easy but technical and non-essential here.



**Figure 2: Overview.** VPLs shown with yellow. (a) At every frame, we first sample points (in blue) on the surfaces visible to the camera (in green). We also invalidate some VPLs as described in Sections 3.2.1 and 3.2.4 (shown with a red cross). (b) We generate candidate VPLs (in blue) by tracing rays from the directly visible points. (c) We then estimate the radiosities and probability densities of these candidates. (d) Finally, we replace the invalidated VPLs with the candidates that have the strongest intensities.

notation, we define the *intensity*  $I(\mathbf{y})$  of a VPL  $\mathbf{y}$  to be the ratio

$$I(\mathbf{y}) = \frac{L(\mathbf{y})}{p(\mathbf{y})} \quad (6)$$

of its estimated radiosity and the probability density of the point  $\mathbf{y}$ . Note that if the VPLs are distributed according to their radiosities, i.e.  $p(\mathbf{y}) \propto L(\mathbf{y})$ , then all VPLs will have the same intensity.

### 3.2 Sequential Sampling Algorithm

Our goal is to incrementally maintain a temporally coherent VPL distribution with uniform intensities when moving from one frame to the next. Our main idea is to incrementally replace VPLs that have the smallest intensities with new VPLs that have larger intensities. This drives the distribution towards uniform intensity.

The key challenge is a combination of two factors. First, light and camera movement cause the target *probability density function* (PDF) for VPL placement to change over time: moving lights change the radiosity of indirectly visible surfaces and camera movement changes the set of indirectly visible surfaces itself. Second, to maintain temporal stability, we wish to re-use as many VPLs from the previous frame as possible. This brings up a crucial point. As we migrate a VPL distribution from a previous frame, we have no tractable way of analytically computing the PDF  $p(\mathbf{y}_i)$  in the new frame: it depends on the sampling decisions made in all previous frames. Instead, we rely on the insight that the probability density on a surface patch can be estimated as the fraction of the VPLs that lie on it normalized by its area. Consequently, we approximate the PDFs of all VPLs using k-nearest neighbor density estimation (Section 3.2.3). This allows us to adapt to any given VPL distribution and drive it towards a uniform intensity distribution.

**Algorithm.** See Figure 2 for an overview and Algorithm 1 for a pseudo-code description of the sampling algorithm. Here,  $I$  is the total VPLs budget and  $M_{\max}$  is the number of VPLs allowed to move between frames. A lower  $M_{\max}$  results in faster updates, but gives the sampler less room to produce a high-quality distribution. Unless mentioned otherwise, we set  $I = 2048$  and  $M_{\max} = 16$ .

When a frame begins, we need to decide which of the existing

---

### Algorithm 1 Sequential Monte Carlo Instant Radiosity

---

```

1:  $\mathcal{Y} \leftarrow \text{INITIALIZEVPLS}()$ 
2: for each new frame do
3:    $\mathcal{X} \leftarrow \text{RAYCASTFROMCAMERA}()$ 
4:    $\text{INVALID} \leftarrow \mathcal{Y} \setminus \text{INDIRECTLYVISIBLEVPLS}(\mathcal{X}, \mathcal{Y})$ 
5:    $\hat{\mathcal{Y}} \leftarrow \text{RAYCASTFROMPOINTS}(\mathcal{X})$ 
6:    $\text{ESTIMATERADIOSITIES}(\hat{\mathcal{Y}})$ 
7:    $\text{ESTIMATEPROBABILITYDENSITIES}(\hat{\mathcal{Y}})$ 
8:    $M_{\min} \leftarrow \text{DETECTUNDERSAMPLEDREGIONS}(\mathcal{Y} \setminus \text{INVALID})$ 
9:   while  $|\text{INVALID}| < M_{\min}$  do
10:     $\mathbf{y}_{\min} \leftarrow \text{FINDSMALLESTINTENSITY}(\mathcal{Y} \setminus \text{INVALID})$ 
11:     $\text{INVALID} \leftarrow \text{INVALID} \cup \{\mathbf{y}_{\min}\}$ 
12:    $\mathcal{Y} \leftarrow \mathcal{Y} \setminus \text{CLAMPSIZE}(\text{INVALID}, M_{\max})$ 
13:   while  $|\mathcal{Y}| < I$  do
14:      $\mathbf{y}_{\max} \leftarrow \text{FINDLARGESTINTENSITY}(\hat{\mathcal{Y}})$ 
15:      $\hat{\mathcal{Y}} \leftarrow \hat{\mathcal{Y}} \setminus \{\mathbf{y}_{\max}\}$ 
16:      $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\mathbf{y}_{\max}\}$ 
17:    $\text{ESTIMATERADIOSITIES}(\mathcal{Y})$ 
18:    $\text{ESTIMATEPROBABILITYDENSITIES}(\mathcal{Y})$ 
19:    $\text{RENDERFRAME}(\mathcal{Y})$ 

```

---

VPLs, if any, to replace. To ensure that the VPLs are located on the indirectly visible surfaces, we first invalidate all VPLs that cannot be verified as indirectly visible (lines 3-4, Section 3.2.1).

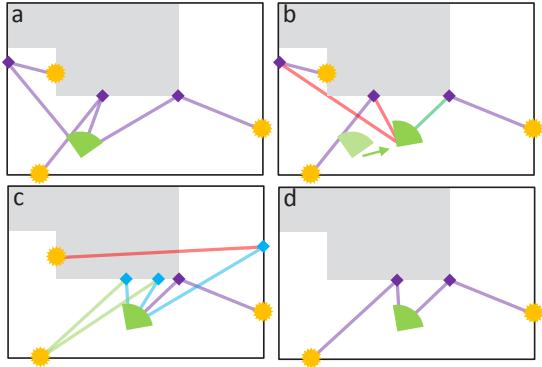
We then generate a set of candidate VPLs by tracing rays from the directly visible surfaces (line 5) and estimate their intensities by performing density estimation and computing their radiosities (lines 6 and 7, Sections 3.2.2 and 3.2.3). The radiosities and probability densities of VPLs carried over from the previous frame are not re-estimated at this point. We then use these intensities to detect undersampled regions (line 8) — bright indirectly visible surfaces where VPL density is too low — and iteratively fill them in. This happens by removing VPLs from oversampled regions and generating new VPLs in undersampled regions (lines 9-16). Oversampling and undersampling are easily detected by examining the intensities of the VPLs (Section 3.2.4). Finally, the intensities are re-estimated for the remaining VPLs and the image is rendered.

We initialize the algorithm by sampling VPLs on the indirectly visible surfaces identically to how we generate candidate VPLs. To ensure that the VPLs are well distributed, we run 100 iterations of the algorithm before rendering the first frame.

#### 3.2.1 Resolving indirect visibility

We estimate indirect visibility using a mail-boxing scheme that associates each VPL with a point visible to both the camera and the VPL, cf. Figure 3. When a new VPL is created we first associate it with the point it was generated from. When the camera moves, we can easily validate a VPL as indirectly visible by checking if its associated point is still visible to the camera. A similar scheme has been used for occlusion culling [Aila and Miettinen 2004].

As this is not an exhaustive visibility test, some of the remaining invalid VPLs may be indirectly visible. We approximately correct for this with an iterative scheme that employs the directly visible points we sample at the start of each frame. At each iteration, we check the indirect visibility for any remaining invalid VPLs by casting rays from these points towards  $V$  randomly chosen invalid VPLs. If a ray cast succeeds, we mark the chosen VPL as valid and associate it with the point that chose it. Note that VPLs can only be chosen or marked as valid between iterations as the ray casts are performed in parallel. In our experiments we set  $V = 16$  and perform two



**Figure 3: Verifying indirect visibility.** (a) Each VPL (in yellow) is associated with a point (in purple) visible to both the camera and the VPL. (b) When the camera moves, some VPLs can be validated as indirectly visible if their associated points are still seen by the camera (shown with green). (c) We validate more VPLs by casting rays from points directly visible to the camera (in blue) towards randomly chosen VPLs. (d) If a ray cast succeeds, we mark the target VPL as valid and associate it with the point that cast the ray.

iterations per frame. While this does not ensure an exact result, the rest of our method is robust to occasional erroneous classifications.

### 3.2.2 Estimating Radiosities

Any global illumination algorithm can be used to estimate the VPL radiosities. In our experiments, we use 16 samples from a path tracer with next event estimation [Kajiya 1986]. To ensure that the estimate remains temporally coherent, we make sure that the path tracer generates similar paths for the same VPL throughout the lifetime of the VPL. We achieve this by initializing the pseudo-random number used by path tracer with a VPL-specific seed.

### 3.2.3 Estimating probability densities

We approximate the density of VPLs on the surfaces with  $k$ -nearest neighbor ( $k$ NN) density estimation based on the same assumptions used by photon mapping [Jensen 2001]. That is, instead of performing a costly search along the surfaces in the scene, we assume that the  $k$ -nearest neighbors we find with an unconstrained 3D search all lie on the same, planar surface. We then approximate the probability density of a VPL  $\mathbf{y}$  w.r.t. the surface area measure as

$$p(\mathbf{y}) \approx \frac{k}{I} \frac{1}{r_k(\mathbf{y})^2 \pi}, \quad (7)$$

where  $I$  is the total number of VPLs and  $r_k(\mathbf{y})$  is the distance to the  $k$ th nearest neighbor of  $\mathbf{y}$  among the *current* VPLs in  $\mathcal{Y}$ .

We use a different  $k$  when estimating PDFs for candidate samples (line 7) and for rendering (line 18). For candidate VPLs, the probability densities are used to decide which candidates are selected. We set  $k = 1$  to ensure that candidates close to any current VPLs are unlikely to be selected. This prevents VPLs from clumping together and encourages blue-noise properties in the resulting distribution.

When computing intensities for final rendering, we want to maximize quality of the estimated probability density. To achieve this, we choose  $k$  in a way that can be shown to result in a consistent estimate [Loftsgaarden and Quesenberry 1965]:

$$k = \max(1, \left\lfloor \frac{\sqrt{I}}{32} \right\rfloor). \quad (8)$$

### 3.2.4 Detecting undersampled regions

To detect undersampled regions on indirectly visible surfaces (regions with too few VPLs in relation to their brightness), we observe that the estimated probability density  $p(\mathbf{y})$  of any VPL which lies in such a region will be small relative to its radiosity  $L(\mathbf{y})$ . As a consequence, each such candidate will have an uncharacteristically large intensity  $I(\mathbf{y})$  compared to the current set of VPLs  $\mathcal{Y}$ . Based on this, we compute the number  $M_{min}$  of VPLs to invalidate by counting how many candidates have stronger intensities than a large majority of the current VPLs. Specifically, we set

$$M_{min} = \min(M_{max}, M_{strong}), \quad (9)$$

where  $M_{strong}$  is the number of candidates with stronger intensities than 95% of the current VPLs in  $\mathcal{Y}$ . When there is room in the total VPL budget, choose the highest-intensity candidate VPLs to replace them (lines 9-16).

## 4 Implementation

All steps of our algorithm are performed on the GPU in either OpenGL or CUDA. We use the OptiX Prime library [Opt 2015] to cast the rays necessary for sampling new VPL candidates and verifying indirect visibility and in our CUDA path tracer we use it for estimating VPL radiosities. We use CUDA to perform the  $k$ NN search with the help of a bounding volume hierarchy (BVH) constructed using the fast tree building algorithm by Karras [2012].

Similarly to Segovia et al. [2006], we compute the illumination from the VPLs using deferred rendering and interleaved sampling [Keller and Heidrich 2001]. We partition the image into tiles of size  $4 \times 4$  and use a different subset of the VPLs to compute the indirect illumination for each pixel in a tile. We remove the resulting structured noise with a cross-bilateral filter whose weights are determined by the dot product between the normals and the distance between the world-space locations of the pixel samples.

We use cube maps resampled into paraboloid shadow maps to test the visibility between pixel samples and VPLs, removing the need for finely tessellated geometry [Brabec et al. 2002]. To increase performance, we translate and rotate the cube map so that three of its faces cover the entire hemisphere, making sure to keep the center of projection at the VPL location. Similarly to Laine et al. [2007], we lazily update the shadow maps only for VPLs that move between frames. This saves computation with the caveat that dynamic objects cannot be accounted for in the shadow maps.

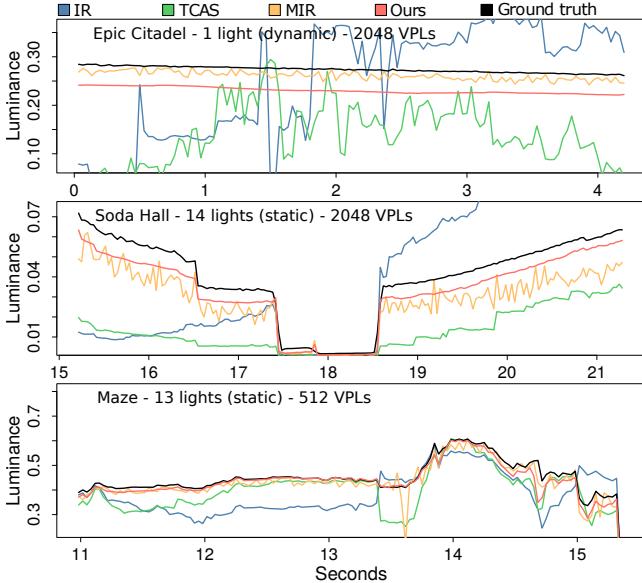
## 5 Experiments

We now evaluate the quality and performance of our method. To examine how the VPL distribution affects the quality of the final animated result, we compare against representative previous algorithms while keeping the VPL budget, method for determining visibility (standard shadow mapping) and noise filtering (interleaved sampling) the same between the algorithms — only the positions and intensities of the VPLs change. To clearly compare temporal stability, we do not use temporal reprojection filtering on the indirect illumination. We quantify error by tracking the absolute error and temporal stability of the resulting indirect illumination. We also analyze how the error of our method diminishes as more VPLs are used. Finally, we study the performance of our method with a timing breakdown. We encourage the reader to watch the supplemental videos for a thorough qualitative comparison of these methods.

We simulate three bounces of indirect illumination in all experiments. To make the results agree better with perceived quality, we

Scene	Length	Lights	#VPLs	MIR	Average $ E $			Average $ E' $			
					IR	TCAS	Ours	MIR	IR	TCAS	Ours
Soda Hall	30 s	14 (static)	2048	0.0265	0.0553	0.0530	<b>0.0169</b>	0.0098	0.0077	0.0057	<b>0.0043</b>
Epic Citadel	23 s	1 (static)	2048	0.0492	0.0827	0.0626	<b>0.0432</b>	0.0241	0.0365	0.0228	<b>0.0211</b>
Maze	22 s	13 (static)	512	0.0381	0.1122	0.0528	<b>0.0325</b>	0.0273	0.0176	0.0124	<b>0.0087</b>
Crytek Sponza	20 s	1 (dynamic)	2048	0.0286	<b>0.0215</b>	0.0355	0.0302	0.0073	0.0044	0.0027	<b>0.0007</b>
Epic Citadel	21 s	1 (dynamic)	2048	<b>0.0246</b>	0.0896	0.0650	0.0304	0.0219	0.0325	0.0198	<b>0.0068</b>

**Table 1: Quantitative comparison.** The rows summarize the overall error (Average  $|E|$ ) and the temporal instability (Average  $|E'|$ ) of the competing VPL sampling methods for all tracked pixels in a sequence. The smallest error is in bold.



**Figure 4: Example sequences.** The plots display how the luminance for a single pixel as estimated by the comparison methods changes over time in three of our sequences.

perform all measurements after applying the Reinhard tone mapping operator [Reinhard et al. 2002]. It is well known that VPL algorithms have severe issues with high-frequency textures. In all experiments, we use an untextured version of the scene for the VPL generation and use average material colors as albedos. The path traced reference images use the same approximation.

**Comparison methods.** We compare our method to Instant Radiosity (IR) [Keller 1997] (baseline method, no view adaptivity and no special consideration for temporal coherence), Metropolis Instant Radiosity (MIR) [Segovia et al. 2007b] (high quality VPL distribution but no temporal coherence) and the method by Barák et al. (TCAS) [2013] (view adaptivity, temporal coherence, but limited to a single indirect bounce and does not account for indirect visibility). Of these, MIR can be seen as a gold standard for single frame image quality as it produces, independently for each frame, a VPL distribution that matches the power brought to the image.

Since IR generates VPLs by emitting photons from the light sources, it may not use the full VPL budget if some photons exit the scene before depositing the second or third bounce VPLs. This is visible in Epic Citadel where many photons leave the scene early, exacerbating the structured noise caused by interleaved sampling.

MIR relies on light paths that are distributed according to the radiance they bring the camera, which the original implementation generates using Metropolis Light Transport [Veach and Guibas 1997].

As the visibility and the materials are not complicated in our test scenes, we instead generate these paths by resampling paths from our path tracer. We use a resampling rate of 256 : 1.

Our version of TCAS differs from the original implementation in that we perform 16 iterations of Metropolis-Hastings sampling (instead of five) to improve the resulting VPL distributions. In our tests, five iterations produced unacceptable quality in larger scenes.

**Distribution quality.** The image quality and temporal stability of our method is best demonstrated in the supplemental videos. We also perform a numerical comparison by tracking the luminance of 8 pixels over time in five sequences (see Figure 4 for examples). We compute ground truth values for these pixels using 692100 samples from our path tracer. Table 1 shows how closely the indirect illumination follows the reference with the average absolute error  $|E|$  over all pixels in the sequences. To capture perceptually displeasing temporal high frequency changes in the indirect illumination (“flickering”), we compute for each pixel the finite difference

$$E' = \frac{\Delta \text{Error}}{\Delta t} \quad (10)$$

of the error with respect to time. In the table we show the average absolute  $E'$  over all pixels in the sequences. Note that  $E'$  shows how the error fluctuates in the images, not in the scene and can be non-zero even if the VPLs stay fixed, but the camera moves.

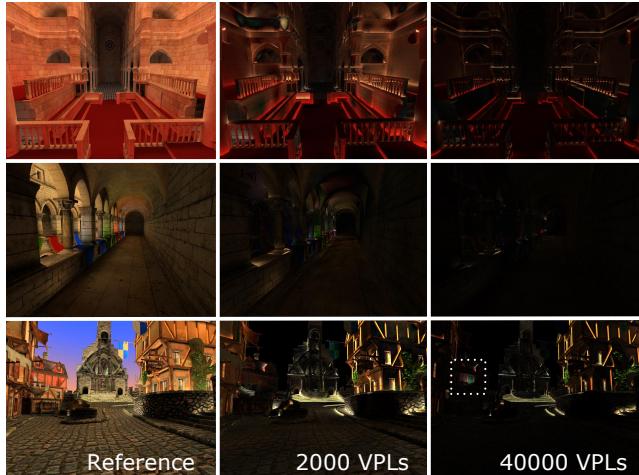
We chose our test scenes to reflect the difficult occlusion characteristics typical for video game levels. This especially true for Soda Hall and Epic Citadel, where only a small portion of the scene is visible at any given time and the set of indirectly visible surfaces (where VPLs can bring light to the image) is small in comparison to the entire scene. This highlights the importance of accounting for precise indirect visibility. First, in Soda Hall even the view-adaptive TCAS algorithm places VPLs in brightly illuminated rooms behind walls where they do not affect the image. Second, bright sunlight illuminates a majority of the Epic Citadel scene. Here, when the camera lies on the street level, the set of illuminated surfaces is much larger than the set of indirectly visible surfaces. Only MIR and our method produce distributions that match the image well.

As the supplemental videos, Figure 4 and Table 1 all show, our method outperforms the others in terms of temporal stability, more so when the light sources are moving (e.g. Epic Citadel). We see that the improved temporal stability does not affect the image quality, as the error made by our method is comparable to MIR. To see the effect of our importance sampling distribution (Section 3.1), we encourage the reader to compare our method to MIR in the Soda Hall sequence and observe new rooms appearing at the end of long corridors. Note that TCAS naturally has a larger error than the other methods as it does not support multi-bounce indirect illumination.

**Error analysis.** We compare images rendered by our method with different VPLs budgets with references produced by our path

Scene	# Tris	# VPLs	Sampling VPLs				Rendering			
			Cand. VPLs & bookkeeping	Indirect visibility	Estimating radiosities	kNN	Total	Shadow maps	Interl. shading	Total
Epic Citadel	373K	2048	2.7	1.4	11.6	2.3	18.0	7.6	19.7	27.3
Soda Hall	551K	2048	2.0	1.0	8.1	1.8	12.9	14.8	19.5	34.3
Crytek Sponza	262K	2048	2.3	1.5	9.5	2.0	15.3	5.7	24.4	30.1
Maze	471K	512	1.8	0.9	6.9	1.6	11.2	11.3	10.7	22.0
										33.2

**Table 2:** Breakdown of the time spent computing indirect light in the scenes. All timings are in milliseconds. **Left to right:** the scene; number of triangles; number of VPLs; generating candidate VPLs, detecting undersampled regions and replacing invalidated VPLs; verifying indirect visibility; estimating VPL radiosities; kNN density estimation (including building the BVH); total time for VPL sampling; rendering shadow maps; interleaved shading (including building the G-Buffer); total time for final rendering; total time for indirect illumination.



**Figure 5:** The error of the indirect illumination produced by our method with different VPL budgets. **Left to right:** Reference image, 2× absolute error (2K VPLs), 2× absolute error (40K VPLs). **Top to bottom:** Sibenik Cathedral, Crytek Sponza, Citadel (© Epic Games). Most of the remaining error with 40K VPLs is attributed to energy loss due to clamping. However, as we discuss in the text, the intensities of some VPLs in Citadel have been over-estimated.

tracer with 32768 samples per pixel. To focus on the error of our sampling method, we only render indirect illumination and evaluate the visibility between VPLs and pixel samples with ray casts. Figure 5 shows that error is much reduced when the VPL budget is increased to 40000, where most of the remaining error is explained by aliasing and energy loss due to clamping. However, one surface in Epic Citadel still exhibits noticeable error. The probability density of strong VPLs illuminating it has been underestimated as they lie on a thin structure (a washing line). This is to be expected, as kNN density estimation cannot detect such high-frequency variation in the PDF with a restricted sample budget.

**Rendering performance.** We measure performance by rendering images in  $1920 \times 1080$  on a PC with an Intel i7 4820K CPU, 16 GB RAM and a NVIDIA GTX Titan X GPU. See Table 2 for a timing breakdown we performed by measuring the speedup from disabling each component of the algorithm and scaling the results to match the total frame time. Most time is spent on interleaved sampling, rendering shadow maps and computing VPL radiosities. Less time can be spent computing radiosities by reducing the number of indirect bounces. Interleaved sampling can be sped up using larger tiles with the risk of introducing structured noise or oversmoothing the indirect illumination. The performance of most components is linear w.r.t. the number of VPLs, except for the kNN search (a neg-

ligible part of the frame time) and rendering shadow maps which depends on the budget  $M_{max}$  of VPLs that can move each frame.

Table 2 shows that more than half of the frame time is spent on parts shared by all VPL rendering algorithms (rendering shadow maps and interleaved sampling). Therefore, other sampling methods can be at most twice as fast as ours. As the comparison methods do not limit how many VPLs move between frames, rendering shadow maps significantly limits their performance [Laine et al. 2007], favoring our technique. We feel these bounds are tight enough to not warrant a study using highly tuned implementations. Hence we do not compare timings and we use the same VPL budget for all methods instead of performing equal-time comparisons.

## 6 Conclusions

The experiments show that our method renders temporally coherent multi-bounce diffuse indirect illumination at real-time rates in large and heavily occluded scenes. We achieve this with an incremental, adaptive sampling algorithm that reuses VPLs from previous frames and places new VPLs to equalize their intensities while accounting for the movement of the camera and the lights. We believe this is the first time this has been shown in highly occluded scenes with a single-frame illumination quality on par with the state of the art that does not enforce temporal coherence. Naturally, as our algorithm takes up the entire GPU, usage in the tightly constrained performance envelopes of actual products is currently infeasible.

In contrast to many prior VPL rendering algorithms, we use GPU ray tracing in the sampling process. Given the recent advent of efficient BVH builders and their increasing support for dynamic scenes, we believe this trade-off is worthwhile particularly considering the future. We feel it is an interesting avenue to combine our sequential sampler with techniques for VPL importance sampling [Walter et al. 2005; Popov et al. 2015]. In fact, an early version of this work features an initial study in this direction [Hedman 2015].

Although we have not shown results with dynamic scenes, our method (like prior methods) is able to illuminate moving objects even if they do not affect the light flow. This is due to the way we resolve VPL-pixel visibility, as shadow maps with moving objects need to be re-rendered every frame. This limitation can be overcome by approximating visibility for the shadow maps [Ritschel et al. 2008], although many shadow maps must still be rendered every frame. In contrast, for a ray tracer it suffices to rebuild the acceleration structure once every frame. We see a strong trend towards replacing shadow maps with ray casts, even in dynamic scenes.

## Acknowledgements

We thank Timo Aila, Samuli Laine, Tobias Ritschel, Perttu Hämäläinen, Clément Godard, Cornelius Iliescu, Moos Hueting, Daniel Worrall and Stefan Garbin for brainstorming and helpful

comments. Thanks to Lars Hedman for help with the text and his support. Thanks to Epic Games for Citadel and Erik Sintorn for help with it. Thanks to Marko Dabrovic for Sibenik, Frank Meinl for Crytek Sponza, the U.C. Berkeley WALKTHRU project for Soda Hall and the Stanford Computer Graphics Lab for Bunny and Armadillo. This work was partly funded by the EU project CR-PLAY (no 611089) www.cr-play.eu and EPSRC EP/K023578/1.

## References

- AILA, T., AND MIETTINEN, V. 2004. dPVS: An occlusion culling system for massive dynamic environments. *IEEE Comput. Graph. Appl.* 24, 2, 86–97.
- BARÁK, T., BITTNER, J., AND HAVRAN, V. 2013. Temporally coherent adaptive sampling for imperfect shadow maps. *Comput. Graph. Forum* 32, 4, 87–96.
- BRABEC, S., ANNEN, T., AND SEIDEL, H.-P. 2002. Shadow Mapping for Hemispherical and Omnidirectional Light Sources. In *Proc. Computer Graphics International*, 397–408.
- CAPPE, O., GODSILL, S., AND MOULINES, E. 2007. An overview of existing methods and recent advances in sequential Monte Carlo. *Proc. IEEE* 95, 5 (May), 899–924.
- DACHSBACHER, C., AND STAMMINGER, M. 2005. Reflective Shadow Maps. In *Proc. I3D '05*, 203–231.
- DACHSBACHER, C., KŘIVÁNEK, J., HAŠAN, M., ARBREE, A., WALTER, B., AND NOVÁK, J. 2014. Scalable realistic rendering with many-light methods. *Comput. Graph. Forum* 33, 1, 88–104.
- GEORGIEV, I., AND SLUSALLEK, P. 2010. Simple and Robust Iterative Importance Sampling of Virtual Point Lights. In *Proc. Eurographics (short papers)*.
- HAŠAN, M., VELÁZQUEZ-ARMENDARIZ, E., PELLACINI, F., AND BALA, K. 2008. Tensor Clustering for Rendering Many-Light Animations. In *Proc. EGSR '08*, 1105–1114.
- HAŠAN, M., KŘIVÁNEK, J., WALTER, B., AND BALA, K. 2009. Virtual spherical lights for many-light rendering of glossy scenes. *ACM Trans. Gr.* 28, 5 (Dec.), 143:1–143:6.
- HEDMAN, P. 2015. *Sequential Monte Carlo Instant Radiosity*. Master’s thesis, University of Helsinki.
- JENSEN, H. W. 2001. *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters, Ltd., Natick, MA, USA.
- KAJIYA, J. T. 1986. The rendering equation. In *Proc. SIGGRAPH '86*, 143–150.
- KARRAS, T. 2012. Maximizing Parallelism in the Construction of BVHs, Octrees, and k-d Trees. In *Proc. High-performance Graphics*, 33–37.
- KELLER, A., AND HEIDRICH, W. 2001. Interleaved sampling. In *Proc. EGWR '01*, 269–276.
- KELLER, A. 1997. Instant Radiosity. In *Proc. SIGGRAPH '97*, 49–56.
- KNECHT, M., TRAXLER, C., MATTIAUSCH, O., PURGATHOFER, W., AND WIMMER, M. 2010. Differential Instant Radiosity for mixed reality. In *Proc. ISMAR*, 99–107.
- KOLLIG, T., AND KELLER, A. 2004. Illumination in the presence of weak singularities. In *Monte Carlo and Quasi-Monte Carlo Methods*, 245–257.
- LAINE, S., SARANSAARI, H., KONTKANEN, J., LEHTINEN, J., AND AILA, T. 2007. Incremental Instant Radiosity for Real-Time Indirect Illumination. In *Proc. EGSR'07*, 277–286.
- LOFTSGAARDEN, D. O., AND QUESENBERRY, C. P. 1965. A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics* 36, 3, 1049–1051.
2015. OptiX. <https://developer.nvidia.com/optix>.
- POPOV, S., RAMAMOORTHI, R., DURAND, F., AND DRETTAKIS, G. 2015. Probabilistic connections for bidirectional path tracing. *Comput. Graph. Forum* 34, 4.
- PRUTKIN, R., KAPLANYAN, A., AND DACHSBACHER, C. 2012. Reflective Shadow Map Clustering for Real-Time Global Illumination. In *Proc. Eurographics (short papers)*.
- REINHARD, E., STARK, M., SHIRLEY, P., AND FERWERDA, J. 2002. Photographic tone reproduction for digital images. *ACM Trans. Gr.* 21, 3 (July).
- 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. Gr.* 27, 5 (Dec.), 129:1–129:8.
- RITSCHEL, T., EISEMANN, E., HA, I., KIM, J. D. K., AND SEIDEL, H.-P. 2011. Making imperfect shadow maps view-adaptive: High-quality global illumination in large dynamic scenes. *Comput. Graph. Forum* 30, 8, 2258–2269.
- RITSCHEL, T., DACHSBACHER, C., GROSCH, T., AND KAUTZ, J. 2012. The state of the art in interactive global illumination. *Comput. Graph. Forum* 31, 1, 160–188.
- SCHERZER, D., YANG, L., MATTIAUSCH, O., NEHAB, D., SANDER, P. V., WIMMER, M., AND EISEMANN, E. 2012. Temporal coherence methods in real-time rendering. *Comput. Graph. Forum* 31, 8 (Dec.), 2378–2408.
- SEGOVIA, B., IEHL, J. C., MITANCHEY, R., AND PÉROCHE, B. 2006. Non-interleaved Deferred Shading of Interleaved Sample Patterns. In *Proc. Graphics Hardware '06*, 53–60.
- SEGOVIA, B., IEHL, J. C., MITANCHEY, R., AND PROCHE, B. 2007. Bidirectional Instant Radiosity. In *Proc. EGSR '06*, 389–397.
- SEGOVIA, B., IEHL, J.-C., AND PÉROCHE, B. 2007. Metropolis instant radiosity. *Comput. Graph. Forum* 26, 3, 425–434.
- SIMON, F., HANIKA, J., AND DACHSBACHER, C. 2015. Rich-VPLs for improving the versatility of many-light methods. *Comput. Graph. Forum* 34, 2, 575–584.
- VEACH, E., AND GUIBAS, L. J. 1997. Metropolis Light Transport. In *Proc. SIGGRAPH '97*, 65–76.
- WALD, I., KOLLIG, T., BENTHIN, C., KELLER, A., AND SLUSALLEK, P. 2002. Interactive Global Illumination Using Fast Ray Tracing. In *Proc. EGWR '02*, 15–24.
- WALD, I., BENTHIN, C., AND SLUSALLEK, P. 2003. Interactive Global Illumination in Complex and Highly Occluded Environments. In *Proc. EGRW '03*, 74–81.
- WALTER, B., FERNANDEZ, S., ARBREE, A., BALA, K., DONIKIAN, M., AND GREENBERG, D. P. 2005. Lightcuts: A scalable approach to illumination. *ACM Trans. Gr.* 24, 3 (July), 1098–1107.