Eurographics Symposium on Rendering 2013
Nicolas Holzschuch and Szymon Rusinkiewicz
(Guest Editors)

Volume 32 (2013), Number 4

Temporally Coherent Adaptive Sampling for Imperfect Shadow Maps

T. Barák, J. Bittner, V. Havran

Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic

Abstract

We propose a new adaptive algorithm for determining virtual point lights (VPL) in the scope of real-time instant radiosity methods, which use a limited number of VPLs. The proposed method is based on Metropolis-Hastings sampling and exhibits better temporal coherence of VPLs, which is particularly important for real-time applications dealing with dynamic scenes. We evaluate the properties of the proposed method in the context of the algorithm based on imperfect shadow maps and compare it with the commonly used inverse transform method. The results indicate that the proposed technique can significantly reduce the temporal flickering artifacts even for scenes with complex materials and textures. Further, we propose a novel splatting scheme for imperfect shadow maps using hardware tessellation. This scheme significantly improves the rendering performance particularly for complex and deformable scenes. We thoroughly analyze the performance of the proposed techniques on test scenes with detailed materials, moving camera, and deforming geometry.

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

1. Introduction

Global illumination algorithms provide important visual cues that add on realistic appearance of rendered scenes. Ray tracing based global illumination methods can provide high quality images, but they are too slow for real-time applications such as games. Therefore we often resort to solutions that approximate global illumination up to different extents, for example they limit the number of indirect illumination bounces or use various visibility approximations. These methods often exploit rasterization, which is easy to parallelize and heavily optimized in the GPUs. Unlike ray tracing, rasterization methods do not rely on acceleration data structures and thus they easily handle dynamic scenes. On the other hand the rasterization methods are in their basic form restricted to computing visibility for coherent groups of rays enclosed by a viewing frustum.

One solution for computing global illumination, which has been designed to exploit rasterization hardware is the *instant radiosity* method [Kel97]. The main principle of the instant radiosity and the follow-up algorithms is the use of Virtual Point Lights (VPLs) that are traced from the primary light sources. The VPLs are then used for gathering of radiance at the shaded points, while visibility of VPLs is determined using shadow maps. The creation of shadow maps and subsequent visibility lookups form a bottleneck of the algorithm and even with recent hardware we cannot achieve real-time performance for moderately complex scenes. This bottleneck has been addressed by the Imperfect Shadow Maps (ISM) algorithm [RGK*08], which uses many low resolution shadow maps each computed using only a subset of the scene geometry. In contrast to the traditional shadow maps, all ISMs are stored in one texture and they are created in one rendering pass. Even though with ISMs the instant radiosity method achieves real-time performance, the number of VPLs that can be handled in real-time is still limited, which is an issue for complex scenes. To improve the quality of ISM based instant radiosity for complex scenes Ritschel et al. [REH*11] designed view adaptive method which optimizes the set of VPLs (and ISMs) for the given frame. This method estimates the importance of possible VPL positions with respect to the rendered image and then uses the inverse transform method to sample with respect to this importance.

In this paper we present a novel algorithm based on Metropolis-Hastings sampling that allows efficient selection of VPLs for complex scenes. Our main contributions are: (1) reducing temporal artifacts caused by adaptive VPL selection using parallel Independent Metropolis-Hastings sampling, (2) fast creation of ISMs using hardware tessellation.

The paper is further structured as follows. The next section presents briefly the work related to our approach. In Section 3 we describe the outline of the algorithm. In Section 4 we present a new method for temporally coherent sampling of VPLs in the context of the imperfect shadow maps algorithm. In Section 5 we describe a novel ISM creation algorithm using GPU tessellation. In Section 6 we give results from measurement for several scenes. Finally, in Section 7 we conclude the paper.

2. Related Work

Real-time global illumination. The global illumination algorithms for interactive applications were surveyed recently in the state-of-the-art report by Ritschel et al. [RDGK12]. Here we pay attention only to the most important papers directly related to our work. The proposed algorithm builds up on the idea of instant radiosity by Keller [Kel97] that introduced the concept of many virtual point lights (abbreviated to VPL). The created VPLs are used to gather the radiance from the indirect illumination in the same way as for direct illumination, while the visibility is resolved using shadow maps. Techniques which use many virtual lights for representing illumination are generally referred to as *many-light methods* and they were recently surveyed by Dachsbacher et al. [DKH*13].

Wald [WKB*02] adopted the instant radiosity method in the context of interactive ray tracing, where visibility between VPL and shaded points is computed by tracing rays on the CPU. Wald [WBS03] also developed an importance sampling approach for large scale and possibly highly occluded scenes. This method uses sparse sampling over a small set of image pixels to compute the approximation of importance of all VPLs and the most important VPLs are selected by thresholding according to the precomputed importance. Georgiev and Slusalek [GS10] propose a tunable importance sampling of VPLs, where a single parameter can be used to select a given portion of light sources. Similarly, Dammertz et al. [DKL10] in their progressive algorithm use stochastic culling of VPLs with low importance, based on the use of Halton random generator.

GPU rendering algorithms. To allow generation of VPLs on the GPU without the necessity of using ray tracing Dachsbacher and Stamminger [DS05] introduced the concept of reflective shadow maps. To generate VPL on scene surfaces the scene is rendered from primary light sources before the gathering from the generated VPLs takes place. Ritschel et al. [RGK*08] introduced imperfect shadow maps that lift restriction on the visibility computation in instant radiosity. The visibility for shadow maps is sub-sampled by coarse shadow maps of low resolution. This allows to use more VPLs for gathering, while the error of the proposed approximation is visually acceptable. In the follow-up paper Ritschel et al. [REH*11] use importance sampling based on the inverse transform method that first constructs the cumulative distribution (CDF) and then applies a binary search. The importance of all VPLs is computed for only small set of pixels (such as 0.1%) randomly selected for each light. Another alternative, which is possibly less prone to temporal artifacts and can better handle glossy scenes is to perform fast gathering of illumination [REG*09, MW11] for which visibility is evaluated with respect to gather points rather than VPLs.

Metropolis Sampling. Veach and Guibas [VG97] presented the Metropolis Light Transport (MLT) algorithm, which computes an unbiased estimate of the rendered image. The method uses mutations of light paths within the ingenious Markov chain Monte Carlo (MCMC) algorithm proposed by Metropolis et al. [MRR*53] and generalized by Hastings [HAS70]. Metropolis-Hastings sampling allows to sample from unknown possibly multidimensional distributions without the need to compute all probabilities of the underlying state space. Szirmay-Kalos et al. [SKDP99] studied the start-up bias of MLT for different ray distributions. Ashikhmin et al. [APSS01] analyzed the variance of MLT concluding the variance is inversely proportional to the number of samples taken. In context of interactive rendering and VPL-based method Segovia et al. [SIP07a] presented the use of Multiply-Try Metropolis-Hasting (MTMH) sampling of VPLs to get such sets of VPLs that are sufficiently spatially coherent and can be ray traced with fast packet based algorithms. Another method of Segovia et al. [SIP07b] exploits MTMH to provide such a set of VPLs in which each VPL contributes the same amount of power to the rendered image. Note that an open topic of research in statistics is the usage of quasi-random sequences within the MCMC methods [CDO11].

Temporally coherent rendering algorithms. A number of rendering algorithms have been specifically targeted at rendering animations and walkthroughs. In general these methods aim either to make the rendering more efficient by exploiting frame-to-frame coherence and/or to minimize the disturbing temporal artifacts usually perceived as flickering. The temporally coherent methods were surveyed by Tawara et al. [TMD*04] in the context of offline rendering algorithms and more recently by Scherzer et al. [SYM*12] in the context of real-time rendering. Directly related to our work, Laine et al. [LSK*07] presented a method that updates only a small part of all VPLs for subsequent frames. This method reduces temporal artifacts, but it also introduces a latency in handling fast illumination changes or fast camera movement for the case of adaptive VPL sampling.

3. Algorithm Overview

Our method builds on the idea of Instant Radiosity with Imperfect Shadow Maps (ISM) [RGK*08]. We use a deferred shading pipeline and thus in the first stage we create a layered framebuffer representation of the camera view, often referred to as *g-buffer* [AMHH08]. Then we create reflective shadow map (RSM) for each primary light source [DS05]. The RSM contains geometry and material information for surfaces visible from the corresponding primary light source. Similarly to the camera view the RSMs are stored as g-buffers.

The RSMs are then used to construct the set of VPLs, which then approximate the first bounce indirect illumination. We follow the adaptive sampling approach of Ritschel et al. [REH*11] who defined an importance metric for VPLs, which is used to create the importance map (IM). The IM represents the estimated contribution of the VPL at the given scene position to the rendered image. Unlike Ritschel et al. who used the inverse transform method, we use the Metropolis-Hastings sampling to sample according to the importance map, which in turn reduces the temporal artifacts caused by VPL adaptations.

When the VPLs are established, the ISMs are created by resampling the scene geometry to points and stochastically splatting these points to the ISMs. For this step we propose to use hardware tessellation, which provides significant speedup and which is also better suited to highly dynamic and deformable scenes. The ISM splatting phase is followed by the pull-push algorithm [MKC07], which fills holes caused by undersampling.

When ISMs are created, indirect illumination is gathered by summing up the contributions of the VPLs to shaded points, while using the ISMs for resolving approximate visibility of VPLs. Each pixel uses only a subset of the VPLs in order to speed up this phase of the algorithm [RGK*08]. Note, that we also use g-buffer splitting to shuffle the shaded pixels in order to improve the coherence of shadow lookups [SIMP06]. Finally, the indirect illumination is filtered using geometry aware filter and the result is summed up with the direct illumination to finalize the rendered image. The overview of the whole algorithm and its parts which will be discussed below in the paper is shown in Figure 1.

4. Temporally Coherent VPL Sampling

4.1. View Adaptive Imperfect Shadow Maps

The ISM algorithm can handle only a limited number of VPLs (and corresponding ISMs) in order to achieve real-time performance. Thus for scenes with more complex structure it is important to allocate these VPLs so that their contribution to the image is balanced.

Ritschel et al. [REH*11] proposed to treat every RSM pixel as a *potential VPL* (pVPL). Using a random subset of P scene points visible from the camera, the importance of each pVPL is estimated as follows:

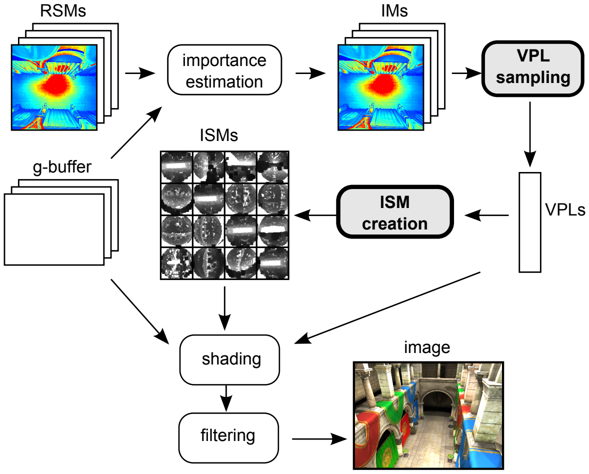


Figure 1: *Overview of the indirect illumination computation using view adaptive Imperfect Shadow Maps. The two steps of the algorithm which we address in the paper are highlighted in bold (VPL sampling, ISM creation).*

$$f(\mathbf{v}) = \sum_{k=1}^{P} I(\mathbf{v})H(\mathbf{x}_k, \omega_k, \mathbf{v}) \quad (1)$$

$$H(\mathbf{x}, \omega, \mathbf{v}) = f_r(\mathbf{x}, \omega, r(\mathbf{v}) \to \mathbf{x})G(\mathbf{x}, r(\mathbf{v})) \quad (2)$$

where $\mathbf{v} = (l, \theta, \phi)$ is the *VPL description vector*, i.e. the vector of parameters describing a VPL (l is the index of the RSM, θ and ϕ are the coordinates of the VPL in the RSM), $I(\mathbf{v})$ is the VPL intensity, $\mathbf{x}_k$ is the world space position of image sample k, ω_k is the direction vector from the camera towards $\mathbf{x}_k$, $r(\mathbf{v})$ is the position of VPL $\mathbf{v}$, $f_r(\mathbf{x}, \omega, r(\mathbf{v}) \to \mathbf{x})$ is BRDF at point $\mathbf{x}$, $G(\mathbf{x}, r(\mathbf{v}))$ is the geometry factor between point $\mathbf{x}$ and VPL $\mathbf{v}$. Note that this function does not include visibility.

The importance computed for each pVPL forms an *importance map* (IM) of the same resolution as RSM (also called Bidirectional Reflective Shadow Map [REH*11]). This map after normalization corresponds to *probability density function*. A required number of VPLs is then established by sampling according to this density. The quality of the generated VPL pattern depends on the importance function definition and also on the actual VPL sampling algorithm. Ritschel et al. used the inverse transform method [REH*11] briefly discussed in the next section.

4.2. Inverse Transform Method

Using the inverse transform method (InvTM) the estimated contributions $f(\mathbf{v})$ in the importance map are normalized in order to create a discrete probability density $p(\mathbf{v}) \propto f(\mathbf{v})$.

The normalization this $p(\mathbf{v})$ requires the knowledge of sum of the values of $f(\mathbf{v})$ over all lights l and directions (θ, ϕ).

The cumulative distribution function CDF is calculated on the graphics hardware using multiple parallel prefix scans. For our case, the CDF is first constructed over all rows of all IMs, then over the last column of all IMs and finally, the values of all lights/IMs are cumulated. This calculation requires multiple kernel/shader launches and additional memory to store the CDF.

The successive sampling phase generates three random numbers (ξ_1, ξ_2, ξ_3) with the uniform distribution in the range $[0,1]^3$. The first random number ξ_1 is used to select the light l, ξ_2 and ξ_3 are used to select the row and column of the matching IM. This is done by using three consecutive binary searches in the CDF function (see Figure 2). Another possibility is to use two random numbers and the number used for the light selection renormalize back to the unit interval as for direct lighting [SW91]. We will show the differences between both approaches in Section 6.

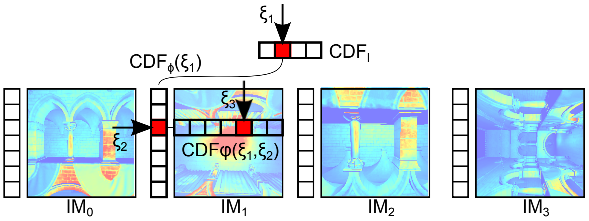


Figure 2: *Illustration of the InvTM for sampling with input of three random numbers. Each of the four IMs corresponds to one primary light source.*

For generating the random numbers (ξ_1, ξ_2, ξ_3) we use a Halton sequence evaluated directly on the GPU. Even if we use the same Halton sequence every frame, it turns out that the InvTM sampling exhibits insufficient temporal coherence. In particular even a local change in the $p(\mathbf{v})$ causes global changes in the corresponding $cdf(\mathbf{v})$s and potentially alters positions of a high number of VPLs (almost all VPLs slightly float on the surfaces). Although the VPL positions are not modified dramatically, the changes in scene geometry and materials cause sudden changes in the VPL image contribution. This is emphasized if the scene contains high geometric detail, detailed textures, or bump maps. As a result the adaptation of VPLs to the $p(\mathbf{v})$ introduces unwanted illumination flickering. In the next section we present a method which reduces the temporal artifacts, while being faster than the InvTM and simpler to implement.

4.3. Metropolis-Hastings Sampling

The Metropolis-Hastings (M-H) algorithm [HAS70] can be used to draw samples from an arbitrary probability distribution. The algorithm constructs a Markov chain and in each iteration it selects a sample that relies on the result of the previous iteration. The tentative samples are proposed using a *proposal distribution* and the algorithm decides either to accept the proposed sample or to keep the previously reported sample as the current one. After certain number of samples has been drawn the generated samples follow the target distribution. A know issue of the method is that a number of initial samples in so called *burn-in* phase will not follow the target distribution and thus they are often discarded. Alternatively we can use a compensation method which takes the accommodation phase into account. This issue is sometimes referred to as *start-up bias* [SKDP99].

A particular subclass of Metropolis-Hastings sampling are the Independent Metropolis-Hastings (IM-H) methods [Tie94]. In the IM-H methods the proposal distribution does not depend on the current state of the Markov chain. In our work we exploit the IM-H principle as this approach has several benefits for our application: (1) it is easier to control the correlation of the samples by using disjoint sets of proposals, (2) we can use a simple method to weight the samples and thus to reduce the start-up bias of the Monte Carlo estimator based on the VPLs determined by IM-H.

Traditionally the M-H algorithms are used to solve complex high dimensional integrals in cases when the InvTM method cannot be easily used. Contrary to the standard usage of M-H our primary motivation for using M-H algorithm is achieving better temporal coherence of the generated samples than the InvTM method does.

Our sampling strategy is based on parallel evaluation of N Markov chains, where N corresponds to the number of desired samples, that is the number of VPLs. From each chain we compute M samples and we only take the *last* generated sample as the representative of the target distribution. Within each chain we use a unique uniform proposal density generated using Halton sequence in the style of Wang and Hickernell [WH00]. For a chain with index j we take quasi-random numbers for seeds starting at $j + N * k$ $(0 < k < M)$, which produces disjoint proposals for each of the sequences. Together with using just one sample per chain this leads to disjoint set of generated M samples. Thus we avoid the common problem of Metropolis-Hastings sampling which is the potentially high correlation of samples, particularly the issue of one sample (VPL) being drawn multiple times.

All N chains can be evaluated in parallel using a simple algorithm, in which each chain performs M steps. The algorithm is illustrated in Figure 3 and outlined in Algorithm 1 (the use of weights $w(\mathbf{v}_j)$ will be explained in the next section). We refer to this algorithm as M-H-Our in Section 6.

4.3.1. Gathering VPL illumination

The indirect illumination at point $\mathbf{x}$ visible from the camera, which corresponds to a pixel in the image, is gathered from

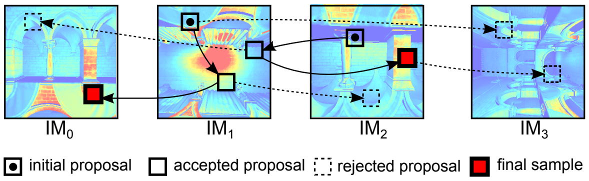


Figure 3: *Illustration of the Metropolis-Hastings VPL sampling. The figure shows two Markov chains of length $M = 5$ in four IMs. The first chain starts at IM$_1$ and two of the $M-1$ proposed moves were accepted. The result of this chain is a VPL sample at IM$_0$. The second chain starts at IM$_2$. Here again two of the proposed $M-1$ moves were accepted. The result of this chain is a VPL sample at IM$_2$.*

```
input  : $j$, $f()$, $M$
output: $\mathbf{v}_j$, $f(\mathbf{v}_j)$, $w(\mathbf{v}_j)$
begin
    $s \longleftarrow 0$;
    $rand \longleftarrow$ initialize, use $j$ as a seed;
    $\mathbf{v}_j \longleftarrow rand.nextVector3()$;
    for $i \leftarrow 1$ to $M$ do
        $\mathbf{v}_{\mathbf{new}} \longleftarrow rand.nextVector3()$;
        $a \longleftarrow f(\mathbf{v}_{\mathbf{new}})/f(\mathbf{v}_j)$;
        $s \longleftarrow s + f(\mathbf{v}_{\mathbf{new}})$;
        if $a \geq rand.nextScalar()$ then
            $\mathbf{v}_j \longleftarrow \mathbf{v}_{\mathbf{new}}$;
        end
    end
    $w(\mathbf{v}_j) \longleftarrow s/M$;
    return $\mathbf{v}_j$, $f(\mathbf{v}_j)$, $w(\mathbf{v}_j)$
end
```

Algorithm 1: Pseudocode of the Independent Metropolis-Hastings algorithm for a single VPL that runs in parallel with different sequences of random numbers for each thread.

the VPLs using the following Monte Carlo estimator:

$$L(\mathbf{x},\omega) \;\sim\; \frac{1}{N}\sum_{j=1}^{N}\frac{1}{p(\mathbf{v}_j)}I(\mathbf{v}_j)H(\mathbf{x},\omega,\mathbf{v}_j)V(\mathbf{x},r(\mathbf{v}_j))$$

where N is the number of VPLs, $I(\mathbf{v}_j)$ is the intensity of VPL $\mathbf{v}_j$, $p(\mathbf{v}_j)$ is the probability of generating $\mathbf{v}_j$, $r(\mathbf{v}_j)$ is the position of $\mathbf{v}_j$, and $V(\mathbf{x},r(\mathbf{v}_j))$ is the binary visibility function between points $\mathbf{x}$ and $r(\mathbf{v}_j)$ and $H(\mathbf{x},\omega,\mathbf{v}_j)$ is defined according to Eq. 2.

In our case the target probability density of the IM-H algorithm is proportional to the importance function $f(\mathbf{v})$ and we use the following estimator:

$$L(\mathbf{x},\omega) \sim \frac{1}{N}\sum_{j=1}^{N}\frac{w(\mathbf{v}_j)}{f(\mathbf{v}_j)}I(\mathbf{v}_j)H(\mathbf{x},\omega,\mathbf{v}_j)V(\mathbf{x},r(\mathbf{v}_j)) \qquad (3)$$

where $w(\mathbf{v}_j)$ is the weight of VPL $\mathbf{v}_j$ calculated in order to normalize the importance function $f(\mathbf{v})$ and also to reflect the below discussed issue connected with the burn-in phase of the M-H algorithm.

A common problem of M-H sampling is the burn-in phase in which the samples do not follow the target distribution. Using these samples in a Monte Carlo estimator might result in so called start-up bias of the estimator.

We use a simple strategy for reducing the start-up bias, which in the same time accounts for the normalization of the importance function $f(\mathbf{v})$. As the weight $w(\mathbf{v}_j)$ for a sample generated by chain j we use a local mean of the uniformly distributed proposals for that chain. For the case that $M = 1$ the algorithm degenerates to uniform sampling of the target distribution and the weights become $w(\mathbf{v}_j) = f(\mathbf{v}_j)$. For the case that $M \to \infty$ the weights correspond to the expected value of the importance function $f(\mathbf{v})$ and the distribution of samples $\mathbf{v}_j$ follows $f(\mathbf{v})$. Thus for very small M the generated samples $\mathbf{v}_j$ are closer to the uniform distribution, but the weights $w(\mathbf{v}_j)$ compensate for that and the estimator is normalized accordingly. Our experiments show that for $M > 4$ the samples already follow the importance function sufficiently well. It can also be shown that if $f(\mathbf{v}) \propto I(\mathbf{v}_j)H(\mathbf{x},\omega,\mathbf{v}_j)V(\mathbf{x},r(\mathbf{v}_j))$ the resulting estimator is unbiased for $M > 1$. This generally does not hold in our case, however we will show that in our target application the proposed method leads to images with error comparable to the InvTM method, while achieving lower temporal flickering.

The proposed weighting strategy resembles the method for eliminating startup bias proposed by Veach and Guibas for the Metropolis Light Transport algorithm [VG97]. Their technique draws the initial samples from a given stationary distribution and then weights the Markov chains by the image contribution function divided by the proposal distribution probability density at the initial sample points. In the results we will show that this weighting technique applied in our setting leads to higher variation of VPL intensities and thus exhibits higher temporal artifacts and higher error compared to a reference image. We refer to this variant as M-H-Single in Section 6.

Note that we use the same Halton sequence in each frame to achieve temporal coherence of the generated random numbers.

4.3.2. Performance

Unlike the InvTM which requires to store the computed CDFs the IM-H algorithm requires no additional data storage. The IM-H algorithm also avoids the CDF calculation and it does not require to calculate the total sum of importances for normalization. On the other hand the IM-H does not scale well with the length of the chains M, since $M-1$ samples are always discarded. Using the proposed method for reducing the start-up bias we however observed that very short sequences (e.g. $M > 4$) already provide high VPL adaptation to the underlying importance function (evaluation will be presented in Section 6).

5. Tessellation based ISM Rendering

5.1. Precomputed points

For the ISM creation step Ritschel et al. [RGK*08] use an approximate point representation of the scene. In the preprocessing stage, they create a point based representation of the scene by sampling the surface of the scene triangles. Later, in the ISM creation step, they use splatting of the point samples. Each point sample is splatted to a randomly chosen VPL/ISM.

This approach introduces additional complexity for moving or deforming geometry, as the point representation has to be updated with the moving or deforming triangles. This requires storing supplementary information with each point in the point-based representation, namely the index of the triangle and the barycentric coordinates of the point. To deal with deformable scenes a subset of point samples is regenerated in each frame. In the next section we propose a straightforward replacement of this approach which creates the whole point representation of the scene triangles on the fly.

5.2. Dynamic GPU tessellation

Instead of creating the point cloud off-line, we create the point representation of the scene during the ISM splatting phase by tessellation. For this task, we exploit the capabilities of modern graphics hardware that can tessellate the input geometry during the rendering stage (this functionality is called tessellation shaders in OpenGL 4.3 API).

Instead of generating finer triangle representation (which is the common use-case of the hardware tessellation), we use the *point mode tessellation* capability. During the point mode tessellation, the graphics hardware is instructed to change the primitive type from triangles to points after the tessellation stage. New point primitives are generated instead of vertices of the finer tessellated geometry. This feature allows us to transform the input triangle representation into points on the fly during the ISM splatting phase without storing the intermediate point representation.

Our ISM splatting algorithm works as follows (using the OpenGL pipeline terminology):

1. *CPU:* render the whole triangle representation of the scene with the tessellation turned on,
2. *vertex shader:* pass the vertex attributes to the next stage,
3. *tessellation control shader:* calculate the triangle surface area, set the number of points to be generated,
4. *tessellation evaluation shader:* generate a new position on the triangle, generate a random number generator seed for the next stage,
5. *geometry shader:* randomly choose VPL/ISM using the supplied seed, translate and project the splat into the selected shadow map, calculate the size of the splat,
6. *fragment shader:* store the depth into the ISM.

The process of rendering ISMs using hardware tessellation is illustrated in Figure 4.

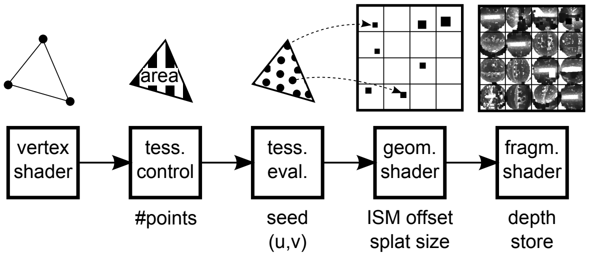


Figure 4: *Illustration of the tessellation based ISM splatting.*

The key observation behind our approach is that in common scenarios the number of required splats is much higher than the number of scene triangles. For higher quality indirect shadows, the number of required point splats becomes prohibitively high and the memory accesses can create a bottleneck in the rendering process. By omitting the need of storing and reading the large point cloud representation, we achieve a notable performance boost of this rendering stage.

Another benefit of using this approach is the additional flexibility it introduces for dynamic and deformable scenes. The proposed approach can be seamlessly connected to usual rasterization pipeline and can generate point splats directly from the deformed or moved meshes. There is also no need to maintain additional data structure containing the point splats.

6. Results

We test our implementation on a Core i7-3770, 16 GB RAM desktop computer, running 64 bit Linux operating system, equipped with GeForce GTX 470, 1280 MB GRAM graphics card with 310.32 NVIDIA driver. Rasterization uses OpenGL 4.3 API, all GPGPU tasks are implemented as GLSL compute shaders. The computation times are measured using high resolution GPU timers exported through the OpenGL timer query capability.

If it is not stated otherwise, we use the following settings in our measurements: 512 × 512 pixels output resolution, 2048 × 2048 pixels ISM resolution, 1024 VPLs are created, 128 VPLs are evaluated at each shaded pixel.

6.1. Run-time Performance

We tested the algorithm on four scenes shown in Figure 9. The timings of the individual algorithmic stages and the average frame times are shown in Table 1. The achieved frame rates are sufficient for the use in real-time applications. The time needed to create VPL by three different methods is shown in Table 2. For the InvTM method the measured times also include the CDF construction and are therefore higher

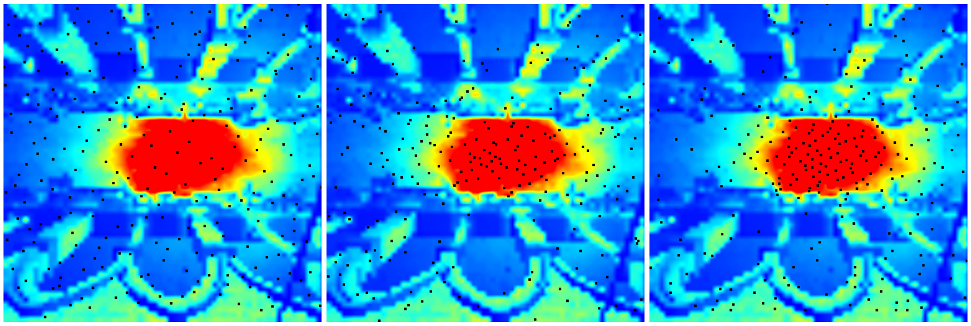


Figure 5: *The visualization of the distribution of selected VPLs in the Crytek-Sponza scene: (left) Uniform sampling, (center) Metropolis-Hastings, (right) InvTM. The intensity of indirect illumination mapped by pseudo-color from blue (low intensity) to red (high intensity). Note how both adaptive sampling methods distribute the samples according to the underlying importance function.*

than for the Uniform and M-H methods. Note that the times are almost equal for generating different number of VPLs. This follows from the fact that the GPU is capable of running more threads in parallel than the number of generated VPLs and thus in this step it is actually underutilized.

6.2. Evaluation of Image Quality and Flickering

To evaluate the frame-to-frame temporal coherence, we need a method to evaluate the amount of disturbing temporal flickering, which appears particularly during a walkthrough. Although several video-quality metrics were developed and published, e.g. [Win09, ČHM*12], they are not well suited for our evaluation for which we need a simple measure to compare the different VPL sampling techniques.

Our comparison metric is specifically designed for comparing renderings of static scenes with mostly diffuse surfaces. The radiosity solution should be temporally constant and independent of the camera movements. We therefore reproject pixels including their intensities from the previous frame to the current frame by taking their world-space coordinates from the previous frame. The *temporal image difference* (TID) is then given as a RMS value of differences among intensities of matching pixels for all consecutive pairs of frames in a walkthrough.

In Figure 5 we show the visualization of the VPL distribution for all three sampling algorithms being compared. We show the above described TID metric for the walkthrough and the RMSE against a reference walkthrough in Figure 7. The uniform sampling (VPLs are sampled uniformly from the RSMs using the same Halton sequence every frame) has the lowest temporal flickering, but the quality of the indirect illumination is lower, which leads to higher RMSE with respect to a reference. The InvTM sampling decreases RMSE, but it exhibits significant temporal flickering. Metropolis-Hastings sampling provides the good trade-off: the RMSE is slightly increased, while the temporal flickering is significantly reduced. Start-up bias depending on the M-H chain length for the tested sampling methods is shown in Figure 6.

Figure 8 shows the temporal image difference TID for varying number of VPLs and the number of VPLs used in gathering. The results are depicted for the sampling using either 3 (r3) or 2 (r2) random numbers from the Halton generator. In this context the temporal flickering is significantly lower when using 3 random numbers compared to the renormalization approach used for direct lighting [SW91].

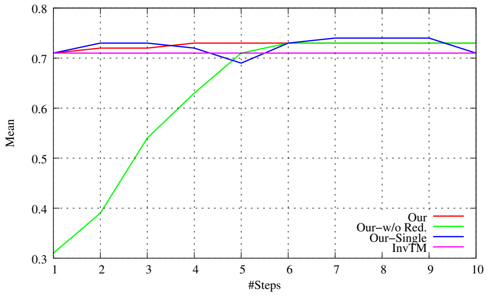


Figure 6: *Mean value of pixel intensities for various lengths of IM-H chains (i.e. number of steps) for the Crytek-Sponza scene. Note that in contrast to the IM-H without bias reduction reduction ($Our-w/o-Red.$), the proposed method (Our) successfully reduces the bias even for lower numbers of steps.*

6.3. Dynamic GPU Tessellation for ISMs

We measured the dynamic tessellation performance on a few scenes with various complexities. The rendered images for the tested scenes are shown in Figure 9. Table 3 shows the timings for the two evaluated ISM splatting methods in dependence on the number of splats N_{splats}. Our approach is 2 to 5 times faster than using the preprocessed points. Detailed comparison of splatting time for the two evaluated ISM splatting methods is shown in Figure 10.

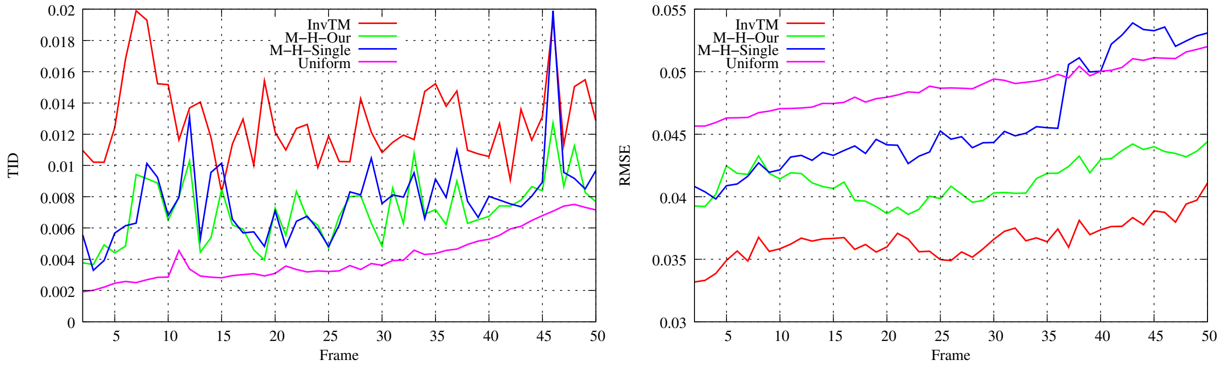


Figure 7: *Temporal image differences (left) and reference error (right) for the three compared methods (InvTM, M-H-Our, M-H-Single, Uniform). The plots show the first 50 frames of the Crytek-Sponza walkthrough using 1024 VPLs and 128 VPL evaluations per pixel.*

scene	N_L	T_{dir}	T_{ISM}	T_{indir}	T_{sum}
Conference R.	6	5.09	15.50	10.39	30.99
Sibenik Cath.	6	6.10	14.59	10.29	30.98
Crytek-Sponza	6	7.42	15.52	10.57	33.52
Armadillo	(*)	3.06	8.27	7.03	18.37

Table 1: *The timings of different algorithm phases and the total time in [ms] for four test scenes. T_{dir} is the direct illumination computed by shadow mapping, T_{ISM} covers VPL sampling and ISM splatting including the pull-push phase, T_{indir} is indirect illumination evaluation from VPLs. N_L the number of primary light sources, (*) for Armadillo the VPLs are generated on the sphere to simulate environment map lighting.*

	256 VPLs	512 VPLs	1024 VPLs
VPL Sampling algorithm	T_{sample} $[ms]$	T_{sample} $[ms]$	T_{sample} $[ms]$
Uniform	0.25	0.24	0.25
InvTM	0.62	0.62	0.63
M-H, 1 step	0.26	0.26	0.26
M-H, 5 steps	0.30	0.30	0.31
M-H, 10 steps	0.34	0.34	0.35
M-H, 20 steps	0.44	0.45	0.46

Table 2: *Timings for the generation of different count of VPLs. For InvTM the sampling includes the computation of CDF, for Metropolis-Hastings the different length of chain (i.e. the number of steps) is reported.*

6.4. Discussion and Limitations

Table 3 shows that our dynamic splatting approach can actually be slower if the number of generated splats N_{splats} is lower than the total number of scene triangles N_{tris}. In this case the preprocessing based approach transfers lower amount of data during the splatting phase compared to our approach which has to read and evaluate all the scene triangles. This setup is however a rare case that will unlikely appear in a real applications as the ISM algorithm requires a high number of point samples to create ISMs of sufficient quality. The method does not resolve the case, when the light sources are moving during the animation, for which other techniques such as spatio-temporal filtering are needed as discussed in [SYM*12].

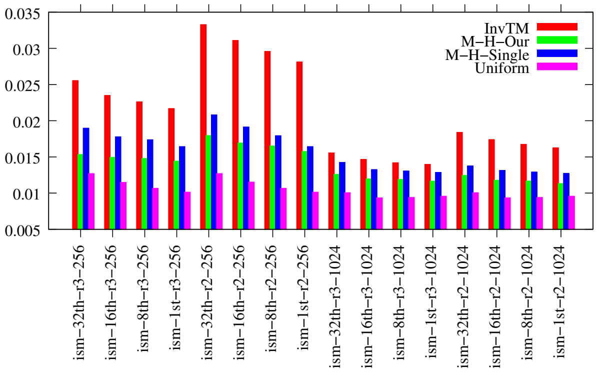
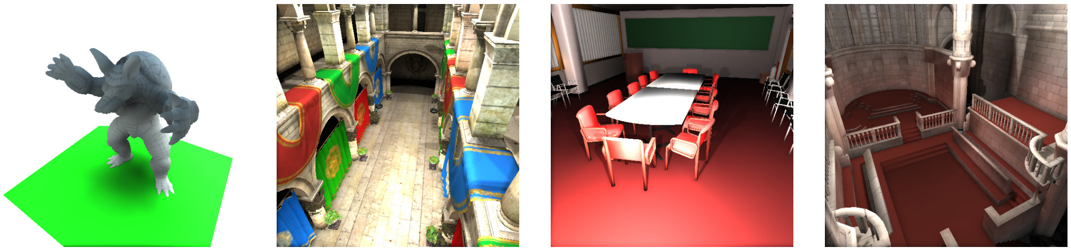


Figure 8: *The average temporal image differences for the walkthrough in Crytek-Sponza. Average sums of temporal image differences are reported for different settings referred to as ism-A-rB-C, where A denotes that every n-th VPL is used for rendering per pixel (1st for all generated VPL hence the highest computation time and quality), B denotes using either 2 or 3 dimensions of random generator and C denotes the number of VPLs generated.*

7. Conclusion

We proposed a novel algorithm for view adaptive computation of VPLs in the context of real-time instant radiosity with imperfect shadow maps. The method is based on paral-



scene	Armadillo	Crytek-Sponza	Conference	Sibenik
N_{tris}	345,946	262,267	331,179	75,284

Figure 9: *The rendered images and the count of triangles in the used test scenes.*

scene	Armadillo			Crytek-Sponza			Conference			Sibenik		
N_{SPLAT}	T_{PR} $[ms]$	T_{DT} $[ms]$	s $[-]$	T_{PR} $[ms]$	T_{DT} $[ms]$	s $[-]$	T_{PR} $[ms]$	T_{DT} $[ms]$	s $[-]$	T_{PR} $[ms]$	T_{DT} $[ms]$	s $[-]$
10,000	0.36	2.42	0.15	0.25	1.21	0.21	0.31	1.40	0.22	0.39	0.56	0.69
100,000	1.66	2.45	0.67	1.34	2.14	0.62	1.90	2.42	0.78	1.81	2.05	0.88
250,000	3.55	2.41	1.48	2.71	2.89	0.94	3.61	3.62	1.00	3.20	2.85	1.12
500,000	5.43	2.37	2.30	4.84	3.49	1.39	5.36	3.95	1.36	5.31	4.55	1.17
1,000,000	8.69	3.84	2.26	8.78	4.63	1.90	8.80	4.84	1.82	9.83	7.38	1.33
2,000,000	17.32	5.35	3.24	17.30	6.52	2.65	16.79	6.95	2.42	18.56	13.35	1.39
4,000,000	34.28	13.25	2.59	34.30	11.23	3.05	33.67	11.42	2.95	34.88	24.06	1.45
8,000,000	68.56	25.87	2.65	68.29	18.74	3.64	66.90	18.82	3.55	68.17	43.94	1.55
16,000,000	136.58	52.12	2.62	136.70	33.03	4.14	133.10	32.55	4.09	136.18	75.71	1.80
32,000,000	272.93	107.25	2.54	272.42	55.20	4.94	266.13	55.54	4.79	274.88	119.40	2.30

Table 3: *Dynamic GPU tessellation versus precomputed point-based sampling. N_{SPLAT} refers to the number of splats used to create the ISM. T_{PR} refers to the time needed to splat the precomputed point cloud [RGK*08, REH*11]. T_{DT} refers to the dynamic tessellation and splatting to ISM. Both T_{PR} and T_{DT} are reported in [ms] excluding the processing of the pull-push phase. Value $s = T_{PR}/T_{DT}$ refers to speedup of the dynamic GPU tessellation.*

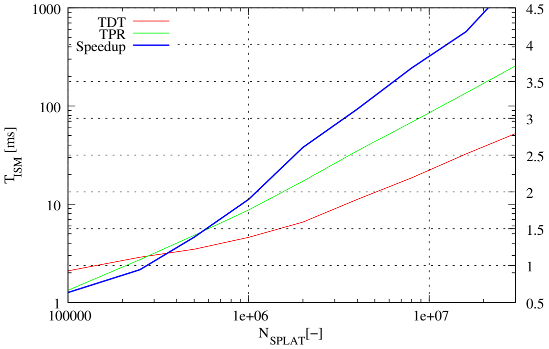


Figure 10: *ISM splatting time T_{ISM} as a function of the splat count N_{SPLAT} measured in the Crytek-Sponza scene. Our approach T_{DT} achieves better performance than the offline method T_{PR} [REH*11] for higher numbers of splats.*

lel Independent Metropolis-Hastings sampling, it is easy to implement, it is fast, and it requires no additional data storage. The results show that the method significantly reduces temporal flickering compared to the sampling based on the inverse transform method.

As a second contribution we proposed to accelerate the ISM creation by using hardware tessellation instead of precomputed point cloud. The results document significant speedup of the proposed technique and its better scalability towards large dynamic scenes.

In the future we would like to study other applications of the proposed temporally coherent Independent Metropolis-Hastings sampling. We also want to test different VPL importance metrics and evaluate them in a perceptual user experiment.

Acknowledgments

Our research was supported by the Czech Science Foundation under research programs P202/11/1883 (Argie) and P202/12/2413 (Opalis), and the Grant Agency of the Czech Technical University in Prague, grant No. SGS13/214/OHK3/3T/13, supported by Ministry of Education of the Czech Republic.

References

[AMHH08] Akenine-Möller T., Haines E., Hoffman N.: *Real-Time Rendering 3rd Edition*. A. K. Peters, Ltd., Natick, MA, USA, 2008. 3

[APSS01] Ashikhmin M., Premoze S., Shirley P., Smits B. E.: A Variance Analysis of the Metropolis Light Transport Algorithm. *Computers & Graphics 25*, 2 (2001), 287–294. 2

[CDO11] Chen S., Dick J., Owen A. B.: Consistency of markov chain quasi-monte carlo on continuous state spaces. *Annals of Statistics 2011 39*, 2 (2011), 673–701. 2

[ČHM*12] Čadík M., Herzog R., Mantiuk R., Myszkowski K., Seidel H.-P.: New Measurements Reveal Weaknesses of Image Quality Metrics in Evaluating Graphics Artifacts. In *ACM Transactions on Graphics (Proc. of SIGGRAPH Asia)* (2012), vol. 31, ACM, pp. 1–10. 7

[DKH*13] Dachsbacher C., Křivánek J., Hašan M., Arbree A., Walter B., Novák J.: Scalable Realistic Rendering with Many-Light Methods. In *Eurographics 2013 state-of-the art report (STAR), to appear* (2013), pp. 1–16. 2

[DKL10] Dammertz H., Keller A., Lensch H. P. A.: Progressive Point-Light-Based Global Illumination. *Computer Graphics Forum 29*, 8 (Dec 2010), 2504–2515. 2

[DS05] Dachsbacher C., Stamminger M.: Reflective Shadow Maps. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games* (New York, NY, USA, 2005), I3D '05, ACM, pp. 203–231. 2, 3

[GS10] Georgiev I., Slusallek P.: Simple and Robust Iterative Importance Sampling of Virtual Point Lights. *Proceedings of Eurographics (short papers)*, 4 pages, (2010). 2

[HAS70] HASTINGS W. K.: Monte Carlo Sampling Methods using Markov Chains and their Applications. *Biometrika 57*, 1 (1970), 97–109. 2, 4

[HREB11] Hollander M., Ritschel T., Eisemann E., Boubekeur T.: Manylods: Parallel Many-View Level-of-Detail Selection for Real-time Global Illumination. *Computer Graphics Forum 30*, 4 (Jun 2011), 1233–1240.

[Kel97] Keller A.: Instant Radiosity. In *SIGGRAPH 97 Conference Proceedings* (Aug. 1997), Whitted T., (Ed.), Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 49–56. 1, 2

[LSK*07] Laine S., Saransaari H., Kontkanen J., Lehtinen J., Aila T.: Incremental Instant Radiosity for Real-Time Indirect Illumination. In *Proceedings of the 18th Eurographics conference on Rendering Techniques* (EGSR'07), 2007, pp. 277–286. 2

[MKC07] Marroquim R., Kraus M., Cavalcanti P. R.: Efficient point-based rendering using image reconstruction. In *Proceedings Symposium on Point-Based Graphics* (2007), pp. 101–108. 3

[MRR*53] Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H., Teller E.: Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics 21*, 6 (1953), 1087–1092. 2

[MW11] Maletz D., Wang R.: Importance Point Projection for GPU-based Final Gathering. *Computer Graphics Forum 30*, 4 (2011), 1327–1336. 2

[RDGK12] Ritschel T., Dachsbacher C., Grosch T., Kautz J.: The State of the Art in Interactive Global Illumination. *Computer Graphics Forum 31*, 1 (Feb 2012), 160–188. 2

[REG*09] Ritschel T., Engelhardt T., Grosch T., Seidel H.-P., Kautz J., Dachsbacher C.: Micro-Rendering for Scalable, Parallel Final Gathering. *ACM Transactions on Graphics 28*, 5 (2009), 132:1-132:8. 2

[REH*11] Ritschel T., Eisemann E., Ha I., Kim J. D. K., Seidel H.-P.: Making Imperfect Shadow Maps View-Adaptive: High-Quality Global Illumination in Large Dynamic Scenes. *Computer Graphics Forum 30*, 8 (2011), 2258–2269. 1, 2, 3, 9

[RGK*08] Ritschel T., Grosch T., Kim M. H., Seidel H.-P., Dachsbacher C., Kautz J.: Imperfect Shadow Maps for Efficient Computation of Indirect Illumination. In *ACM Transactions on Graphics 27*, 5 (2008), 129:1–129:8. 1, 2, 3, 6, 9

[SIMP06] Segovia B., Iehl J. C., Mitanchey R., Péroche B.: Bidirectional Instant Radiosity. In *Proceedings of the 17th Eurographics conference on Rendering Techniques* (2006), EGSR'06, pp. 389–397. 3

[SIP07a] Segovia B., Iehl J.-C., Péroche B.: Coherent Metropolis Light Transport with Multiple-Try Mutations. *Tech. Rep. RR-LIRIS-2007-015*, LIRIS UMR 5205 CNRS/INSA de Lyon, Apr. 2007, pp. 1–12. 2

[SW91] Shirley P., Wang C.: Direct lighting calculation by monte carlo integration. In *Eurographics Workshop on Rendering* (1991), pp. 54–59. 4, 7

[SIP07b] Segovia B., Iehl J.-C., Péroche B.: Metropolis Instant Radiosity. *Computer Graphics Forum 26*, 3 (Sept. 2007), 425–434. 2

[SKDP99] Szirmay-Kalos L., Dornbach P., Purgathofer W.: On the Start-up Bias Problem of Metropolis Sampling. In proceedings *Seventh International Conference in Central Europe on Computer Graphics and Visualization (WSCG'99)* (Feb. 1999), pp. 273–280. 2, 4

[SYM*12] Scherzer D., Yang L., Mattausch O., Nehab D., Sander P. V., Wimmer M., Eisemann E.: Temporal Coherence Methods in Real-Time Rendering. *Computer Graphics Forum 31*, 8 (Dec. 2012), 2378–2408. 2, 8

[Tie94] Tierney L.: Markov Chains for Exploring Posterior Distributions. *Annals of Statistics 22*, 4 (1994), 1701–1762. 4

[TMD*04] Tawara T., Myszkowski K., Dmitriev K., Havran V., Damez C., Seidel H.-P.: Exploiting Temporal Coherence in Global Illumination (an invited paper). In *Spring Conference on Computer Graphics (SCCG 2004)*, 23–33. 2

[VG97] Veach E., Guibas L. J.: Metropolis Light Transport. In *SIGGRAPH 97 Conference Proceedings*, ACM SIGGRAPH, 65–76. 2, 5

[WBS03] Wald I., Benthin C., Slusallek P.: Interactive Global Illumination in Complex and Highly Occluded Environments. In *Proceedings of the 14th Eurographics workshop on Rendering* (2003), EGWR'03, pp. 74–81. 2

[WH00] Wang X., Hickernell F.: Randomized Halton Sequences. *Mathematical and Computer Modelling 32*, 7-8 (2000), 887–899. 4

[Win09] Winkler S.: Video Quality Measurement Standards: Current Status and Trends. In *Proceedings of the 7th international conference on Information, communications and signal processing* (2009), ICICS'09, IEEE Press, pp. 848–852. 7

[WKB*02] Wald I., Kollig T., Benthin C., Keller A., Slusallek P.: Interactive Global Illumination using Fast Ray Tracing. In *Proceedings of the 13th Eurographics workshop on Rendering* (2002), EGWR'02, pp. 15–24. 2