CHAPTER 47

LIGHT SAMPLING IN QUAKE 2 USING SUBSET IMPORTANCE SAMPLING

Tobias Zirr
Karlsruhe Institute of Technology

ABSTRACT

In the context of path tracing, to compute high-quality lighting fast, good stochastic light sampling is just as important as light culling is in the context of raster graphics. In order to enable path tracing in *Quake 2*, multiple solutions were evaluated. Here, we describe the light sampling solution that ended up in the public release. We also discuss its relation to more recent approaches like ReSTIR [4], real-time path guiding [11], and stochastic lightcuts [19]. Finally, we leverage the power of variance reduction techniques known from offline rendering, by providing an extension of our stochastic light sampling technique that allows use of *multiple importance sampling* (MIS). The resulting algorithm can be seen as a variant of stochastic MIS, which was recently proposed in the framework of continuous MIS [34]. To this end, we derive additional theory to introduce pseudo-marginal MIS, allowing for effective variance reduction by marginalization with respect to only parts of the sampling process.

47.1 INTRODUCTION

The Q2VKPT project—leveraging ray tracing to bring unified solutions for the simulation and filtering of all types of light transport into a playable game—was created by Christoph Schied [25] to build an understanding of what is already feasible, and what remains to be done for future ray traced game graphics. The release of GPUs with ray tracing capabilities has opened up new possibilities, yet making good use of ray tracing remains nontrivial: ray tracing alone does not automatically produce realistic images. Light transport algorithms like path tracing can be used for that, realistically simulating the complex ways that light travels and scatters in virtual scenes. However, though elegant and powerful, naive path tracing is also very costly and takes a long time to produce stable images. Even when using smart

© NVIDIA 2021
A. Marrs, P. Shirley, I. Wald (eds.), *Ray Tracing Gems II*, https://doi.org/10.1007/978-1-4842-7185-8_47

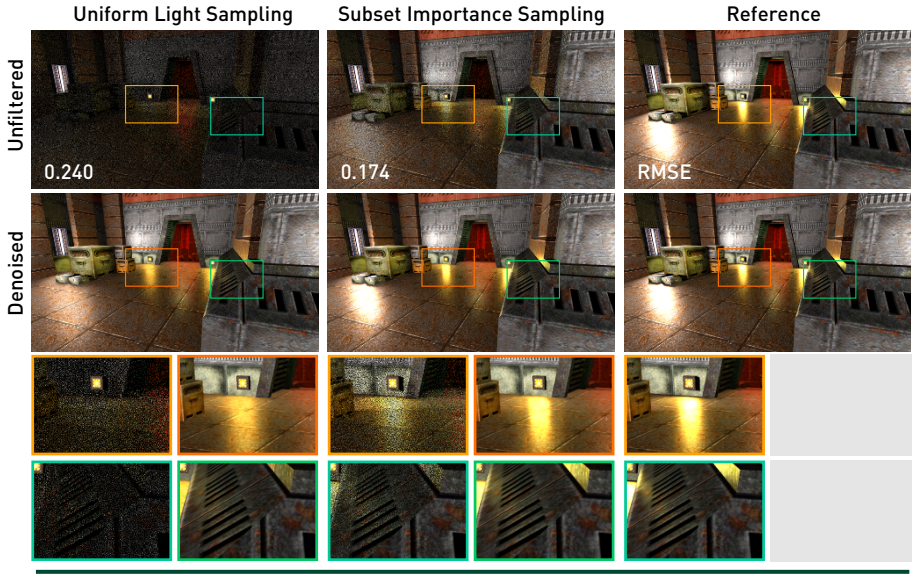

Figure 47-1. *Path tracing in Q2VKPT at one sample per pixel, with and without denoising (using Adaptive Spatiotemporal Variance-Guided Filtering, ASVGF [27]). Sampling all lights in the scene uniformly leads to poor shading and shadow quality, both with and without denoising (left). For the public release of Q2VKPT, we implemented stratified resampled importance sampling (RIS) [30, 5], performing approximate product importance sampling of light and material contributions within stochastic subsets of all lights (middle).*

adaptive filters [26, 7, 27] that reuse as much information as possible across many frames and pixels—similarly to *temporal antialiasing* (TAA)—care has to be taken to keep variance manageable in order to efficiently produce clean and stable images.

Monte Carlo (MC) techniques like path tracing work by tracing random light paths that connect the camera and light sources via scattering surfaces in the scene. Ray tracing is used to resolve visibility: for each scattering interaction, it finds the next visible surface in a given direction. For the resulting shading estimates to be robust (i.e., for them to efficiently converge to the correct result with few path samples), the random paths need to represent the actual illumination in the scene well. However, such good *importance sampling* (IS), ensuring that the path density closely matches the distribution of light transport in the scene, is hard: perfect importance sampling would require predicting the distribution of (indirect) light via arbitrary scattering interactions in advance.

In Q2VKPT, following common practice in (offline) movie rendering [13, 6, 14, 9], path tracing is used with additional light sampling (*next event*

estimation (NEE)) in order to reliably find the relevant lights for each surface in a scene: At each visible surface point, two rays are traced. One is a shadow ray traced toward a random point on a random light source. This point on the emitter is sampled in a way that makes it representative for the entire illumination in the scene (i.e., the probability of sampling specific emitters is proportional to their shading contribution; this is crucial and the concern of this chapter). To recursively accumulate multi-bounce illumination, another ray points in a random direction sampled proportionally to the scattering of the shaded material. The next visible surface in this direction is then shaded in the same way.

Similarly to light culling in real-time rendering, picking the right lights for each surface point is crucial for image quality: in the MC path tracing framework, picking the wrong lights too often results in highly unreliable shading estimates, which then force reconstruction filters to suppress the resulting outliers, removing all the details that the path tracer was supposed to produce in the first place.

47.2 OVERVIEW

In this chapter, we present the constant-time light sampling algorithm implemented in Q2VKPT. The algorithm uses stratified sampling to obtain random subsets with constant size for every frame and pixel, sampled from a potentially long list of relevant light sources. Thus, it quickly hits all light sources over time, while the stratified sampling, with the right locality in light lists, keeps subsets representative of the full illumination in the scene. Within a stochastic subset, importance sampling according to the precisely estimated influence of each light source can be done in a controlled time budget.

In contrast to more recent approaches like ReSTIR [4], our approach is more primitive and thus less optimal, but independent of spatiotemporal data structures (see also Section 47.3)—how to best adapt ReSTIR to multiple bounces in this regard is a question for future research. During development, Q2VKPT was also tested with advanced hierarchical light sampling approaches [10, 20, 19] inspired by movie production: by clustering lights hierarchically, the influence of many lights can be estimated at once, allowing for quick exclusion of far away, dim lights and of lights pointing the wrong way. However, in our experience, such estimates are hard to get precise (see also Section 47.3).

For this chapter, an additional challenge we address is the implementation of *multiple importance sampling* (MIS) with our subset sampling technique. MIS is crucial to limit variance in some common cases, e.g., for specular highlights of nearby reflected emitters and for emitters in close proximity to lit surfaces. Without MIS, these cases require aggressive clamping, potentially hampering the physically based appearance that is expected from a path tracing–based renderer. To integrate MIS with our technique, we build on the framework of stochastic MIS (recently introduced in the context of *continuous MIS*), and we propose pseudo-marginal MIS to allow effective variance reduction using only the readily available information of a stochastic subset of lights, i.e., without requiring expensive full marginalization of probability densities.

47.3 BACKGROUND

Scenes in games and movie productions can contain large amounts of light sources, and as such, sampling of light sources has been a long-standing topic of research both in academia and production rendering. Our goal is to sum the contribution of all light sources $\mathcal{L}_l$ for $l = 0, \ldots, K-1$, emitting light $L_l(\mathbf{x}, \mathbf{y})$ from points $\mathbf{y} \in \mathcal{L}_l$ to *each* shading point $\mathbf{x}$,[1] given its normal $\mathbf{n}$, its material as defined by the *bidirectional scattering distribution function* (BSDF) f_r, and a view direction $\mathbf{o}$:

$$L_o := \sum_{l=0}^{K-1} \int_{\mathcal{L}_l} f_r\left(\mathbf{o}, \mathbf{x}, \mathbf{i}\right)\ \mathbf{n}^T\mathbf{i}\ V(\mathbf{x}, \mathbf{y})\ L_l(\mathbf{x}, \mathbf{y})\, \mathrm{d}\mathbf{y}, \quad \text{where } \mathbf{i} = \frac{\mathbf{y} - \mathbf{x}}{\|\mathbf{y} - \mathbf{x}\|}, \tag{47.1}$$

and the visibility $V(\mathbf{x}, \mathbf{y})$ of $\mathbf{y}$ from $\mathbf{x}$ is determined via ray tracing to obtain accurate shadowing. Computing accurate shading for all shading points and a large number of light sources would be prohibitively expensive. Instead, Monte Carlo integration evaluates a random variable F that computes the shading integrand for only one random point on only one randomly chosen light source for each shading point, in a way such that the expected value of F happens to coincide with the accurate shading by all light sources:

$$F := \frac{f_r\left(\mathbf{o}, \mathbf{x}, \mathbf{i}\right)\ \mathbf{n}^T\mathbf{i}\ V(\mathbf{x}, \mathbf{y})\ L_l(\mathbf{x}, \mathbf{y})}{p(\mathbf{y}|\mathcal{L}_l)P(\mathcal{L}_l)}, \quad \mathbf{E}_{p(\mathbf{y}|\mathcal{L}_l)P(\mathcal{L}_l)}[F] = L_o. \tag{47.2}$$

As long as the chance of sampling is nonzero for all the points on the light sources that contribute nonzero shading to the point $\mathbf{x}$, we are free to choose

[1] For an area light, $L_l(\mathbf{x}, \mathbf{y}) = (|\mathbf{n}_l^T\mathbf{i}|\ L_l(\mathbf{y}, -\mathbf{i}))/\|\mathbf{y} - \mathbf{x}\|^2$, where $\mathbf{n}_l$ is the normal of the emitting surface and $L_l(\mathbf{y}, -\mathbf{i})$ is its radiance at $\mathbf{y}$ toward $\mathbf{x}$. For a point light, $L_l(\mathbf{x}, \mathbf{y}) = (\delta(\mathbf{y} - \mathbf{y}_l)\ I_l(-\mathbf{i}))/\|\mathbf{y} - \mathbf{x}\|^2$, where $\mathbf{y}_l$ is its position and $I_l(-\mathbf{i})$ is its intensity toward $\mathbf{x}$. For a directional light, $L_l(\mathbf{x}, \mathbf{y}) = \delta(\mathbf{y} - \mathbf{y}_l(\mathbf{x}))\ E_l(\mathbf{x})$, where $E_l(\mathbf{x})$ is its irradiance toward $\mathbf{x}$ and $\mathbf{y}_l(\mathbf{x})$ projects $\mathbf{x}$ onto its orthogonal plane.

any sampling strategy: In F, divide by the probability $P(\mathcal{L}_l)$ of randomly choosing the light source $\mathcal{L}_l$, and divide by the probability density $p(\mathbf{y}|\mathcal{L}_l)$ of randomly choosing the point $\mathbf{y}$ on $\mathcal{L}_l$, then compensate for how often each of the samples occurs. Both divisions are canceled out in the expected value.

In order to obtain good shading estimates fast, we want to limit the potential deviation of F from the accurate shading L_o (typically quantified by the variance $\mathbf{V}[F]$). This is achieved when the light source $\mathcal{L}_l$ and $\mathbf{y} \in \mathcal{L}_l$ are sampled often where their contribution to the shading point $\mathbf{x}$ is high: ideally, we want the probability density $p(\mathbf{y}|\mathcal{L}_l)P(\mathcal{L}_l)$ of light sampling to cancel out the shading integrand [29], such that F always equals the correct result for *any* random samples $\mathcal{L}_l, \mathbf{y} \in \mathcal{L}_l$.

47.3.1 LIGHT RESAMPLING

To achieve such *importance sampling* in real-time path tracing, Bikker [2] uses *resampled importance sampling* (RIS) [30, 5], where first a set of candidate points on light sources is sampled, then their contribution is estimated to sample one final point, using a distribution function proportional to their contribution relative to the other candidate points. For direct illumination, Bitterli et al. [4] propose the ReSTIR algorithm, which optimizes and refines this approach in many ways: Based on the framework of reservoir sampling [8], they avoid storing any candidate points, resampling candidates such that at any time only one tentative final point, surviving the sampling process, needs to be stored. Furthermore, they observe that resampling results can be reused across pixels and even multiple frames, applying resampling hierarchically. By this parallel distribution of sampling efforts over space and time, ReSTIR quickly achieves importance sampling with respect to very large numbers of light sources and candidate emitter points, leading to high-quality, low-variance shading results with little computational overhead.

In the public release of Q2VKPT, our stochastic light subset sampling without MIS bears similarities to Bikker's approach, but adds a stratified sampling scheme that leads to better screen-space error distribution, uses fewer random variables, and avoids creating many candidate points on light sources (see Section 47.4.3). Internally, we also tested reservoir sampling to avoid explicit storage of candidate contributions (see Section 47.4.4), but because it reduced the benefits of using blue noise pseudo-random numbers, we decided against it in the public release. As we show in this chapter, our approach can be extended to allow MIS [31], which efficiently reduces

variance and thus potential energy loss due to clamping, especially for reflective materials. In comparison to ReSTIR, our algorithm is more primitive and less optimal, but in contrast, our approach also handles indirect illumination easily, requiring no changes.

47.3.2 HIERARCHICAL LIGHT SAMPLING

Shirley et al. [29] describe sampling strategies for different light source shapes. To reduce the amount of light sources they need to consider, they precompute importance stored in an octree. It has since become a common approach in offline rendering to organize light sources into a tree data structure, which is then typically stochastically traversed [13, 18] for efficient importance sampling among many lights. Depending on performance trade-offs, lights are clustered hierarchically by either higher-quality *bounding volume hierarchies* (BVHs), mixed-quality two-level BVHs [21], or faster-to-construct *linear-time BVHs* (LBVHs) [19].

A difficulty of such hierarchical importance sampling schemes is estimating the contribution of many lights from purely aggregate information in coarser levels during traversal: when a shading point is close to a cluster of lights, or even contained therein, the contribution of each contained light may depend strongly on the location and orientation of the shading point. To this end, Estevez and Kulla [10] complement contribution estimation heuristics with reliability heuristics during traversal, collecting lights from all subtrees whenever an aggregate contribution estimate is deemed unreliable. Similarly, stochastic light cuts [19] combine the ideas of light cuts [33] (representing the emissions of many lights approximately by fewer aggregate lights, clustering hierarchically and refining adaptively during rendering) and of light sampling hierarchies (replacing any selected aggregate lights in the light cut by hierarchically sampling leaf lights to recover unbiasedness).

Variations of these hierarchical algorithms vary in their aggregate reliability heuristics, the number of lights selected, and the data structures used. We refer to Moreau and Clarberg [20] and Estevez and Kulla [10] for a broader overview. As common in real-time rendering, to further improve sampling and performance at the cost of introducing bias, the number of light sources considered at each point can be reduced by restricting the emission range [28, 3, 14].

In Q2VKPT, we experimented with light hierarchies following Moreau and Clarberg [20], adapting the heuristics of Estevez and Kulla [10] to real-time

rendering. However, lacking the adaptive splitting heuristic [10] (which would require costly traversal of all subtrees when encountering unreliable contribution estimates), the algorithm was hard to get robust. We often saw aggregate contribution heuristics failing and leading to visible noise artifacts in the resulting image. With the right tuning, a two-phase parallel approach of first splitting then stochastic traversal as in stochastic light cuts [19] might work for certain GPU applications. However, in either form, stochastic tree traversals cause incoherent memory access patterns. Finally, in animations, tree updates can cause drastic changes of split quality metrics and thus tree topology, making the sampling quality temporally inconsistent. Our simpler light subset selection algorithm avoids these caveats.

47.3.3 SAMPLE REUSE AND GUIDING

Similarly to how ReSTIR [4] distributes importance sampling over multiple pixels and frames, caching and reusing information for improved importance sampling has been done in more general settings, e.g., as importance caching [15] of light importance distributions on a sparse set of surface points, distributed in the scene before rendering. Caching has found its way into production for both direct [6] and indirect [32, 22] illumination. Dittebrandt et al. [11] come up with data structures and compression schemes to support online learning of light sampling distributions in real time, which also works for indirect bounces. Their paper shows results for *Quake 2* RTX. Typical challenges of guiding, learning, caching, and reusing algorithms are the detection of similarity, applicability, and expiration of cached or shared information, which we avoid by the simple approaches in this chapter.

47.4 STOCHASTIC LIGHT SUBSET SAMPLING

Because tracing rays for shadow tests is still a costly operation, we follow common path tracing practice and select only one light source per shading point. It may seem like we can skip shading computations for many light sources altogether, however this would not lead to high-quality shading but to a lot of noise: We still need to approximate shading contributions of all lights that we choose from. Only then, we can choose each light with the right probability proportional to its contribution [29], ensuring MC sample values that are close to the correct result.

The challenge we address in this section is to reduce the number of light sources for which we approximate shading contributions, while keeping the

probability of sampling each light approximately proportional to its contribution relative to all light sources. In Q2VKPT, for each shading point we have a (possibly long) *light list* of emitters that are potentially relevant for shading that point. We build these lists from the *potentially visible set* (PVS) for each cluster of level geometry (binary space partitioning (BSP) node) at load time. Nearby dynamic lights are appended at runtime. Building lists of relevant lights for level geometry is a known problem to game developers with various solutions (parallel grid/cluster culling based on proximity heuristics, potentially visible sets, etc.).

To achieve our goal of cutting down on the long lists of relevant light indices J at a shading point **x**, let us split the contribution of all K relevant light sources into S parts, iterating through K/S light indices J_s in each part:

$$L_o = \sum_{s=0}^{S-1} \sum_{l \in J_s} \int_{\mathcal{L}_l} f_r\left(\mathbf{o}, \mathbf{x}, \mathbf{i}\right) \; \mathbf{n}^T\mathbf{i} \; V(\mathbf{x}, \mathbf{y}) \; L_l(\mathbf{x}, \mathbf{y}) \, \mathrm{d}\mathbf{y}. \tag{47.3}$$

This allows us to construct a corresponding Monte Carlo estimator F_s that computes L_o by first randomly selecting a subset of lights J_s and then sampling a light index l from the subset only:

$$F_S := \frac{f_r\left(\mathbf{o}, \mathbf{x}, \mathbf{i}\right) \; \mathbf{n}^T\mathbf{i} \; V(\mathbf{x}, \mathbf{y}) \; L_l(\mathbf{x}, \mathbf{y})}{p(\mathbf{y}|\mathcal{L}_l)P(l|J_s)P(s)}, \quad \mathbf{E}_{p(\mathbf{y}|\mathcal{L}_l)P(l|J_s)P(s)}[F_S] = L_o. \tag{47.4}$$

As we want to avoid computations for more than one subset J_s, we choose one s uniformly at random ($P(s) = 1/S$). In order to still obtain high-quality, low-variance results, each subset J_s should be representative of all relevant light indices J; that is, we would like to achieve approximately the same probabilities sampling from J_s as sampling from the full distribution:

$$P(l|J_s)P(s) \approx P(l|J), \quad \text{where } J = \bigcup_{s=0}^{S-1} J_s \text{ and } J_s \cap J_{s'} = \emptyset. \tag{47.5}$$

Within the shorter lists J_s, it becomes feasible to perform product importance sampling for sampling $l \in J_s$, i.e., to sample each contained light proportional to its predicted shading contribution (neglecting visibility). If the subset J_s is indeed representative of J, the procedure will not produce much more noise than sampling from the full light list J.

47.4.1 PRACTICAL STRIDED SUBSETS

In Q2VKPT, we use constant-size subsets of (maximum) length R, splitting the list J of K relevant light indices for a shading point into $S = \lceil K/R \rceil$ subsets J_s. If

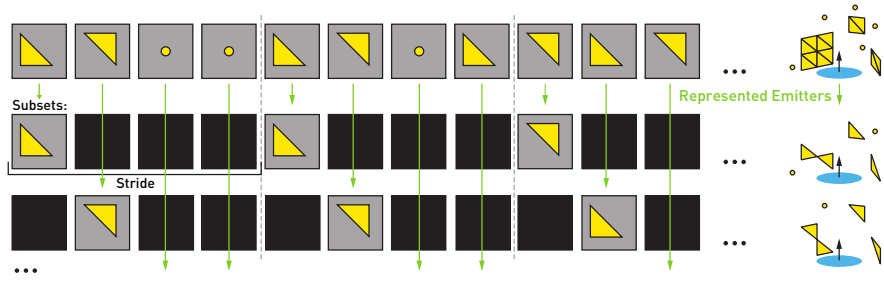

Figure 47-2. *Our light sampling performs approximate product importance sampling by importance sampling from strided "representative" subsets of the full light list for each shading point.*

the full list J of lights is small, there will be only one subset and the light sampling is optimal. For subsets of long lists J to stay representative, we choose a *strided* subset selection strategy as illustrated in Figure 47-2; we pick every Sth light from J with varying initial offsets s:

$$J_s := J\left[s + iS | i \in \mathbb{N}_0\right]. \tag{47.6}$$

The reasoning behind this strategy is based on the way our light lists are constructed in Q2VKPT: Due to our light geometry being sourced from the game level's hierarchical space partitions, spatially colocated emitters are often colocated in memory. Thus, a strided subset of light sources often still gives a good idea of their spatial distribution in the scene.

An important advantage of choosing the subset by one offset s is that we can benefit from low-discrepancy random variables: We use the same random variable to first select one subset out of S subsets, then rescale the interval that maps to offset s such that the random values contained by it can be reused to sample the light within the selected subset. If we map exactly one random variable interval to every light source (proportional to its contribution), then the resulting noise will show the pleasant dither pattern of low-discrepancy points (they are optimized to sample intervals well).

47.4.2 WORST-CASE VARIANCE ANALYSIS

The values of F_S can be bounded relative to the values of an idealized estimator F^*, which would perform importance sampling with respect to all relevant lights, sampling with the probability distribution function (PDF) $p(Y|\mathcal{L}_l)P(l|J)$:

$$F^* = \frac{f_r\left(\mathbf{o}, \mathbf{x}, \mathbf{i}\right)\ \mathbf{n}^T\mathbf{i}\ V(\mathbf{x}, \mathbf{y})\ L_l(\mathbf{x}, \mathbf{y})}{p(\mathbf{y}|\mathcal{L}_l)P(l|J)}, \tag{47.7}$$

$$F_S = F^*\frac{P(l|J)}{P(l|J_s)P(s)} = F^* S\frac{\sum_{i\in J_s} C(\mathcal{L}_i)}{\sum_{i\in J} C(\mathcal{L}_i)} \leq \lceil K/R\rceil F^*, \tag{47.8}$$

for estimated contributions $C(\mathcal{L}_l)$ of lights $\mathcal{L}_l$ to the shading point, such that $P(l|J) = C(\mathcal{L}_l)/\sum_{i\in J} C(\mathcal{L}_i)$. It follows from $P(s) = 1/S$ in our sampling procedure that, in the worst case, our subset sampling causes at most $S = \lceil K/R\rceil$ times higher error than ideal importance sampling, while guaranteeing constant runtime. In Q2VKPT, we choose $R = 8$ representatives and thus our subset count S is small. In practice, the variance is even lower: when each subset is representative of the full illumination, it is close to optimal (then $F_S \approxeq F^*$).

47.4.3 TWO-SWEEP ALGORITHM

Listing 47-1 provides pseudocode for the direct implementation of our light sampling algorithm, illustrated in Figure 47-2. Our light lists are concatenated in one long array `light_pointers`. Given a shading point **x**, the enclosing potentially visible set cluster provides an index range [`light_list_begin`, `light_list_end`) for the relevant lights. The contribution of contained light sources is predicted by the function `light_contrib`, as detailed in Section 47.4.5.

First, we split the potentially long list of K relevant lights into S representative parts of length R = `MAX_SUBSET_LENGTH`. We do this using the number of parts as a stride, such that all `subset_stride`-th lights in the list become part of the same subset. We then randomly select one of these subsets by sampling a start offset `subset_offset` relative to the beginning of the light list.

In a first loop over all lights in the selected (strided) subset, we compute a conservative estimate for the contribution of each subset light, which we store in a fixed-length importance sampling array `is_weights` and sum up to obtain a normalization constant for random sampling. Afterward, we determine a random threshold for sampling one of the subset lights with a probability proportional to its contribution weight. We do this in a second loop, reiterating the stored contribution weights until the prefix weight mass reaches the random threshold.

Finally, we compute the probability of sampling the chosen light. This probability is a coefficient that can afterward be multiplied to the PDF of sampling one point on the chosen emitter, to obtain the final light sampling PDF.

Listing 47-1. *Sampling a light source from a strided stochastic subset in two sweeps, using one random variable ξ_1.*

```
S = ⌈(light_list_end-light_list_begin) / MAX_SUBSET_LEN⌉
subset_stride = S
subset_offset = ⌊ξ1 * S⌋
ξ1 = ξ1 * S - ⌊ξ1 * S⌋

total_weights = 0
float is_weights[MAX_SUBSET_LEN]

light_idx = light_list_begin + subset_offset
for (i = 0; i < MAX_SUBSET_LEN; ++i) {
    if (light_idx >= light_list_end) {
        break
    }
    w = light_contrib(v, p, n, light_pointers[light_idx])
    is_weights[i] = w
    total_weights += w

    light_idx += subset_stride
}

ξ1 *= total_weights
mass = 0

light_idx = light_list_begin + subset_offset
for (i = 0; i < MAX_SUBSET_LEN; ++i) {
    if (light_idx >= light_list_end) {
        break
    }
    mass = is_weights[i]

    ξ1 -= mass
    if not (ξ1 > 0) {
        break
    }
    light_idx += subset_stride
}

probability = mass / (total_weights * S)
return (light_pointers[light_idx], probability)
```

47.4.4 ONE-SWEEP ALGORITHM

The requirement of intermediately storing contribution weights can increase register and/or memory pressure, depending on the compiler and architecture. We can avoid this storage and iterating over the light subset twice (at the cost of potentially less stratified sampling, e.g., if blue noise is used for random variables).

Listing 47-2. *Sampling a light source from a strided stochastic subset in one sweep, using one random variable ξ_1.*

```
1  S = ⌈(light_list_end-light_list_begin) / MAX_SUBSET_LEN⌉
2  subset_stride = S
3  subset_offset = ⌊ξ1 * S⌋
4  ξ1 = ξ1 * S - ⌊ξ1 * S⌋
5
6  selected = (light_idx: -1, mass: 0)
7  total_weights = 0
8
9  light_idx = light_list_begin + subset_offset
10 for (i = 0; i < MAX_SUBSET_LEN; ++i) {
11     if (light_idx >= light_list_end) {
12         break
13     }
14     w = light_contrib(v, p, n, light_pointers[light_idx])
15     if (w > 0) {
16         τ = total_weights / (total_weights+w)
17         total_weights += w
18
19         if (ξ1 < τ) {
20             ξ1 /= τ
21         } else {
22             selected = (light_pointers[light_idx], w)
23             ξ1 = (ξ1-τ) / (1-τ)
24         }
25         ξ1 = clamp(ξ1, 0, MAX_BELOW_ONE) // Avoid numerical problems.
26     }
27
28     light_idx += subset_stride
29 }
30
31 probability = selected.mass / (total_weights * S)
32 return (selected.light, probability)
```

For this, we use an incremental sampling scheme (reservoir sampling [8]) that retains the same probabilities of sampling each light, but redistributes the value range of the random variable ξ_1 differently: for each iterated light, a decision is made whether to keep the previously selected light or to switch to the current light as the next candidate for the final sampled emitter.

Listing 47-2 provides the altered pseudocode for the one-sweep, incremental sampling implementation of our light sampling algorithm. The threshold τ computed in line 16 ensures that the final probability of keeping the light source stored in `selected` as the final sampled emitter is still exactly proportional to its contribution weight: given the probability $P_s(L_i) = 1 - \tau$ of selecting emitter L_i in step i, we find the final probability $P(L_i)$ of selecting and

keeping L_i to be

$$P_s(L_i) = 1 - \tau = \frac{w_i}{\sum_{j=0}^{i} w_j} \tag{47.9}$$

$$\Rightarrow P(L_i) = P_s(L_i) \prod_{j=i+1} \left(1 - P_s(L_j)\right) = \frac{w_i}{\sum_{j=0}^{\#J_s-1} w_j}. \tag{47.10}$$

Besides this, Listing 47-2 closely follows what we saw in Listing 47-1.

47.4.5 PREDICTING THE CONTRIBUTION OF LIGHT SOURCES

Making a good prediction for the contribution of a single light source $\mathcal{L}_l$ to a given shading point $\mathbf{x}$ (i.e., evaluating Equation 47.1 with only one fixed l) can be a challenge in itself, depending on the involved materials, the visibility, and the shape of $\mathcal{L}_l$. As it is unpredictable, we neglect visibility (note that more recent approaches like ReSTIR [4] and real-time path guiding [11] improve on this). For predicting the unshadowed contribution, we use a rather simplistic heuristic: First, we evaluate the BRDF of the shading point for the center of the light source, clipping the resulting light direction to the horizon of the shaded surface. In order to prevent misguiding by the peaks of highly reflective materials, we limit the material roughness in this context. All our lights are triangles. To account for their extent, we compute the solid angle covered by the light source, projecting its triangle onto the sphere around $\mathbf{x}$. A precise formula is given by Arvo [1]. To conservatively bound the cosine, we compute the dot product of the normal $\mathbf{n}$ at $\mathbf{x}$ and the direction toward the highest light vertex above the shading horizon.

47.4.6 PRACTICAL IMPROVEMENTS

For more precise predictions of shading contributions with arbitrary BRDFs, there is a body of recent research to draw on, such as accurate analytic estimation of the unshadowed shading via *linearly transformed cosines* (LTCs) for area lights [16] and linear lights [17]. Optimal light sampling strategies on the level of individual light sources are a topic of ongoing research (and with it, analytic integration to obtain corresponding PDFs), with notable recent advancements for area lights [23] and sphere lights [24].

47.5 REDUCING VARIANCE WITH PSEUDO-MARGINAL MIS

A typical issue of light sampling algorithms is that, with highly reflective materials, predicting the shading contribution of lights is difficult to impossible (depending on visibility): the shading contribution

$f(\mathbf{y}) = f_r\left(\mathbf{o}, \mathbf{x}, \mathbf{i}\right)\ \mathbf{n}^T\mathbf{i}\ V(\mathbf{x}, \mathbf{y})\ L_l(\mathbf{x}, \mathbf{y})$ (see Equation 47.1) for such BSDFs f_r may be very different even for individual points $\mathbf{y}$ on the same light $\mathcal{L}_l$, making its shading $\int_{\mathcal{L}_l} f(\mathbf{y})\,\mathrm{d}\mathbf{y}$ dependent on the precise shape of the light source and its visibility. In production rendering, such shading is often left to other sampling techniques than light sampling, such as finding lights by simple ray tracing instead: rays into relevant directions are sampled proportionally to the scattering profile of the material. In this section, we adapt this strategy into the context of real-time light sampling in Q2VKPT.

47.5.1 MULTIPLE IMPORTANCE SAMPLING

The decision where which sampling technique works best and should therefore be trusted most can be made by the heuristics of multiple importance sampling [31]. The key observation of MIS is that in MC estimation, for each sampled point $\mathbf{y}$ that contributes light $f(\mathbf{y})$, we can decide individually what fraction $w_1(\mathbf{y})$ of its contribution should be estimated by one technique such as light sampling and what fraction $w_2(\mathbf{y})$ by another technique like tracing with BSDF scattering rays. The two sampling techniques will generate the same points $\mathbf{y}$ with different probability distributions, e.g., $p_1(\mathbf{y})$ for light sampling and $p_2(\mathbf{y})$ for BSDF sampling, while the weighted sum of their samples will still converge to the shading we want to compute:

$$\mathbf{E}_{\substack{p_2(\mathbf{y}_2)\\ \cdot p_1(\mathbf{y}_1)}}\left[\sum_{i=1}^{2} w_i(\mathbf{y}_i)\frac{f(\mathbf{y}_i)}{p_i(\mathbf{y}_i)}\right] = \int_{\mathcal{L}}\int_{\mathcal{L}} \sum_{i=1}^{2} w_i(\mathbf{y}_i) f(\mathbf{y}_i) \prod_{j\neq i}^{2} \underbrace{p_j(\mathbf{y}_j)\,\mathrm{d}\mathbf{y}_j}_{\int_{\mathcal{L}} p_j(\mathbf{y}_j|\mathrm{U})\,\mathrm{d}\mathbf{y}_j=1} \mathrm{d}\mathbf{y}_i \tag{47.11}$$

$$= \sum_{i=1}^{2} \int_{\mathcal{L}} w_i(\mathbf{y}_i) f(\mathbf{y}_i)\,\mathrm{d}\mathbf{y}_i = \int_{\mathcal{L}} \underbrace{\left(\sum_{i=1}^{2} w_i(\mathbf{y})\right)}_{=1} f(\mathbf{y})\,\mathrm{d}\mathbf{y}. \tag{47.12}$$

A simple weighting heuristic that retains the good parts of light sampling with $p_1(\mathbf{y})$ and BSDF sampling with $p_2(\mathbf{y})$ is the balance heuristic [31] with weights $w_i(\mathbf{y}) = p_i(\mathbf{y}) / \sum_j p_j(\mathbf{y})$, conveniently resulting in the same MC estimates regardless of the sampling strategy used to generate $\mathbf{y}$:

$$w_i(\mathbf{y})\frac{f(\mathbf{y})}{p_i(\mathbf{y})} = \frac{f(\mathbf{y})}{\sum_j p_j(\mathbf{y})}. \tag{47.13}$$

Note that the resulting denominator in fact corresponds to the marginal probability distribution of using both sampling techniques in parallel.

47.5.2 STOCHASTIC MULTIPLE IMPORTANCE SAMPLING

In the following, we adapt MIS to add BSDF sampling along with the light sampling in Q2VKPT. This significantly reduces noise and denoiser artifacts for glossy materials in some hard cases (see Figure 47-5 in Section 47.7). However, an interesting challenge arises when we want to apply MIS to our stochastic subset sampling strategy: in order to compute the denominator $p_1(\mathbf{y}) + p_2(\mathbf{y})$ for our MIS-weighted shading estimates, we would need to know the marginal probability density of sampling from all subsets J_s (see Section 47.4) rather than from one subset only. For our previous strided subset selection scheme, if we found a light by ray tracing with PDF $p_2(\mathbf{y})$, for computing $p_1(\mathbf{y})$ we would at least have to look at all the lights in the subset of that light source, also.

To avoid this overhead, we construct a generalized stochastic variant of MIS that allows us to circumvent marginalizing with respect to all possible ways of sampling $\mathbf{y}$. Such generalizations were recently discussed as approximations of continuous MIS [34] and as generalized (discrete) MIS [12]. As a generalized form of generating stochastic subsets of all relevant lights J, let I_U denote subsets that are constructed using any number of uniform random variables $U = (u_1, u_2, ..., u_R) \in \mathcal{U}, \mathcal{U} = [0, 1]^R$. The balance heuristic can be applied to only the sampling technique resulting from the stochastic subset:

$$\mathbf{E}_{p_2(\mathbf{y}_2|U)\,p_1(\mathbf{y}_1|U)\,p(U)}\left[\sum_{i=1}^{2}\frac{f(\mathbf{y}_i)}{p_1(\mathbf{y}_i|U)+p_2(\mathbf{y}_i)}\right]$$

$$= \int_{\mathcal{U}}\int_{\mathcal{L}}\int_{\mathcal{L}}\sum_{i=1}^{2}\frac{f(\mathbf{y}_i)}{p_1(\mathbf{y}_i|U)+p_2(\mathbf{y}_i)}p_i(\mathbf{y}_i|U)\underbrace{p(U)}_{=1}\prod_{j\neq i}^{2}\underbrace{p_j(\mathbf{y}_j|U)\,\mathrm{d}\mathbf{y}_j}_{\int_{\mathcal{L}}p_j(\mathbf{y}_j|U)\,\mathrm{d}\mathbf{y}_j=1}\mathrm{d}\mathbf{y}_i\,\mathrm{d}U \tag{47.14}$$

$$= \int_{\mathcal{U}}\sum_{i=1}^{2}\int_{\mathcal{L}}\frac{f(\mathbf{y}_i)}{p_1(\mathbf{y}_i|U)+p_2(\mathbf{y}_i)}\underbrace{p_i(\mathbf{y}_i|U)}_{\substack{\text{note:}\\ p_2(\mathbf{y}|U)=p_2(\mathbf{y})}}\mathrm{d}\mathbf{y}_i\,\mathrm{d}U = \int_{\mathcal{L}}\left(\underbrace{\int_{\mathcal{U}}\mathrm{d}U}_{=1}\right)f(\mathbf{y})\,\mathrm{d}\mathbf{y}. \tag{47.15}$$

However, it is important to note that such naive randomization of MIS may render its weighting ineffective for variance reduction: whenever BSDF sampling hits a light source that is not contained by the current stochastic subset I_U, the PDF $p_1(\mathbf{y}|U)$ becomes zero and MIS is effectively inactive.

47.5.3 PSEUDO-MARGINAL MIS

For good variance reduction performance, we need to ensure that any stochastic subset we select also helps with variance reduction for any lights

hit by BSDF samples. Our key insight is that we do not have to choose between full marginalization or full randomization of MIS weights. In particular, it is also possible to marginalize with respect to one random variable only. For example, if in the subset I_U each selected light source was chosen with one random variable u_j, then we can marginalize with respect to u_1, while reusing all the information of light sources chosen with the other random variables $u_{j\neq 1}$:

$$\mathbf{E}_{p_2(\mathbf{y}_2|\mathsf{U})\,p_1(\mathbf{y}_1|\mathsf{U})\,p(\mathsf{U})}\left[\sum_{i=1}^{2}\frac{f(\mathbf{y}_i)}{\int_{[0,1]}p_1(\mathbf{y}_i|\mathsf{U})\,\mathrm{d}u_1+p_2(\mathbf{y}_i)}\right]$$

$$=\int_{\mathcal{U}}\sum_{i=1}^{2}\int_{\mathcal{L}}\underbrace{\frac{f(\mathbf{y}_i)}{\int_{[0,1]}p_1(\mathbf{y}_i|\mathsf{U})\,\mathrm{d}u_1+p_2(\mathbf{y}_i)}}_{\text{note: does not depend on } u_1}p_i(\mathbf{y}_i|\mathsf{U})\,\mathrm{d}\mathbf{y}_i\,\mathrm{d}\mathsf{U} \tag{47.16}$$

(see Equations 47.14 and 47.15)

$$=\int_{[0,1]^{R-1}}\sum_{i=1}^{2}\int_{\mathcal{L}}\frac{f(\mathbf{y}_i)}{\int_{[0,1]}p_1(\mathbf{y}_i|\mathsf{U})\,\mathrm{d}u_1+p_2(\mathbf{y}_i)}\underbrace{\left(\int_{[0,1]}p_i(\mathbf{y}_i|\mathsf{U})\,\mathrm{d}u_1\right)}_{\substack{\text{note:}\\ \int_{[0,1]}p_2(\mathbf{y}|\mathsf{U})\,\mathrm{d}u_1=p_2(\mathbf{y})}}\mathrm{d}\mathbf{y}_i\,\mathrm{d}(u_2,\ldots) \tag{47.17}$$

$$=\int_{\mathcal{L}}\underbrace{\left(\int_{[0,1]^{R-1}}\mathrm{d}(u_2,\ldots)\right)}_{=1}f(\mathbf{y})\,\mathrm{d}\mathbf{y}. \tag{47.18}$$

The marginalization of $p_1(\mathbf{y}|\mathsf{U})$ with respect to u_1 is simple, as the resulting PDF only depends on whether or not u_1 adds l to I_U for the respective $\mathcal{L}_l \ni \mathbf{y}$. The probability of the respective events is multiplied by the respective values of $p_1(\mathbf{y}|\mathsf{U})$. With this, we can ensure that the marginal density $\int_{[0,1]}p_1(\mathbf{y}|\mathsf{U})\,\mathrm{d}u_1$ is nonzero for any $\mathbf{y}\in\mathcal{L}_l$ found by BSDF sampling, as long as we ensure that any l can be selected with u_1.

47.5.4 STRATIFIED PSEUDO-MARGINAL MIS

Sampling light sources in the stochastic subsets I_U completely independently has disadvantages, because the same light source may unnecessarily be chosen twice. We can simplify computing the marginalization of $p(\mathbf{y}|\mathsf{U})$ by stratification, and even increase the resulting probability density reliably: We define a stratified stochastic subset J_U, based on a list of random variables $\mathsf{U}=(u_1,\ldots,u_R)$, each random variable u_i independent and uniform, as

$$\mathsf{J}_\mathsf{U} := \mathsf{J}\left[iS+\lfloor u_i S\rfloor \,|\, i\in\mathbb{N}_0\right], \tag{47.19}$$

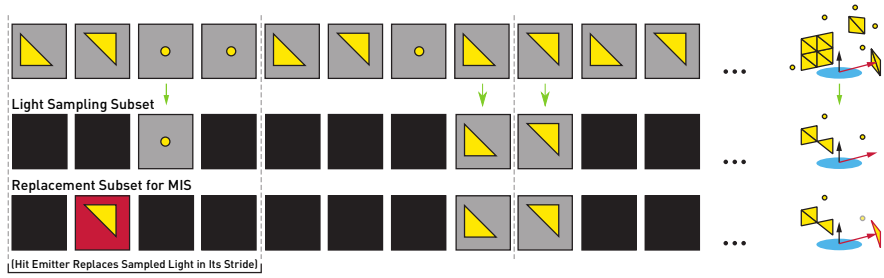

Figure 47-3. *To enable MIS, stochastic light subsets are randomized independently within each stride. This allows us to compute additional sampling weights for emitters hit by BSDF rays: We can replace any light selected in the corresponding subset stride by the hit emitter, and thus obtain another representative stochastic subset for that emitter with minimal additional information.*

that is, each random variable u_i selects an index from the list of all relevant lights J within a stride of S light indices. This is illustrated in Figure 47-3. For a given $\mathbf{y}$, we marginalize with respect to the random variable u_j that is responsible for selecting $\mathcal{L}_l$, where $y \in \mathcal{L}_k$. The only case in which $p_1(\mathbf{y}|\mathrm{U})$ is nonzero is when $\mathrm{J_U}$ contains within the respective stride $[jS, (j+1)S)$ in the list J:

$$\int_{[0,1]} p_1(\mathbf{y}_i|\mathrm{U})\,\mathrm{d}u_j = \frac{1}{S}\,p_1(\mathbf{y}_i|\mathrm{U}\text{ setting }u_j\text{ such that }l \in \mathrm{J_U}). \tag{47.20}$$

Note that with this we effectively perform different marginalizations for different $\mathbf{y} \in \mathcal{L}$, but the derivation in Equation 47.18 still works after the integral over $\mathcal{L}$ is split into respective regions, each cancelling out a different marginalized PDF.

47.6 STOCHASTIC LIGHT SUBSET MIS

To implement the idea of pseudo-marginal MIS, we need to change our way of stochastically sampling subsets, such that for each subset selected in light sampling, it becomes easy to construct a similar, equally probable subset that is guaranteed to contain a specific emitter hit by ray tracing. This can be achieved by independently randomizing the subset selection process in each stride of the light list (see Equation 47.19), such that for every subset, there exists another equally probable subset that replaces one light by the hit emitter, within the respective light list stride (see Figure 47-3). In Listing 47-3, we identify this stride during the light sampling process and additionally keep track of the total weight of all lights in the subset except for the one in the stride of the hit emitter.

Listing 47-3. *Light sampling with information for MIS weighting of BSDF-ray light hits, using one random variable ξ_1.*

```
S = ⌈(light_list_end-light_list_begin) / MAX_SUBSET_LEN⌉
subset_stride = S

selected = (light_idx: -1, mass: 0)
total_weights = 0

// Additional information and randomization for MIS
hit_caught = not hit_emitter
other_weights = 0 // Weights excluding hit_emitter stride
pending_weight = 0
FastRng small_rnd(seed: ξ1, mod: S)

light_offset = light_list_begin
for (i = 0; i < MAX_SUBSET_LEN; ++i) {
    light_idx = light_offset + small_rnd()
    if (light_idx >= light_list_end) {
        // Detect if hit_emitter is in current stride.
        hit_caught ||=   light_offset < light_list_end
                      && light_pointers[light_offset] <= hit_emitter
        break
    }
    w = light_contrib(v, p, n, light_pointers[light_idx])
    // Accumulate all weights outside hit_emitter's stride.
    wo = w
    if (not hit_caught && hit_emitter <= light_pointers[light_idx]) {
        // Is the emitter in this or the last stride?
        if (light_pointers[light_offset] <= hit_emitter)
            wo = 0 // This stride
        else
            pending_weight = 0 // Last stride
        hit_caught = true // Found hit_emitter
    }
    other_weights += pending_weight
    pending_weight = wo

    if (w > 0) {
        τ = total_weights / (total_weights+w); total_weights += w
        if (ξ1 < τ) { ξ1 /= τ }
        else { selected = (light_pointers[light_idx], w); ξ1 = (ξ1-τ) / (1-τ) }
        ξ1 = clamp(ξ1, 0, MAX_BELOW_ONE)
    }
    light_offset += subset_stride
}
// Compute pseudo-marginal probability of sampling hit_emitter.
if (hit_caught)
    other_weights += pending_weight
hit_w = light_contrib(v, p, n, hit_emitter)
hit_probability = hit_w / ((other_weights + hit_w) * S)

probability = selected.mass / (total_weights * S)
return (selected.light, probability, hit_probability)
```

47.6.1 INDEPENDENTLY SELECTING LIGHTS PER STRIDE

In order to independently sample one light per light list stride, we use a fast linear congruential generator (`FastRng` in Listing 47-3) to generate integers in $[0, S - 1]$ in each iteration of the loop (line 15). Note that simply using one floating-point random number, as in the incremental sampling scheme of the previous section, is not generally feasible because the number of random bits may be exceeded even for lower stride widths and counts.

47.6.2 IDENTIFYING THE STRIDE OF HIT EMITTERS

Once a ray hits an emitter by chance, e.g., by BSDF sampling, we need be able to identify its subset stride in the light sampling process and replace the corresponding importance sampling weight therein by that of the hit emitter. It is nontrivial to keep track of the offsets of the emitters in all light lists of the scene because, as in any real-world application, there are likely many light lists adapted to capture the relevant illumination for different shading points.

We tackle this problem by enforcing an order for emitters in light lists, i.e., we introduce an (arbitrary) order defined by the memory location of each emitter. Once all light lists are sorted in ascending order, we can infer the strides that potentially contain a given hit emitter `hit_emitter` during light sampling by simple pointer comparisons with other lights in the subset (Listing 47-3, lines 18 and 23). This allows us to obtain the weight sum `other_weights` that only contains the weights of lights in the current subset that do not overlap with the stride of the hit emitter.

Finally, we are able to compute the pseudo-marginal probability required for MIS at the hit emitter (line 48), knowing the total weight `hit_w` + `other_weights` of the corresponding alternative subset (compare to Figure 47-3).

47.7 RESULTS AND DISCUSSION

All variants of our method run in real time on an NVIDIA GeForce RTX 2070, at a resolution of 1920×1080 at 11–17 ms of frame time (including denoising). We implemented our method in the open source Q2VKPT project [25], which is based on Vulkan using the hardware-accelerated ray tracing extension. We show unfiltered one-sample-per-pixel path tracer outputs with one indirect light bounce and outputs that were filtered using a variant of spatiotemporal filtering [27].

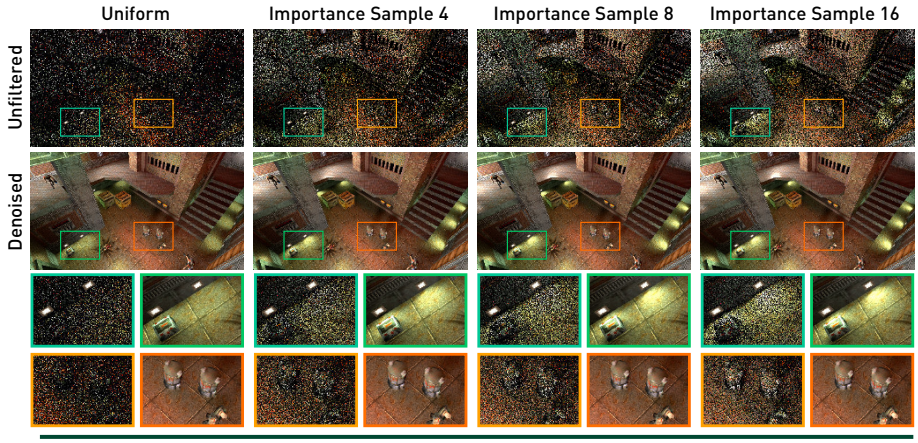

Figure 47-4. *Light importance sampling with various stochastic subset sizes, at one sample per pixel, unfiltered (top) and denoised with ASVGF [27] (bottom). The quality of light sampling increases with the size of the stochastic subsets, balancing more samples with respect to other lights. With uniform sampling, even clamped path tracing produces high variance, as lights are randomly far away or nearby, in or out of the focus of specular highlights. This leads to unstable results, even after temporal filtering, causing flickering in motion. Importance sampling in stochastic subsets is effective in reducing variance, bringing results closer to a converged result even at one sample per pixel.*

47.7.1 RUNTIMES

The path tracer runs at 7–9 ms per frame with one indirect bounce and a stochastic light subset size of 8. Randomizing the subset by choosing individual offsets per stride as in Section 47.6 adds about 0.1 ms compared to the randomization with regular intervals as in Section 47.4. Enabling MIS adds another 1–1.5 ms, as it requires computing and tracking a few additional quantities and additional emitter texture accesses for the hit points of BSDF-sampled rays.

47.7.2 SUBSET SIZES

As shown in Figure 47-4, subset sizes affect the quality of results, demonstrating how good importance sampling improves the performance of a Monte Carlo path tracer. Especially for smaller subset sizes, the resulting noise is reduced significantly with every additional light in the subset. For Q2VKPT, we chose a subset size of 8 as a trade-off between sampling quality and performance because it gave decent results that looked stable after denoising. In our (unprofiled, not thoroughly optimized) implementation, going from a subset size of 8 to 16 adds 1 ms, going to 32 another 1.5 ms, and up to 64 another 3 ms.

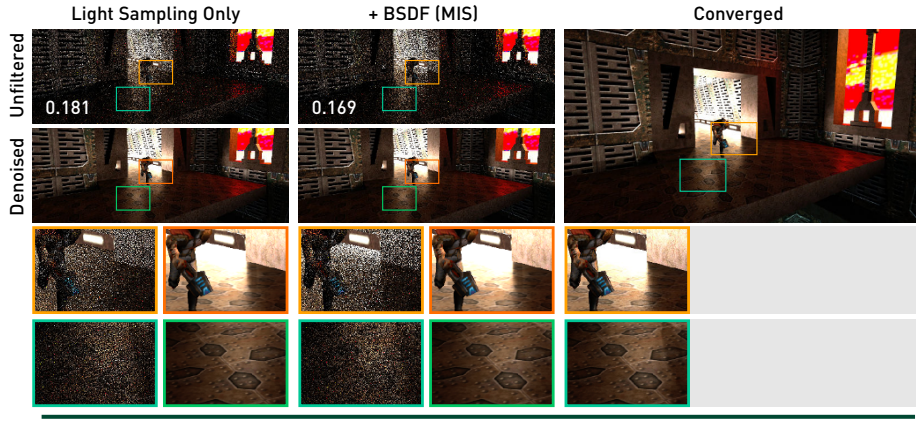

Figure 47-5. *Our light sampling with and without MIS-weighted BSDF samples, at one sample per pixel, unfiltered (top) and denoised (bottom). As expected, the RMSE numbers (top row) decrease with our pseudo-marginal MIS. Top inset row: MIS makes up for cases where our light sampling alone underestimates contributions and samples the floor lights too rarely. Without MIS, we would need to clamp these samples to obtain robust denoising results under motion. Bottom inset row: unfortunately, the additional randomization, required for our pseudo-marginal MIS, destroys the stratification of pixel error in screen space. Under motion, this results in slightly less stable denoising results.*

47.7.3 MULTIPLE IMPORTANCE SAMPLING

The impact of MIS (with emitters selected by light sampling and emitters hit by BSDF sampling) on our results is shown in Figures 47-5 and 47-6. There are two noticeable cases where MIS with BSDF sampling makes up for typical shortcomings of light sampling approaches. In the first case, around the center of a specular highlight, when the corresponding emitter is nearby, different points on the emitter can have very different contributions to the shaded point. Here, our approximate product importance sampling fails to predict the exact contribution of the emitter as a whole, and thus assigns misleading importance weights to it. The other typical failure case affects points in direct proximity of emitters, where the inverse squared distance law of light falloff results in individual emitter points contributing unbounded sample values for light sampling. In both cases, MIS identifies BSDF sampling as the better sampling technique, because its probability density covers both specular peaks and nearby points well. As a result, light samples are downweighted and BSDF samples upweighted in these locations, making up for the shortcomings of light sampling and ensuring stable shading estimates.

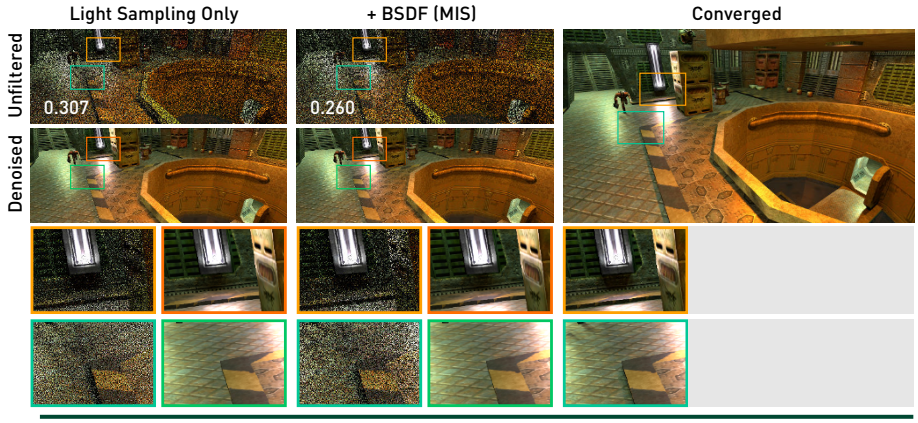

Figure 47-6. *Our light sampling with and without MIS-weighted BSDF samples, at one sample per pixel, unfiltered (top) and denoised (bottom). Root mean square error numbers (RMSE) are inset. See Figure 47-5 for discussion.*

47.7.4 STRATIFICATION

Our stratified subset sampling strategy without MIS (Section 47.4) works well with stratified random variables (in our case blue noise points), as is visible in Figure 47-7 (left). Unfortunately, this is no longer the case when we add pseudo-marginal MIS (Section 47.6). The required additional randomization destroys the correspondence between convex random variable intervals and light sources, destroying the positive effect of stratification. In Figure 47-7 (middle) we can see the noticeable clumping of white noise that we would expect from independent random variables. Regrettably, the resulting higher variance in the denoised output in most cases cancels out any benefits we received from implementing our MIS. Therefore, it is likely a better idea to use the variant without MIS and instead improve on the product importance estimation in the future. To verify that the additional randomization is the problem, Figure 47-7 (right) shows a biased variant of MIS that fixes the stratification problems, at the cost of an unpredictable systematic error in the rendered output.

47.8 CONCLUSIONS

Variance reduction techniques like importance sampling and MIS are crucial tools not only in offline rendering, but even more so in upcoming real-time path tracing applications with low sample counts. For Q2VKPT, we found that

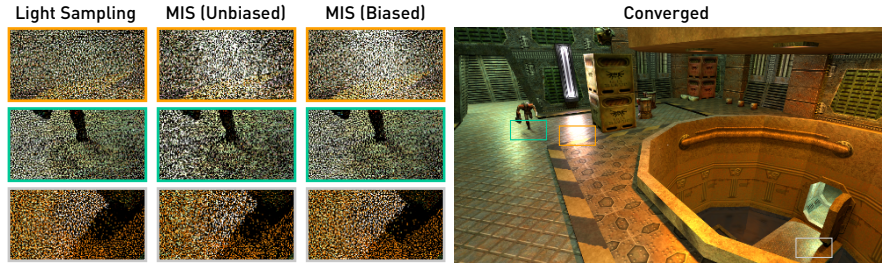

Figure 47-7. *Left: our stratified subset sampling works well with blue noise points, and the error is nicely diffused in screen space. Middle: unfortunately, our unbiased pseudo-marginal MIS cannot profit from stratified random variables. In the noise, we can see the typical clumping of white noise, which we would expect from independent random variables. Using pure light sampling and developing more accurate product importance sampling strategies is therefore likely a more promising avenue for future work. Right column: we can verify that the additional randomization required by pseudo-marginal MIS is the culprit. Disabling it leads to a biased result, but recovers the stratification effects in screen space. Note that this will only look similar as long as the lights contained in each stratified subset are sufficiently representative of the full illumination. In practice, the unpredictable bias is likely undesirable.*

even a simplistic resampling method, working with small stochastic subsets of the full scene illumination, can bring tremendous quality improvements. The feasibility of real-time Monte Carlo rendering with denoising depends on such strategies of variance control, as they directly affect the amount of reliable information in the framebuffer that can be used to reconstruct correct and stable results. More recent developments than our approaches like ReSTIR [4] and real-time path guiding [11] make good steps toward even more robust techniques, which in some cases barely require denoising at all. We are hopeful that these techniques can be combined in their robustness and generality, leading to even more reliable and versatile rendering algorithms in the future.

Finally, in this chapter we explored the benefits and challenges of adding MIS to the light sampling technique that was released with Q2VKPT. We proposed pseudo-marginal stochastic MIS to improve some difficult corner cases where our approximate product importance sampling fails. In our use case, we found its usefulness to be limited due to the resulting loss of stratification benefits. It will likely be made obsolete by more sophisticated real-time light sampling strategies and future product importance sampling techniques. Nevertheless, we hope that our look into pseudo-marginal stochastic MIS may serve as an inspiration for the design of variance reduction techniques in other use cases where time for the exhaustive evaluation of alternative PDFs is limited.

ACKNOWLEDGMENTS

We thank Christoph Schied, Addis Dittebrandt, and Christoph Peters for many insightful discussions around light sampling and the Q2VKPT project. Q2VKPT was built by Christoph Schied [25], and we thank him for creating the opportunity of contributing to the project, with additional contributions coming from Johannes Hanika, Addis Dittebrandt, Florian Reibold, Stephan Bergmann, Emanuel Schrade, Alisa Jung, and Christoph Peters.

REFERENCES

[1] Arvo, J. Stratified sampling of spherical triangles. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, pages 437–438, 1995. DOI: 10.1145/218380.218500.

[2] Bikker, J. *Ray Tracing for Real-Time Games*. PhD thesis, Technische Universiteit Delft, 2012.

[3] Bikker, J. Real-time ray tracing through the eyes of a game developer. In *IEEE Symposium on Interactive Ray Tracing*, page 1, 2007. DOI: 10.1109/RT.2007.4342583.

[4] Bitterli, B., Wyman, C., Pharr, M., Shirley, P., Lefohn, A., and Jarosz, W. Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 39(4):148:1–148:17, 2020. DOI: 10.1145/3386569.3392481.

[5] Burke, D., Ghosh, A., and Heidrich, W. Bidirectional importance sampling for direct illumination. In *Proceedings of the Sixteenth Eurographics Conference on Rendering Techniques*, pages 147–156, 2005.

[6] Burley, B., Adler, D., Chiang, M. J.-Y., Driskill, H., Habel, R., Kelly, P., Kutz, P., Li, Y. K., and Teece, D. The design and evolution of Disney's Hyperion renderer. *ACM Transactions on Graphics*, 37(3):33:1–33:22, 2018. DOI: 10.1145/3182159.

[7] Chaitanya, C. R. A., Kaplanyan, A. S., Schied, C., Salvi, M., Lefohn, A., Nowrouzezahrai, D., and Aila, T. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics*, 36(4):98:1–98:12, 2017. DOI: 10.1145/3072959.3073601.

[8] Chao, M.-T. A general purpose unequal probability sampling plan. *Biometrika*, 69(3):653–656, 1982.

[9] Christensen, P., Fong, J., Shade, J., Wooten, W., Schubert, B., Kensler, A., Friedman, S., Kilpatrick, C., Ramshaw, C., Bannister, M., et al. RenderMan: An advanced path-tracing architecture for movie rendering. *ACM Transactions on Graphics*, 37(3):30:1–30:21, 2018. DOI: 10.1145/3182162.

[10] Conty Estevez, A. and Kulla, C. Importance sampling of many lights with adaptive tree splitting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(2):25:1–25:17, 2018. DOI: 10.1145/3233305.

[11] Dittebrandt, A., Hanika, J., and Dachsbacher, C. Temporal sample reuse for next event estimation and path guiding for real-time path tracing. In *Eurographics Symposium on Rendering—DL-Only Track*, pages 39–51, 2020. DOI: 10.2312/sr.20201135.

[12] Elvira, V., Martino, L., Luengo, D., and Bugallo, M. F. Generalized multiple importance sampling. *Statistical Science*, 34(1):129–155, 2019. DOI: 10.1214/18-STS668.

[13] Fascione, L., Hanika, J., Leone, M., Droske, M., Schwarzhaupt, J., Davidovič, T., Weidlich, A., and Meng, J. Manuka: A batch-shading architecture for spectral path tracing in movie production. *ACM Transactions on Graphics*, 37(3):31:1–31:18, 2018. DOI: 10.1145/3182161.

[14] Georgiev, I., Ize, T., Farnsworth, M., Montoya-Vozmediano, R., King, A., Lommel, B. V., Jimenez, A., Anson, O., Ogaki, S., Johnston, E., Herubel, A., Russell, D., Servant, F., and Fajardo, M. Arnold: A brute-force production path tracer. *ACM Transactions on Graphics*, 37(3):32:1–32:12, 2018. DOI: 10.1145/3182160.

[15] Georgiev, I., Krivanek, J., Popov, S., and Slusallek, P. Importance caching for complex illumination. *Computer Graphics Forum*, 31(2pt3):701–710, 2012. DOI: 10.1111/j.1467-8659.2012.03049.x.

[16] Heitz, E., Dupuy, J., Hill, S., and Neubelt, D. Real-time polygonal-light shading with linearly transformed cosines. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 35(4):41:1–41:8, July 2016. DOI: 10.1145/2897824.2925895.

[17] Heitz, E. and Hill, S. Linear-light shading with linearly transformed cosines. In W. Engel, editor, *GPU Zen Advanced Rendering Techniques*, pages 137–162. Black Cat Publishing, 2017.

[18] Keller, A., Wächter, C., Raab, M., Seibert, D., van Antwerpen, D., Korndörfer, J., and Kettner, L. The iray light transport simulation and rendering system. In *ACM SIGGRAPH 2017 Talks*, 34:1–34:2, 2017. DOI: 10.1145/3084363.3085050.

[19] Lin, D. and Yuksel, C. Real-time stochastic lightcuts. *Proceedings of the ACM on Computer Graphics and Interactive Techniques (Proceedings of I3D 2020)*, 3(1):5:1–5:18, 2020. DOI: 10.1145/3384543.

[20] Moreau, P. and Clarberg, P. Importance sampling of many lights on the GPU. In E. Haines and T. Akenine-Möller, editors, *Ray Tracing Gems*, pages 255–283. Apress, 2019.

[21] Moreau, P., Pharr, M., and Clarberg, P. Dynamic many-light sampling for real-time ray tracing. In *High-Performance Graphics 2019—Short Papers*, 2019. DOI: 10.2312/hpg.20191191.

[22] Müller, T., Gross, M., and Novák, J. Practical path guiding for efficient light-transport simulation. *Computer Graphics Forum*, 36(4):91–100, 2017. DOI: 10.1111/cgf.13227.

[23] Peters, C. BRDF importance sampling for polygonal lights. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 38(4), July 2021. To appear.

[24] Peters, C. and Dachsbacher, C. Sampling projected spherical caps in real time. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(1):1:1–1:16, June 2019. DOI: 10.1145/3320282.

[25] Schied, C. Q2VKPT. http://brechpunkt.de/q2vkpt/, with source code at https://github.com/cschied/q2vkpt, 2019.

[26] Schied, C., Kaplanyan, A., Wyman, C., Patney, A., Chaitanya, C. R. A., Burgess, J., Liu, S., Dachsbacher, C., Lefohn, A., and Salvi, M. Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination. In *Proceedings of High Performance Graphics*. ACM, 2017.

[27] Schied, C., Peters, C., and Dachsbacher, C. Gradient estimation for real-time adaptive temporal filtering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(2):24:1–24:16, 2018. DOI: 10.1145/3233301.

[28] Schmittler, J., Pohl, D., Dahmen, T., Vogelgesang, C., and Slusallek, P. Realtime ray tracing for current and future games. In *ACM SIGGRAPH 2005 Courses*, 23–es, 2005. DOI: 10.1145/1198555.1198762.

[29] Shirley, P., Wang, C., and Zimmerman, K. Monte Carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, 15(1):1–36, 1996. DOI: 10.1145/226150.226151.

[30] Talbot, J., Cline, D., and Egbert, P. Importance resampling for global illumination. In *Proceedings of the Eurographics Workshop on Rendering*, pages 139–146, 2005. DOI: 10.2312/EGWR/EGSR05/139-146.

[31] Veach, E. and Guibas, L. J. Optimally combining sampling techniques for Monte Carlo rendering. In pages 419–428, 1995.

[32] Vorba, J., Karlík, O., Šik, M., Ritschel, T., and Křivánek, J. On-line learning of parametric mixture models for light transport simulation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 33(4):101:1–101:11, 2014. DOI: 10.1145/2601097.2601203.

[33] Walter, B., Fernandez, S., Arbree, A., Bala, K., Donikian, M., and Greenberg, D. P. Lightcuts: A scalable approach to illumination. *ACM Transactions on Graphics*, 24(3):1098––1107, 2005. DOI: 10.1145/1073204.1073318.

[34] West, R., Georgiev, I., Gruson, A., and Hachisuka, T. Continuous multiple importance sampling. *ACM Transactions on Graphics*, 39(4):136:1–136:12, 2020. DOI: 10.1145/3386569.3392436.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (http://creativecommons.org/licenses/by-nc-nd/4.0/), which permits any noncommercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if you modified the licensed material. You do not have permission under this license to share adapted material derived from this chapter or parts of it.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.