# Real-Time Markov Chain Path Guiding for Global Illumination and Single Scattering

LUCAS ALBER, JOHANNES HANIKA, and CARSTEN DACHSBACHER,
Karlsruhe Institute of Technology, Germany


Fig. 1. Left: A game scene from the Arcane Dimensions Quake mod with path traced 2-bounce indirect lighting and single scattering from many dynamic light sources rendered in real-time on an AMD Radeon RX 7900 XTX at 1920 × 1080. From left to right: path tracing with BSDF importance sampling, our new unbiased estimator, our estimator with SVGF denoising. Right: A custom pool scene showing rendering of underwater caustics using our estimator, from left to right: path tracing using phase function importance sampling, equal error render using our estimator, equal sample render using our estimator.

We present a lightweight and unbiased path guiding algorithm tailored for real-time applications with highly dynamic content. The algorithm demonstrates effectiveness in guiding both direct and indirect illumination. Moreover, it can be extended to guide single scattering events in participating media. Building upon the screen-space approach by Dittebrandt et al. [2023], the incident light distribution is represented as a von Mises-Fisher mixture model, which is controlled by a Markov chain process. To extend the procedure to world space, our algorithm uses a unique Markov chain architecture, which resamples Markov chain states from an ensemble of hash grids. We combine multi-resolution adaptive grids with a static grid, ensuring rapid state exchange without compromising guiding quality. The algorithm imposes minimal prerequisites on scene representation and seamlessly integrates into existing path tracing frameworks. Through continuous multiple importance sampling, it remains independent of the equilibrium distribution of Markov chain and hash grid resampling. We perform an evaluation of the proposed methods across diverse scenarios. Additionally, we explore the algorithm's viability in offline scenarios, showcasing its effectiveness in rendering volumetric caustics. We demonstrate the application of the proposed methods in a path tracing engine for the original Quake game. The demo project features path traced global illumination and single scattering effects at frame rates over 30 FPS on NVIDIA's GeForce 20 series or AMD's Radeon RX 6000 series without upscaling.

CCS Concepts: • **Computing methodologies** → **Ray tracing**.

Additional Key Words and Phrases: Ray Tracing, Global Illumination, Path Guiding, Markov Chain, Hash Grid

Authors' Contact Information: Lucas Alber, lucas.alber@kit.edu; Johannes Hanika, hanika@kit.edu; Carsten Dachsbacher, dachsbacher@kit.edu, Karlsruhe Institute of Technology, Karlsruhe, Germany.

© 2025 Copyright held by the owner/author(s).
This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, https://doi.org/10.1145/3728296.

**ACM Reference Format:**
Lucas Alber, Johannes Hanika, and Carsten Dachsbacher. 2025. Real-Time Markov Chain Path Guiding for Global Illumination and Single Scattering. *Proc. ACM Comput. Graph. Interact. Tech.* 8, 1, Article 15 (May 2025), 18 pages. https://doi.org/10.1145/3728296

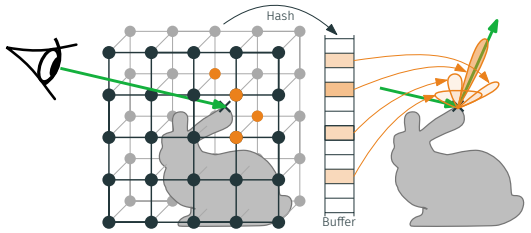


Fig. 2. High-level schematic of our Markov chain path guiding scheme: we use hash grids to store a set of Markov chains controlling von Mises-Fisher distributions in world space. To reconstruct an estimator for the incident light field, we resample states from the grid and the resulting stochastic mixture model is evaluated using stochastic multiple importance sampling (SMIS) [West et al. 2020]. The mixture is updated with maximum likelihood estimation and MCMC acceptance tests.

## 1 Introduction

Over the years, path tracing has emerged as a cornerstone technique in the pursuit of visually stunning graphics in the visual effects industry [Droske et al. 2023]. The fundamental idea involves tracing rays of light through virtual environments, similar to how photons interact with matter, and employing numerical integration techniques to estimate the light received on a virtual camera sensor. This approach grants high flexibility and generality to photorealistic visual effects workflows, while allowing the rendered results to accurately replicate the visual characteristics of the real world. Recently, the possibility of incorporating path tracing into real-time applications has emerged, driven by the widespread adoption of hardware-accelerated ray tracing on GPUs [NVIDIA 2018]. However, even with modern hardware, only a limited number of paths per pixel can be traced while maintaining real-time frame rates. To make the technique applicable, algorithmic improvements have been proposed to gather as much information as possible. For example, ReSTIR-based methods share light samples between neighboring pixels and frames using resampling techniques [Bitterli et al. 2020]. However, these methods often suffer from correlation artifacts, additional rays are required to maintain unbiasedness, and reasoning about convergence is challenging [Lin et al. 2022]. Additionally, denoisers are employed to alleviate the remaining noise.

In offline rendering, path guiding stands out as a standard technique for handling complex scenes [Vorba et al. 2019]. Nevertheless, a significant portion of current techniques require costly fitting procedures, depend on hierarchical spatial data structures, which apply only to static scenes or necessitate complex structural updates, which has proven to be inefficient on GPU architectures. We explore a dynamic game environment where light lists are not available, and emissive surfaces can appear and disappear anywhere and anytime. This setting imposes minimal prerequisites on scene representation and animation, but limits discovery of lights to hemisphere sampling.

We introduce *Markov Chain Path Guiding* (MCPG), a real-time path guiding method tailored for such dynamic environments based on von Mises-Fisher (vMF) mixture models, which we select for their efficient and robust online fitting procedure [Ruppert et al. 2020]. The parameter space is controlled via a population of Markov chains, which are stored in an ensemble of world-space hash grids (see Figure 2). A stochastic resampling process, integrating into the Markov chain equilibrium, exchanges states with adjacent world-space cells. Our method is based on the work of Dittebrandt et al. [2023], which uses continuous multiple importance sampling [West et al. 2020] to ensure independence from the equilibrium distribution. More specifically, our contributions are:

- an extension to indirect illumination by storing Markov chain states in an ensemble of world space hashed grids at a constant memory footprint, a stochastic resampling process selects the most important states while maintaining diversity,
- estimators which are stable in highly dynamic environments, using an irradiance cache for online training on a GPU,
- fast adoption of the guiding distribution in dynamic environments without tracing additional rays using an empirical prior as well as an effective heuristic to invalidate states when light sources disappear,
- an extension to single scattering, including underwater volumetric caustics.

The resulting algorithm can produce smooth, temporally stable path traced images, including indirect lighting and single scattering in a few milliseconds at full HD resolution, even on previous generation hardware from AMD's Radeon RX 6000 series or NVIDIA's GeForce 20 series such as the AMD Radeon RX 6800 XT or the NVIDIA GeForce RTX 2080 Ti.

## 2 Background

In the following, we summarize the most important concepts from physics and Monte Carlo theory.

*Rendering Equation.* The emitted radiance at a surface point $\boldsymbol{z}$ in direction $\boldsymbol{\omega}$ is described by the surface rendering equation [Kajiya 1986]

$$L(\boldsymbol{z}, \boldsymbol{\omega}) = L_\mathrm{e}(\boldsymbol{z}, \boldsymbol{\omega}) + \int_{\mathcal{S}^2} f_\mathrm{r}(\boldsymbol{z}, \boldsymbol{\omega}_\mathrm{i}, \boldsymbol{\omega}) L_\mathrm{i}(\boldsymbol{z}, \boldsymbol{\omega}_\mathrm{i}) \left|\mathsf{n}(\boldsymbol{z}) \cdot \boldsymbol{\omega}_\mathrm{i}\right| \,\mathrm{d}\boldsymbol{\omega}_\mathrm{i}\,, \tag{1}$$

where $L_\mathrm{e}(\boldsymbol{z}, \boldsymbol{\omega})$ describes the emitted radiance at the surface point $\boldsymbol{z}$ in direction $\boldsymbol{\omega}$ and the integral over the unit sphere $\mathcal{S}^2$ describes the amount of incident radiance $L_\mathrm{i}$ that is reflected towards $\boldsymbol{\omega}$. $f_\mathrm{r}$ is the BRDF and $|\mathsf{n}(\boldsymbol{z}) \cdot \boldsymbol{\omega}_\mathrm{i}|$ is a foreshortening term due to the incident radiance direction $\boldsymbol{\omega}_\mathrm{i}$ and the surface normal $\mathsf{n}(\boldsymbol{z})$. For volume rendering, we also collect the emitted $L_\mathrm{e}$ and in-scattered $L_\mathrm{s}$ contributions from all points $\boldsymbol{y}$ along the ray $(\boldsymbol{x}, -\boldsymbol{\omega})$ up to the first surface point $\boldsymbol{z}$ in the medium:

$$L(\boldsymbol{x}, \boldsymbol{\omega}) = \int_0^{\|\boldsymbol{z}-\boldsymbol{x}\|} T(\boldsymbol{x}, \boldsymbol{y}) \left[\mu_\mathrm{a}(\boldsymbol{y}) L_\mathrm{e}(\boldsymbol{y}, \boldsymbol{\omega}) + \mu_\mathrm{s}(\boldsymbol{y}) L_\mathrm{s}(\boldsymbol{y}, \boldsymbol{\omega})\right] \,\mathrm{d}y + T(\boldsymbol{x}, \boldsymbol{z}) L(\boldsymbol{z}, \boldsymbol{\omega})\,, \tag{2}$$

where $\boldsymbol{y} = \boldsymbol{x} - y\boldsymbol{\omega}$, $\mu_\mathrm{a}$, $\mu_\mathrm{s}$ the collision coefficients for absorption, and scattering and $T$ the transmittance in the medium [Novák et al. 2018].

*Parametric mixture models.* Parametric mixture models are a type of statistical model that assumes that the data is generated from a mixture of several probability distributions, each with its own fixed and finite set of parameters. These underlying distributions are characterized by a specific probability distribution. The probability density function of the mixture model

$$p(\boldsymbol{x} \mid \boldsymbol{\pi}, \boldsymbol{\theta}) = \sum_{i=1}^{K} \pi_i p(\boldsymbol{x} \mid \theta_i)\,, \quad \boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)\,, \boldsymbol{\theta} = (\theta_i, \ldots, \theta_K) \tag{3}$$

is a weighted sum of the individual component probability density functions, where $\pi_i$ represents the mixing proportion of the $i$-th component, and $p_i(\boldsymbol{x} \mid \theta_i)$ is the probability density function of the $i$-th component with parameters $\theta_i$. The von Mises-Fisher distribution is a probability density on the hypersphere, suitable to model directional data. Its PDF is defined as

$$p(\boldsymbol{\omega}|\theta_i) = \frac{\kappa_i}{4\pi \sinh \kappa_i} \exp(\kappa_i \boldsymbol{\mu}_i^T \boldsymbol{\omega}) \tag{4}$$

where $\theta_i = (\boldsymbol{\mu}_i, \kappa_i)$, and $\boldsymbol{\mu}_i$ and $\kappa_i$ are the mean direction and concentration parameter of the $i$-th component. For small $\kappa$ the distribution approaches the uniform distribution. Special care has to be taken to ensure numerical stability in finite precision arithmetic [Jakob 2012; Tokuyoshi 2025].

## 3 Previous Work

*ReSTIR.* Building on resampled importance sampling [Talbot et al. 2005] and weighted reservoir sampling [Chao 1982], Bitterli et al. [2020] introduced ReSTIR, a resampling algorithm for direct illumination. In order to enhance sample quality, light samples are reused from previous frames and exchanged with nearby pixels. ReSTIR utilizes a screen-space data structure that maintains a reservoir for each pixel, which can be merged to achieve higher quality sample sets while preserving computational efficiency. The unbiased variant necessitates meticulous selection of neighbors and additional shadow rays for each neighboring pixel, thereby increasing computational demands. ReSTIR demonstrated a significant reduction in error relative to previous light sampling techniques. As a resampling scheme, ReSTIR is constrained to explicit light source information or hemisphere samples and can suffer from correlation artefacts. Sawhney et al. [2023] alleviated these artifacts using MCMC mutations of the reservoirs. Furthermore, the resampling process requires an explicit BSDF evaluation, potentially impacting performance with more sophisticated material models. Although the original ReSTIR algorithm is limited to direct illumination, its fundamental principles provide a foundational framework for recent research, exploring the reuse of samples in world space and longer paths for global illumination and volume rendering [Boissé 2021; Boksansky et al. 2021; Lin et al. 2021; Majercik et al. 2021; Ouyang et al. 2021]. However, implementation efforts and considerations regarding unbiasedness and convergence in dynamic scenes become increasingly complex [Lin et al. 2022].

*Path Guiding.* There is a wide plethora of techniques which use an estimate of the scene's light distribution, acquired either on-the-fly during rendering or pre-learned in advance, to guide paths into important directions. The techniques differ in the spatial data structure that is used to store the caches [Bashford-Rogers et al. 2012; Hey and Purgathofer 2002; Jensen 1995; Lafortune and Willems 1995; Lu et al. 2024; Müller et al. 2017], the data structure and domains of the caches themselves [Dodik et al. 2022; Müller et al. 2017; Reibold et al. 2018; Ruppert et al. 2020; Vorba et al. 2014; Zheng and Zwicker 2019] and the approximated target function [Herholz et al. 2016; Rath et al. 2020]. Deep learning [Bako et al. 2019; Müller et al. 2019; Zheng and Zwicker 2019] and reinforcement learning [Dahm and Keller 2017] have also been applied to path guiding. However, most offline path guiding methods prove impractical for real-time applications. For example, these methods may employ tree-like data structures [Droske et al. 2023; Lafortune and Willems 1995; ler et al. 2022; Müller et al. 2017], which are inefficient to traverse on GPU architectures. Some require computationally demanding fitting of mixture models [Ruppert et al. 2020; Vorba et al. 2014] or training of neural networks [Dong et al. 2023], and some may involve dedicated learning phases [Müller et al. 2017; Vorba et al. 2014] and lack the adaptability to accommodate dynamic changes in scene geometry or lighting conditions. Path guiding might even be disabled in production environments to avoid overhead in simpler cases [Vorba et al. 2019].

Table 1. Overview of data that is stored for every Markov chain state at hash grid vertices. We use luminance weighted mean light source positions $\overline{\boldsymbol{y}}$ and mean cosines of directions $\overline{r}$ for our fitting procedure [Ruppert et al. 2020]. The values are reconstructed by dividing $\overline{\boldsymbol{y}}$ and $\overline{r}$ by $\overline{w}$, respectively.

| Symbol | Meaning |
|---|---|
| $\overline{w}$ | luminance estimate of Equation (1), used as resampling weight |
| $N$ | number of samples |
| $\overline{\boldsymbol{y}}$ | luminance-weighted mean of light source positions |
| $\overline{r}$ | luminance-weighted mean cosine of light source directions |
| $T$ | time of last update |
| $\boldsymbol{v}$ | last known velocity of the light source |
| hash | hashed grid position for collision detection |

*Markov Chain Mixture Models.* In the recent research of Dittebrandt et al. [2023], a real-time guiding method for direct light was presented, using screen-space mixture models. Each pixel stores the state of a Markov chain that runs in the parameter space of a vMF distribution. The components of neighboring pixels are combined in a randomized mixture model to obtain an estimate of the full distribution. The corresponding vMF component is updated using a maximum likelihood estimation step, and a Markov chain Monte Carlo acceptance test decides whether the new state should be shared with neighboring pixels. Temporally dependent samples are decorrelated using a shuffle of Markov chain states between neighboring pixels after every update, which can be implemented as an efficient operation on modern GPUs. Stochastic multiple importance sampling (SMIS) is employed to seamlessly integrate the model into the path tracer without knowing the equilibrium distribution of the Markov chain process, since the PDF cancels in the estimator. The Markov chain mixture model guiding method quickly adapts to new lighting conditions, at a computational overhead similar to that of ReSTIR. A notable downside is the tendency to overuse dominant light sources and parameter-space proximity bias when sharing lobes of occluded light sources, which manifests as increased variance at shadow edges. A drawback of utilizing a screen space data structure is the absence of information about off-screen surfaces, leading to potentially unstable results when moving the camera. Additionally, the approach cannot effectively guide multi-bounce global illumination.

## 4 Hashed Markov Chain Path Guiding (MCPG)

*Overview.* Figure 3 illustrates the sampling and update procedure of our method, which we detail in the following sections for a surface interaction at hit. The system consists of three main components: (a) an ensemble of world space hash grids, storing Markov chain states, operating on sufficient statistics of a vMF distribution, which approximate the incident light field as a continuous mixture model. (b) a four-step sampling and update procedure, involving a stochastic resampling process, a maximum-likelihood estimation of the vMF lobes, and a Markov chain Monte Carlo (MCMC) acceptance step, and (c) an irradiance cache to prevent backtracking of light paths on the GPU for training and to stabilize the maximum-likelihood estimation of the individual lobes.

*Hash Grid.* Inspired by multi-resolution hash grid encodings for neural networks [Müller et al. 2022], we store Markov chain states at the vertices of a multi-resolution hash grid, which we refer to as the adaptive grid. The grid adaptively selects the appropriate resolution level for each world space point based on the camera distance, thereby ensuring a roughly constant projected area on the screen. To exchange states between cells and to access preexisting states during camera movement

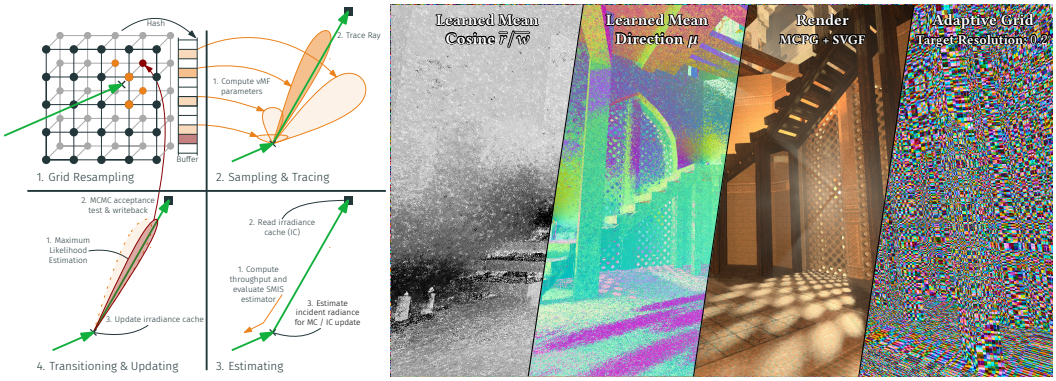


Fig. 3. Left: Overview of our Markov chain path guiding system procedure in four steps: (1) a stochastic mixture is obtained by resampling from near-by positions in the hash grid, (2) the mixture is sampled, and an indirect ray is traced, (3) the SMIS estimator is evaluated, and (4), the result is used to update the local irradiance cache, the guiding distribution is adjusted by maximum likelihood estimation, and an MCMC acceptance step is performed before potentially writing back to the grid. Right: Illustrations from our technique; from left to right: the learned mean cosine and mean direction of a converged guiding distribution, the final render and the target level of our adaptive grid. Frequency limitations arise from the hash grid resolution only, not scene geometry.

and grid refinement, we incorporate stochastic access to coarser levels. We choose a target grid resolution that matches the level of detail and lighting frequency of the scene, as illustrated in Figure 3 (right). Additionally, to ensure efficient and reliable exchange between adaptive grid borders, we introduce a coarse hash grid with a constant cell width, which we refer to as the static grid. To alleviate block artifacts, we interpolate between grid vertices through a stochastic process, accessing a vertex with a probability proportional to the trilinear interpolation weight. For a buffer of size $B$, we obtain the buffer index of a grid vertex by hashing its integer grid coordinate $\boldsymbol{x}$ according to $\mathsf{hash}(\boldsymbol{x}) = \bigoplus_{i=1}^{d} \pi_i x_i \mod B$, with $\pi_i$ being large prime numbers and $\oplus$ the bit-wise XOR operation. Müller et al. [2022] use $\pi_1 = 1$ for better cache coherency. To prevent information from leaking through surfaces and around sharp corners, where the lighting is likely to change, we incorporate the surface normal into the hash for the adaptive grid buffer. For that, we computed the cube map index based on the surface normal and hash the result along with the grid coordinates. We conducted experiments to augment the hash with a random number within the range $[0, i]$, thereby implicitly storing $i$ Markov chain states at each vertex. This approach aligns with the concepts introduced in the context of jittered spatial hashing [Boissé 2021]. However, our findings did not reveal any discernible benefit from this adjustment. In addition to the statistics required to operate the Markov chain, we store the grid position, hashed with a separate hash function, to detect collisions between two grid vertices that map to the same buffer index, similar to the concept proposed in the work by [Binder et al. 2018]. An overview of the data stored for every Markov chain state can be found in Table 1.

*Step 1: Grid Resampling.* In the first step, the algorithm chooses a promising state $s$ from the Markov chain hash grids out of $N_{\text{mc}}$ candidates, as outlined in Algorithm 1. The selection process employs weighted reservoir resampling [Chao 1982], choosing a state with a probability proportional to its luminance estimate $\overline{w}$ of the incident light (Equation (1)) in a single pass. This methodology shares similarities with the resampling procedure utilized in ReSTIR. Regardless of the specific state selected, the distribution parameters of all accessed states are retained in memory

for SMIS in Step 3. The candidates are obtained independently: Based on a fixed probability, the grid.load procedure selects the static or adaptive grid (we use $p_{\text{adaptive}} = 0.7$) and an exponential distribution is sampled to select a level that is coarser than the target level of the adaptive grid. In particular, duplicate states in multiple grid locations are not explicitly prohibited for the sake of simplicity and efficiency. The hash grid already ensures proximity, therefore we reject Markov chain states only based on the surface normal to mitigate variance. This rejection is implicitly enforced in the adaptive grid by hashing the normal states and discarding the states upon hash collision, as elaborated in the previous section. In the case of the static grid, where the normal is not part of the hash, we employ a threshold criterion involving the disparity between the surface normal and the mean direction of the vMF distribution derived from the Markov state. This combination of strategies enables the exchange of information among surfaces with normals corresponding to distinct cube indices.

**Algorithm 1:** Grid Resampling

```
S ← array of N_mc Markov chain states
s ← {0}                                   /* empty Markov chain state */
sum_mc ← 0
foreach i ← 0 to N_mc - 1 do
    S[i] ← grid.load(hit.x, hit.n)
    if hash grid collision then
        S[i].w̄ = 0
    sum_mc ← sum_mc + S[i].w̄
    ξ ← uniform random in [0, 1)
    if ξ < S[i].w̄ / sum_mc then
        s ← S[i]
```

*Step 2: Sampling and Tracing.* In order to facilitate the sharing of Markov chain states across grid vertices, the states do not store the vMF parameters directly. Instead, they retain sufficient statistics including a weighted mean of the positions of the light source $\overline{\boldsymbol{y}}$ and a weighted mean cosine of directions to the light source $\overline{r}$ from which the distribution parameters can be reconstructed in a way that is more robust compared to using a direction and a distance [Ruppert et al. 2020]. Furthermore, the number of samples $N$ that went into the state is used to stabilize the parameters by incorporating a prior of the mean cosine, which vanishes over time (see Algorithm 2, Algorithm 2). We used a prior mean cosine $r_{\text{p}} = 0$ with $N_{\text{p}} = 1/\|s.\overline{\boldsymbol{y}}/s.\overline{w} - \text{hit}.\boldsymbol{x}\|^2$, which facilitates faster convergence to small distant light sources. For infinite distant light sources, directions can still be used and the type can be encoded in the sign of the resampling weight. However, to reduce code divergence we used virtual points displaced in the direction of the light source and did not observe any difference since states are only exchanged locally. In instances where the resampling process fails to produce a valid state, specifically when $\overline{w} = 0$ for all states, the sampling procedure reverts to BSDF importance sampling. This happens frequently when exploring a new part of the scene or when a bright light source disappears (see Section 5). To detect new light sources, BSDF sampling is also used independently of the resampled state with low probability $p_{\text{bsdf}}$. Using the sampled direction, a ray is traced from the current surface point to acquire the next intersection $\text{hit}_{\text{next}}$. The sampling procedure along with the reconstruction of vMF parameters is outlined in Algorithm 2.

**Algorithm 2:** Sampling and Ray Casting

$\xi \leftarrow$ uniform random in $[0, 1)$
**if** $s.\overline{w} = 0$ /* invalid */ or $\xi < p_{\text{bsdf}}$ **then**
  $p(\boldsymbol{\omega}) \leftarrow f_{\text{s}}(\boldsymbol{\omega}_{\text{i}}, \text{hit}.\boldsymbol{n})$ or $|\text{hit}.\boldsymbol{n} \cdot \boldsymbol{\omega}|$
  $s \leftarrow \{0\}$ /* empty Markov chain state */
**else**
  $\boldsymbol{\mu} \leftarrow \text{normalize}(s.\overline{\boldsymbol{y}}/s.\overline{w} - \text{hit}.\boldsymbol{x})$
  $r \leftarrow \left(s.N^2 \cdot s.\overline{r}/s.\overline{w} + N_{\text{p}} \cdot r_{\text{p}}\right) / \left(s.N^2 + N_{\text{p}}\right)$
  $\kappa \leftarrow (3r - r^3)/(1 - r^2)$ /* [Banerjee et al. 2005, eq. 4.4] */
  $p(\boldsymbol{\omega}) \leftarrow p(\boldsymbol{\omega} \mid \boldsymbol{\mu}, \kappa)$ /* Equation (4) */
sample $\boldsymbol{\omega} \sim p(\boldsymbol{\omega})$
$\text{hit}_{\text{next}} \leftarrow$ trace ray $(\text{hit}.\boldsymbol{x}, \boldsymbol{\omega})$

*Step 3: Evaluating the Estimator.* The ensemble of hash grid, resampling process (Step 1) and the Markov chain transitions, described with more detail in step 4, result in an equilibrium distribution $p(t)$ of vMF lobe parameters $t = (\boldsymbol{\mu}, \kappa)$ and the probability density of our estimator is described by the continuous mixture

$$p(\boldsymbol{\omega}) = \int_{\mathcal{T}} p(t) p(\boldsymbol{\omega} \mid t) \, \mathrm{d}t \,. \tag{5}$$

To achieve independence from the equilibrium distribution without the need to estimate reciprocal probabilities, we utilize continuous multiple importance sampling [West et al. 2020]. Let $\overline{\boldsymbol{x}} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ be a surface path at pixel $j$ with measurement contribution function $f_j$ and $\text{S}_v$ the accessed Markov states during the resampling phase at a vertex $\boldsymbol{x}_v$, then the estimator for Equation (1) is

$$\left\langle I_j \right\rangle_{\text{SMIS}} = \frac{f_j(\overline{\boldsymbol{x}})}{\prod_{v=1}^{k} \left( \sum_{s \in \text{S}_{v-1}} p(\boldsymbol{x}_v \mid s) \right)} \,. \tag{6}$$

The estimator for the incident light which is evaluated for Markov chain updates is

$$\left\langle L_{\text{i}}(\text{hit}.\boldsymbol{x}, \boldsymbol{\omega}) \right\rangle_{\text{SMIS}} = \frac{L(\text{hit}_{\text{next}}, \text{-}\boldsymbol{\omega}) \cdot f_{\text{s}}(\text{hit}, \text{hit}_{\text{next}}) \cdot |\text{hit}.\boldsymbol{n} \cdot \boldsymbol{\omega}|}{\sum_{s \in \text{S}} p(\boldsymbol{\omega} \mid s)} \,. \tag{7}$$

Note that the estimator contains the term $L(\text{hit}_{\text{next}}, \text{-}\boldsymbol{\omega})$ which needs to be estimated by itself, due to the recursive nature of the render equation. To address this issue, we employ an irradiance cache, as explained in more detail in the following. In contrast to Metropolis-Hastings MCMC methods which require carefully selected acceptance probabilities to converge towards a specific equilibrium distribution [Hastings 1970], our approach provides flexibility in tuning various aspects of the algorithm for efficiency, temporal adaptation, or quality, depending on the scene and use case.

*Step 4: Markov Chain Update.* In the final stage of the path guiding procedure, the Markov state is transitioned and written on the hash grid, based on a MCMC acceptance test. Unlike conventional approaches that modify the existing state in memory, our approach calculates a new buffer position, analogous to the resampling procedure. Subsequently, we store the transitioned state at this new position, potentially overwriting another pre-existing state. Thereby, the stochastic interpolation when reading or writing allows information to spread between neighboring cells.

The acceptance probability which dictates whether a transition occurs is computed as

$$p_{\text{accept}} = \min\left(\frac{f_{\text{mc}}}{(sum_{\text{mc}}/N_{\text{mc}})}, 1\right), \tag{8}$$

where $f_{\text{mc}}$ is a luminance estimate of the incident light (Equation (7)). In common Markov chain methodologies the acceptance probability usually only depends on the current state, here we use the mean score of all states accessed during resampling, which improves the likelihood of acceptance in scenarios where the grid is empty, in particular during warm-up phases. Upon accepting a transition, the state is mutated using a maximum likelihood estimation step. For the first few samples, a luminance-weighted arithmetic average is computed, while for subsequent samples, we compute an exponential average, which allows to dynamically adjust to changes. The complete procedure is detailed in Algorithm 3.

**Algorithm 3:** State Transition and Maximum Likelihood Estimation

$f_{\text{mc}} \leftarrow \texttt{lum}(\langle L_{\text{i}}(\text{hit}.\boldsymbol{x}, \boldsymbol{\omega})\rangle_{SMIS})$
$\xi \leftarrow$ uniform random in $[0, 1)$
**if** $\xi < f_{\text{mc}}/(sum_{\text{mc}}/N_{\text{mc}})$ `/* MCMC acceptance test, Equation (8) */` **then**
  $s.N \leftarrow \min(s.N + 1, N_{\max})$
  $\alpha \leftarrow \max(1/s.N, \alpha_{\min})$  `/* blend factor, cf. Dittebrandt et al. [2023] */`
  $\boldsymbol{\mu} \leftarrow \texttt{normalize}(s.\overline{\boldsymbol{y}}/s.\overline{w} - \text{hit}.\boldsymbol{x})$
  $s.\overline{w} \leftarrow \texttt{mix}(s.\overline{w}, f_{\text{mc}}, \alpha)$
  $s.\overline{\boldsymbol{y}} \leftarrow \texttt{mix}(s.\overline{\boldsymbol{y}}, f_{\text{mc}} \cdot \text{hit}_{\text{next}}.\boldsymbol{x}, \alpha)$
  $s.\overline{r} \leftarrow \texttt{mix}(s.\overline{r}, f_{\text{mc}} \cdot \texttt{dot}(\texttt{normalize}(\text{hit}_{\text{next}}.\boldsymbol{x} - \text{hit}.\boldsymbol{x})), \boldsymbol{\mu}), \alpha)$
  $s.T \leftarrow T$  `/* current time, see Section 5 */`
  $s.\boldsymbol{v} \leftarrow (\text{hit}_{\text{next}}.\boldsymbol{x} - \text{hit}_{\text{next}}.\boldsymbol{x}_{\text{prev}})/(T - T_{\text{prev}})$
  grid.`store`$(s)$

In addition to maximum likelihood estimation, the grid resampling and acceptance steps form a stochastic process that controls the population of Markov chains in the grid, similar to a genetic algorithm. This optimizes the learned radiance distribution across states beyond what the maximum likelihood estimation of a single state is able to represent.

*Irradiance Caching.* A crucial detail, briefly mentioned in the preceding section, is that an estimate of incident light, including emitted and reflected light at $\text{hit}_{\text{next}}$, is required in Algorithm 3, Algorithm 3. However, due to the recursive nature of the render equation, the term $L(\text{hit}_{\text{next}}, \text{-}\boldsymbol{\omega})$ in the estimator (Equation (7)) is unknown when processing individual path vertices independently and would require storing and backtracking full paths. To address this issue, we employ an irradiance cache, which can be queried for an estimate of $L(\text{hit}_{\text{next}}, \text{-}\boldsymbol{\omega})$ when training the Markov chain. We used a Lambertian BSDF and assume that it successfully approximates the radiance transfer in the scene, while acknowledging that the path guiding may be suboptimal in scenarios where the path length is constrained, since the cache approximates infinite-bounce radiance transfer. Note that in our implementation, the cache is exclusively used for training and plays no role in the computation of the final render. Nevertheless, it could still be utilized for other purposes, like as a control variate or to obtain an estimate of the path tail [Müller et al. 2021]. Our irradiance cache is implemented using an adaptive hash grid, aligning closely with the the guiding datastructure. However, coarser levels are accessed only upon hash collisions. To ensure responsiveness to changing lighting conditions, we employ a small accumulation factor.

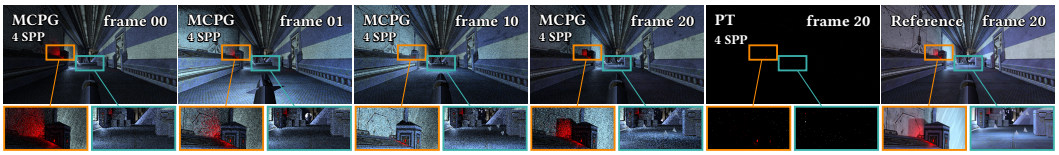


Fig. 4. Temporal adaption to a fast-moving rocket in a 20 frame animation. In frame 00 the rocket has not been fired, subsequent frames illustrate the tracking behavior. In frame 01 the rocket has already been captured; top-to-bottom workgroup execution leads to better a guiding distribution at the bottom of the frame, by frame 10, the light source information has distributed across all surfaces. Our algorithm remains unbiased even with dynamic geometry and light sources. Average frame time: 13.0 ms on an AMD Radeon RX 7900 XTX.

*Comparision to Dittebrandt et al..* In contrast to the screen-space path guiding method [Dittebrandt et al. 2023], our approach exhibits four notable differences. The screen space method uses a distinct score based on the last incident irradiance estimate, while we reuse the mean of luminance estimates. Our prior depends on the distance to the light source, showing a faster adaption to small, distant light sources. Furthermore, Dittebrandt et al. [2023] suggest incorporating additional knowledge, such as mean light source size; however, we observed inferior performance in this case. To decrease variance, their method obtains larger SMIS sets by reusing samples spatially and temporally. Markov chain states are shuffled between nearby pixels after every update to prevent correlation arising from dependent temporal sampling. In contrast, to simplify support for multi-bounce paths, our approach employs independent SMIS estimators for each sample. Lastly, we used an irradiance cache to estimate incident radiance, allowing for paths of arbitrary length without additional register memory.

*Extension to Single Scattering.* Our path guiding algorithm is versatile and requires only minimal modifications to the sampling procedure for rendering of single scattering volumes in real-time: In addition to generating initial vertices within the medium, 1) the BSDF is replaced by the phase function of the volume; this is possible without changing the state space, as the vMF distribution is inherently defined on a spherical domain. 2) we substitute the normal in the hash of the adaptive hash grid with the reverse camera direction, to better handle anisotropic phase functions that exhibit a dependence on the view direction. To share states between grid vertices with different incident directions and to avoid artifacts, we jitter the direction before hashing according to a cosine lobe. For the static grid, the normal threshold is simply removed. Furthermore, for guiding direct single scattered light only, the irradiance cache can be omitted. This modification is reflected in Equation (7) by replacing the emitted and reflected radiance $L$ with the emitted radiance $L_e$. Otherwise, if a bias is acceptable, the irradiance cache can be used to in-scatter from all surfaces into the volume. For caustics, sharing positions of light sources behind refractive interfaces is especially challenging because the angle of refraction depends on the incident direction. To mitigate the training error resulting from this effect, we apply a correction to the target point according to [Ruppert et al. 2020]. This displaces the target point along the incident direction, aligning it with the apparent distance rather than the actual distance.

*Synchronization.* In our light cache update, we explicitly synchronize access to memory regions to prevent the occurrence of invalid data. In contrast, explicit synchronization is not enforced on the Markov chain to enhance performance by eliminating the latency associated with locking and unlocking. We operate the Markov chain under the assumption that data corruptions follow a certain distribution and that inflicted perturbations are part of the equilibrium of the Markov chain. A precise knowledge of the distribution is not required as it cancels out in the SMIS estimator.

## 5 Improvements for Dynamic Content

In the following, we discuss strategies to improve handling dynamic content to ensure effective sampling and tracking, even with occlusions and high-speed movements.

*Discarding Light Sources.* In dynamic game scenes, dominant light sources often disappear. For instance, particles may vanish, the sun might be occluded by a moving object, or a muzzle flash may disappear. This presents a challenge for the current algorithm, where the resampling step continues to designate the old bright light source as the sampling target, resulting in poor sampling behavior and increased noise. Furthermore, older states are unlikely to be replaced, as the substantial contribution of the outdated light source reduces the acceptance probability of other light sources. A straightforward approach involves retracing rays to ensure the Markov state's ongoing validity. However, this strategy would reduce the sampling budget for every frame and increase memory consumption, particularly since multi-bounce rays might need to be stored alongside the vMF distribution parameters. Instead, we employ a heuristic to discard states: When the new luminance estimate (Algorithm 3, Algorithm 3) is significantly smaller than the current luminance estimate $\overline{w}$, we set the weight of the current state to zero. To prevent premature discarding while the guiding distribution has yet to converge, we impose two conditions: first, the stored time point $T$ must differ from the current time, indicating that the light source has not been observed at the current moment. Second, the condition $\boldsymbol{\mu}^T\boldsymbol{\omega} > r$ (refer to Algorithm 2) must be satisfied, ensuring that the outgoing ray direction is sufficiently aligned with the mean target direction; this suggests that the light source would have been detected if it were still present. Consequently, $sum_{\text{mc}}$ is reduced in subsequent samples, increasing the acceptance probability of other states and leading to more intensified non-guided sampling if no valid state is discovered during the resampling step, thereby facilitating the discovery of new light sources. This also encourages the algorithm to learn distributions that do not accommodate multiple spatially independent features in a single state.

*Dynamic Geometry.* If the geometry has moved across multiple grid vertices, the resampling process might not yield a valid Markov chain state, resulting in observable flickering. To overcome this issue and to prevent relearning the guiding distribution, we use the previous world space position $\mathsf{hit}.\boldsymbol{x}_{\text{prev}}$ when loading states in Algorithm 1, Algorithm 1 during the first sample of a frame. This transfers Markov chain states of important light sources reliably to the new grid position. In most cases, geometry movement is limited such that learned incident light distribution remains valid between two frames. When this assumption does not hold, the acceptance test ensures that Markov chain states are only stored in the new grid position if they still contribute to the incident light estimate.

*Dynamic Light Sources.* Thanks to the maximum-likelihood step, the Markov chain guiding method is inherently capable of tracking moving light sources, especially if they are sampled in every frame. To enable the tracking of even very small and fast-moving light sources, we attempt to predict the location where the light source will appear in the current frame. For that, every time a Markov chain state is updated, we store the current velocity of the light source

$$\boldsymbol{v} = (\mathsf{hit}_{\text{next}}.\boldsymbol{x} - \mathsf{hit}_{\text{next}}.\boldsymbol{x}_{\text{prev}})/(T - T_{\text{prev}})\,, \tag{9}$$

which we estimate from the current $\mathsf{hit}_{\text{next}}.\boldsymbol{x}$ and previous $\mathsf{hit}_{\text{next}}.\boldsymbol{x}_{\text{prev}}$ world space position, as well as the current $T$ and previous $T_{\text{prev}}$ time. This way, the mean target position can be updated when the state is loaded in Algorithm 1, Algorithm 1 as

$$\mathrm{S}[i].\overline{\boldsymbol{y}} = \mathrm{S}[i].\overline{\boldsymbol{y}} + \mathrm{S}[i].\overline{w} \cdot (T - \mathrm{S}[i].T) \cdot \mathrm{S}[i].\boldsymbol{v}\,. \tag{10}$$

The validity of this approach relies on the assumption that the velocity remains constant between two frames. However, we have observed effective tracking even for small, rapidly accelerated light sources, such as glowing particles affected by gravity, which we attribute to our update mechanism and the cooperative nature of our algorithm. Figure 4 shows renders of a 20-frame animation of a rapidly moving rocket that is reliably tracked by our algorithm.

## 6 Results

We implemented our method in our custom game engine for the original Quake game, which will be released alongside this work as Merian-Quake[1], including source code. Performance was measured on an AMD Radeon RX 7900 XTX graphics card. Frame times describe the full pipeline cost, including G-Buffer creation, acceleration structure builds for dynamic geometry, and denoising for images annotated with SVGF [Schied et al. 2017]. We report the root mean squared error (RMSE) and ꟻLIP [Andersson et al. 2020] compared to references with a high sample count.

Please refer to the videos in the supplementary material for animations. For reproducibility, we rendered at a constant rate of 60 frames per second (FPS). In most scenes, the actual frame times were lower, ensuring that the recordings accurately depict the learning rate and tracking behavior.

For still images, we allow the guiding distribution in our method to converge, which on average requires approximately 100 frames; refer to Figure 5. For ReSTIR, we present results after a 20-frame warm-up phase, thus avoiding the artificial error reduction resulting from temporal reuse; this is in accordance with the evaluation setting described in [Bitterli et al. 2020]. We use a resampling set size $N_{\text{mc}}$ of 5 for all scenes except for Pool, where we use 15. All methods use BSDF or phase function importance sampling to discover light sources, i.e., no next event estimation was used and no explicit light information is required. We evaluated our method in a range of scenes from the Arcane Dimensions[2], Alkaline[3], and The Immortal Lock[4] mods as well as a custom pool scene with animated water surface normals for underwater caustics. For single scattering, we sampled the transmittance from the camera.

Figure 5 shows the RMSE of the accumulated images for three of our scenes. We compare our estimator with and without the warm-up phase against path tracing with BSDF importance sampling and ReSTIR DI for direct light. In Tears and Azad, our algorithm is competitive with ReSTIR DI even at low sample counts. ReSTIR suffered from outlier samples and correlation artifacts, which slowed down convergence. In many light scenarios, such as Alkaline, our algorithm performs worse, due to the limited size of the SMIS set. Note that our algorithm remains unbiased, even if the guiding distribution has not yet converged.

Figure 6 shows a comparison of rendering a blend of direct and indirect light in real-time with our technique. In both scenes, our method consistently produces temporally stable outputs without any correlation artifacts resulting from Markov chain state sharing, which can be effectively denoised using SVGF [Schied et al. 2017]. We observe effective product sampling of moderately glossy reflections as a result of the integration of the BSDF into the resampling weight (Equation (7)). This is demonstrated in the reflections of the candles in the background of Azad.

Figure 7 shows rendering of single scattering using our method. The left side compares two-bounce indirect lighting combined with single scattering to path tracing with BSDF and phase function importance sampling, at real-time frame rates. The right side shows a rendering of underwater caustics with a larger 8192 sample budget, our estimator reaches the equal error with phase function importance sampling after only 32 samples.

[1] https://github.com/LDAP/merian-quake
[2] https://www.moddb.com/mods/arcane-dimensions
[3] https://alkalinequake.wordpress.com/
[4] https://www.moddb.com/mods/the-immortal-lock

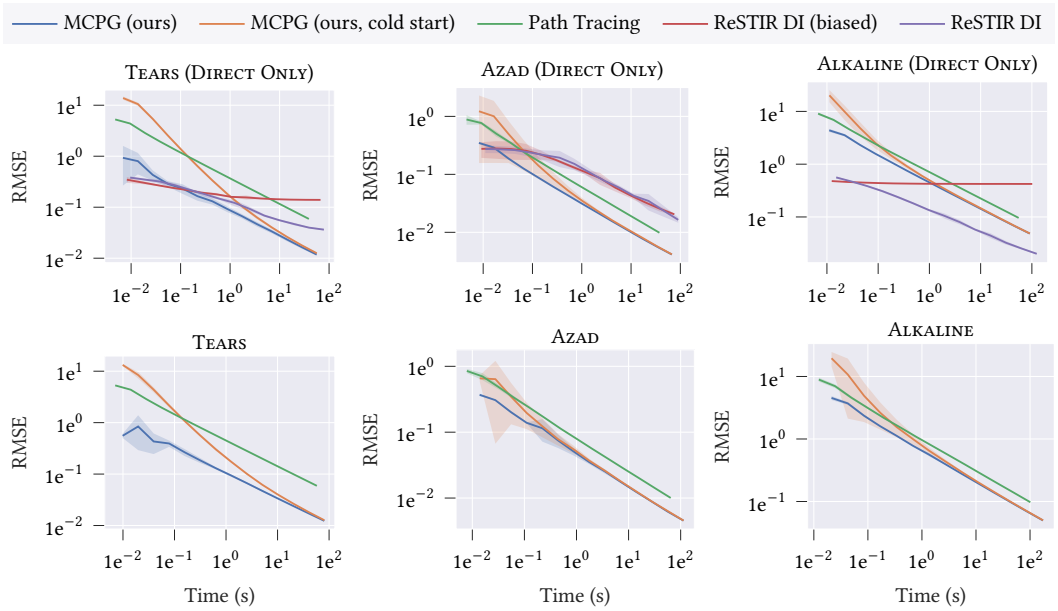


Fig. 5. Accumulation error (root mean squared error) for three of our scenes. We compare our estimator with (blue) and without (orange) warm-up phase against path tracing with BSDF importance sampling and ReSTIR DI. ReSTIR DI can only be used for estimating direct light (top row). The bottom row shows errors for two-bounce indirect lighting. Our guiding distribution usually converges within 100 frames. In Tears and Azad, ReSTIR DI suffered from outlier samples and correlation artifacts, slowing down convergence.

Figure 4 presents a 20-frame animation that includes both a rapidly moving dynamic light source (rocket) that must be tracked by the guiding algorithm and dynamic geometry (particles) between which information about the light source must be shared. In frame 00, the rocket is not yet launched; in subsequent frames the tracking behavior is demonstrated: In frame 01, the rocket has already been successfully captured, and in frame 10, the information has been effectively shared across all surfaces. The algorithm remains unbiased despite the presence of dynamic geometry and light sources. Frame 01 also demonstrates sharing of Markov chain states across workgroup (pixel block) executions in our algorithm: Since workgroups on our GPU are scheduled from top to bottom, we can observe an improved guiding distribution in the lower portion of the frame.

Figure 8 presents equal-time renders, evaluating our method applied to direct light against the screen-space technique of Dittebrandt et al. [2023] and ReSTIR DI using unbiased MIS [Bitterli et al. 2020]. For this experiment, the irradiance cache in our algorithm was deactivated. Our method demonstrates competitive performance in both execution time and approximation error in environments with simple guiding distributions, as shown in Tears (left). Extensive implementation effort and limiting temporal influence were necessary to achieve stable and unbiased results with ReSTIR. Our method remains unbiased even in fully dynamic scenes, using only a single ray per pixel, while ReSTIR requires an additional ray for each reuse pass. Our method is less effective in scenarios with many light sources, as illustrated in Alkaline (right). In these situations, it adapts its sampling distribution to prioritize dominant light sources and maintains a stable output over time. For a detailed examination of this behavior, see Section 7. ReSTIR remains orthogonal to our method and MCPG could be used to guide paths in ReSTIR PT [Lin et al. 2022] for even lower error

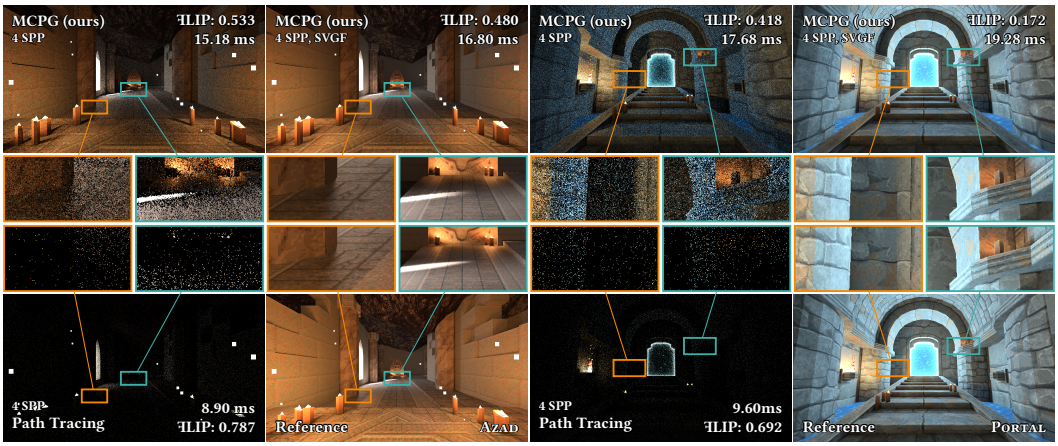


Fig. 6. Two scenes illuminated by a mix of direct and indirect light, rendered in real-time with our estimator at 1920 × 1080, with and without SVGF denoising [Schied et al. 2017]. Left: AZAD is primarily lit by small particles, candles and indirect light from the window border. Right: PORTAL is primarily lit by indirect light coming from the portal, the portal itself is lit by a small strip of light around its borders.

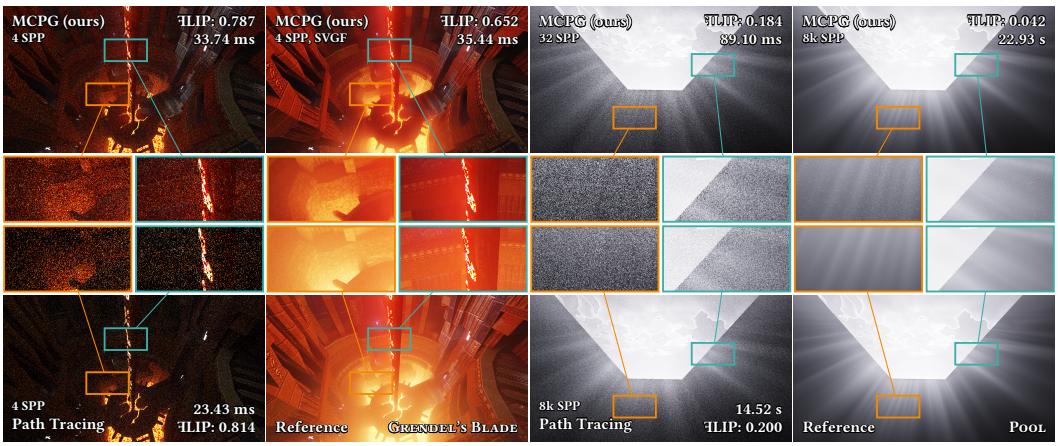


Fig. 7. Rendering of single scattering using our estimator. Left: Rendering of a game scene from the Arcane Dimensions Quake mod real-time at 1920 × 1080, with and without SVGF denoising [Schied et al. 2017]. Right: Rendering of underwater caustics with a larger sampling budget of 8k samples, our estimator reaches equal error with phase function importance sampling after only 32 samples.

output. Dittebrandt et al. [2023] suffers from high register usage for CMIS and correlation between temporal samples, reducing performance and sample quality, respectively.

Figure 9 (left) shows the impact of varying the resolutions of the adaptive hash grid and the corresponding buffer sizes. The resolution is specified in degrees relative to the camera. Higher resolution improves sampling around more detailed geometric structures. Increasing the resolution of the grid beyond the level of detail in the scene does not improve quality. Finer resolutions require larger buffer sizes to avoid frequent hash collisions. In Quake, we found that good quality is achieved in most scenes with a grid resolution of $0.6^\circ$ and a buffer size of $10^6$, resulting in a memory usage of 44 MB. This assessment is based on our encoding for Markov chain states, which requires

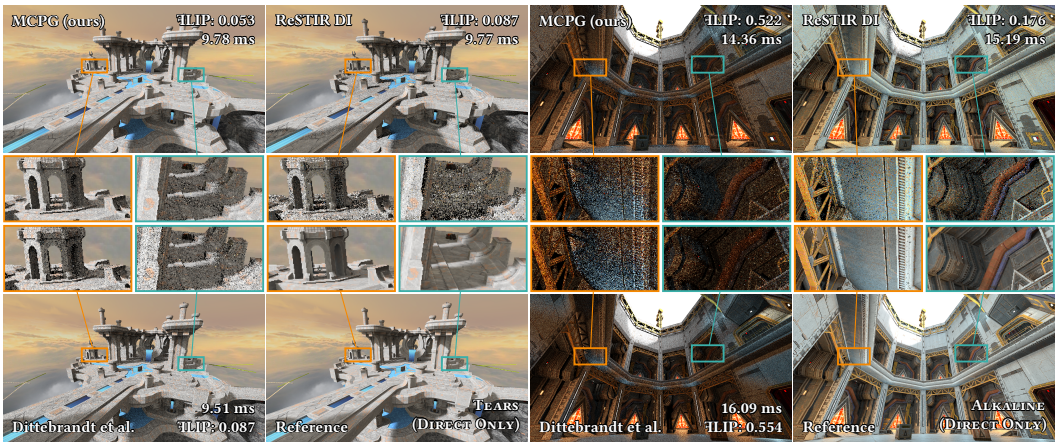


Fig. 8. Roughly equal-time renders, comparing the effectiveness of direct light rendering with the the screen-space technique of Dittebrandt et al. [2023] and the unbiased MIS variant of ReSTIR DI [Bitterli et al. 2020] using one temporal and four spatial reuse passes. Our method demonstrates competitive performance in terms of both execution time and approximation error in environments characterized by simple guiding distributions (left). However, it exhibits limitations in scenarios with many light sources (right) compared to ReSTIR. Nonetheless, our method is capable of adapting its distribution to prioritize the most significant light sources, thereby delivering a temporally stable output.

44 Bytes. For scenes with single scattering, a larger hash buffer of approximately 500 MB is required. Gameplay is enhanced by using larger hash buffers, as this avoids the need to re-learn the guiding distributions upon revisiting a scene. Regarding the static grid, no differences were measured with hash buffers exceeding 10 MB. However, a larger cell size facilitates faster information spread.

In practice, we found 2 SPP for 2-bounce surface paths and 2 SPP for volume paths (7 rays in total: 1 for G-Buffer creation, $2 \times 2$ surface rays, 2 volume rays) an optimal compromise between performance and quality, resulting in around 60 FPS in most scenes of Arcane Dimensions. For older hardware, the surfaces can also be traced with only 1 SPP or, if a slight bias is acceptable, the path length can even be reduced to trace direct light only, and the path tail is read from the irradiance cache. Although this method is biased, it results in more than 120 FPS in most scenes, even on previous generation hardware, while achieving a similar error at low sample counts.

## 7 Limitations

*Proximity Bias and Frequency Limitations.* We observe increased variance at locations where states are shared under varying lighting conditions. This phenomenon occurs when a neighboring state with seemingly high contribution is utilized, yet, from the new position, the light source becomes occluded. In contrast to screen-space approaches, our hash grid is inherently designed to ensure proximity, and information is exchanged exclusively with neighboring vertices. In Quake, the incorporation of the surface normal into the hash proves to be effective in alleviating variance across geometry edges. Proximity bias remains an issue at shadow boundaries and surfaces with high frequency incident lighting, as can be seen in Azad (Figure 6, left). Note that this biases the state space only, the final SMIS estimator remains unbiased. The limiting frequency of incident lighing thereby depends on the hash grid resolution, not scene geometry. However, no artifacts or issues due to increased variance were observed in conjunction with our implementation of SVGF, and contact shadows remained sharp after denoising (see for example Figure 3, right).

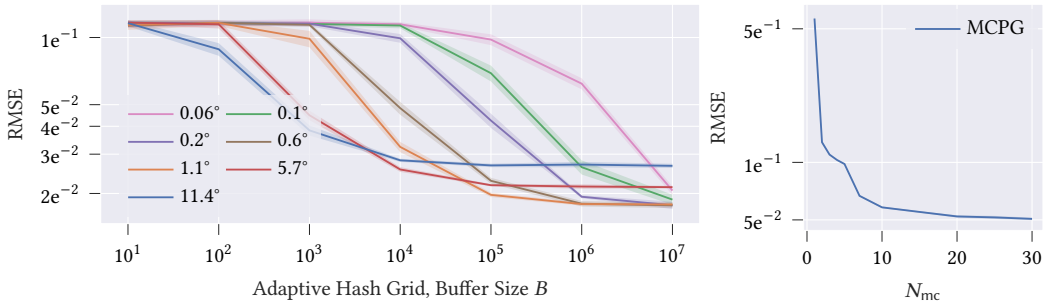


Fig. 9. Influence of algorithm parameters on the approximation error (RMSE). Left: Exploring the impact of different hash grid resolutions and buffer sizes in Azad. The adaptive grid has its target resolution set in degrees from the camera, finer grids achieve lower error but require larger hash buffers to mitigate hash collisions. Right: Influence of resampling set sizes $N_{mc}$ in Alkaline, results reveal diminishing returns attributed to the limited states accessible from a world space point, complex scenes with many light sources benefit most from larger resampling sets.

*SMIS in Many Light Scenarios.* To effectively control outliers and variance, the mixture model must acurately represent the incident light distribution and every visible light source must be included in the SMIS set. In our case, using a relatively small set of $N_{mc} = 5$, we observe overall darkening and energy concentrating in outlier samples due to the unbiased nature of our estimator. This can be seen in Alkaline (Figure 8, right) or at the pillar in Azad (Figure 6, left). Although a spatial MIS scheme as in [Dittebrandt et al. 2023] could enhance the situation for the first vertex, a more sophisticated approach is required to perform MIS on subsequent path vertices. As illustrated in Figure 9 (right), an increase in the sample set enhances error reduction; however, diminishing returns become evident with larger sets. Apart from a limited number of light sources in the scene, this limitation arises from the finite number of states accessible from a given world space position. In simpler scenes like Tears (Figure 8, left), where a single light source predominantly illuminates the scene, minimal darkening is observable. The size of the SMIS set requires careful consideration to strike a balance between quality and runtime performance, given the additional cost of more hash grid buffer accesses. In practice, when focusing on the most important light sources suffices, acceptable image quality can be obtained using an SMIS set as small as 3 when coupled with a firefly filter and a denoiser. This combination proves to be effective in eliminating residual noise, and compensation for any resulting darkening is achieved through an elevated exposure. For single scattering, we found that a set size of 5 is an acceptable compromise.

## 8 Conclusion

We introduced a lightweight and unbiased path guiding algorithm designed for real-time applications with highly dynamic content and showcased its effectiveness in guiding both direct and indirect illumination. It seamlessly extends to guide single scattering events in participating media while maintaining real-time performance. Compared to ReSTIR, our method requires less implementation effort, extends naturally to longer paths, remains unbiased even in fully dynamic scenes, and only falls short in scenes with many light sources. However, it is capable of adapting its distribution to prioritize the most significant light sources, thereby delivering a temporally stable output. We implemented our method as a custom game engine for the original Quake game, demonstrating path traced indirect lighting and single scattering in real-time.

## References


Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020. FLIP: A Difference Evaluator for Alternating Images. *Proc. ACM Comput. Graph. Interact. Tech.* 3, 2, Article 15 (2020), 23 pages. https://doi.org/10.1145/3406183

Steve Bako, Mark Meyer, Tony DeRose, and Pradeep Sen. 2019. Offline Deep Importance Sampling for Monte Carlo Path Tracing. *Computer Graphics Forum* 38, 7 (2019), 527–542. https://doi.org/10.1111/cgf.13858

Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. 2005. Clustering on the Unit Hypersphere using von Mises-Fisher Distributions. *Journal of Machine Learning Research* 6, 46 (2005), 1345–1382. http://jmlr.org/papers/v6/banerjee05a.html

Thomas Bashford-Rogers, Kurt Debattista, and Alan Chalmers. 2012. A Significance Cache for Accelerating Global Illumination. *Comput. Graph. Forum* 31, 6 (2012), 1837–1851. https://doi.org/10.1111/j.1467-8659.2012.02099.x

Nikolaus Binder, Sascha Fricke, and Alexander Keller. 2018. Fast Path Space Filtering by Jittered Spatial Hashing. In *ACM SIGGRAPH 2018 Talks (SIGGRAPH '18)*. Association for Computing Machinery, New York, NY, USA, Article 71, 2 pages. https://doi.org/10.1145/3214745.3214806

Benedikt Bitterli, Chris Wyman, Matt Pharr, Peter Shirley, Aaron Lefohn, and Wojciech Jarosz. 2020. Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Trans. Graph.* 39, 4 (2020). https://doi.org/10/gg8xc7

Guillaume Boissé. 2021. World-space spatiotemporal reservoir reuse for ray-traced global illumination. In *SIGGRAPH Asia 2021 Technical Communications (SA '21 Technical Communications)*. Association for Computing Machinery, New York, NY, USA, Article 22, 4 pages. https://doi.org/10.1145/3478512.3488613

Jakub Boksansky, Paula Jukarainen, and Chris Wyman. 2021. Rendering Many Lights with Grid-Based Reservoirs. In *Ray Tracing Gems II*, Adam Marrs, Peter Shirley, and Ingo Wald (Eds.). APress, 351–365. https://doi.org/10.1007/978-1-4842-7185-8_23

M. T. Chao. 1982. A General Purpose Unequal Probability Sampling Plan. *Biometrika* 69, 3 (1982), 653–656. https://doi.org/10.2307/2336002

Ken Dahm and Alexander Keller. 2017. Learning Light Transport the Reinforced Way. In *ACM SIGGRAPH 2017 Talks (SIGGRAPH '17)*. Association for Computing Machinery, New York, NY, USA, Article 73, 2 pages. https://doi.org/10.1145/3084363.3085032

Addis Dittebrandt, Vincent Schüß ler, Johannes Hanika, Sebastian Herholz, and Carsten Dachsbacher. 2023. Markov Chain Mixture Models for Real-Time Direct Illumination. *Computer Graphics Forum* (2023). https://doi.org/10.1111/cgf.14881

Ana Dodik, Marios Papas, Cengiz Öztireli, and Thomas Müller. 2022. Path Guiding Using Spatio-Directional Mixture Models. *Computer Graphics Forum* 41, 1 (2022), 172–189. https://doi.org/10.1111/cgf.14428

Honghao Dong, Guoping Wang, and Sheng Li. 2023. Neural Parametric Mixtures for Path Guiding. In *ACM SIGGRAPH 2023 Conference Proceedings (SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 29, 10 pages. https://doi.org/10.1145/3588432.3591533

Marc Droske, Johannes Hanika, Jiří Vorba, Andrea Weidlich, and Manuele Sabbadin. 2023. Path Tracing in Production: The Path of Water. In *ACM SIGGRAPH 2023 Courses (SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 12, 66 pages. https://doi.org/10.1145/3587423.3595519

W. K. Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57, 1 (1970), 97–109. https://doi.org/10.1093/biomet/57.1.97

Sebastian Herholz, Oskar Elek, Jiří Vorba, Hendrik Lensch, and Jaroslav Křivánek. 2016. Product Importance Sampling for Light Transport Path Guiding. *Computer Graphics Forum* 35, 4 (2016), 67–77. https://doi.org/10.1111/cgf.12950

Heinrich Hey and Werner Purgathofer. 2002. Importance Sampling with Hemispherical Particle Footprints. In *Proceedings of the 18th Spring Conference on Computer Graphics (SCCG '02)*. Association for Computing Machinery, New York, NY, USA, 107–114. https://doi.org/10.1145/584458.584476

Wenzel Jakob. 2012. Numerically stable sampling of the von Mises Fisher distribution on $S^2$ (and other tricks). (2012). http://infoscience.epfl.ch/record/220041

Henrik Wann Jensen. 1995. Importance Driven Path Tracing using the Photon Map. In *Rendering Techniques '95*, Patrick M. Hanrahan and Werner Purgathofer (Eds.). Springer Vienna, Vienna, 326–335.

James T. Kajiya. 1986. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques (SIGGRAPH '86)*. ACM, New York, NY, USA, 143–150. https://doi.org/10.1145/15922.15902

Eric P. Lafortune and Yves D. Willems. 1995. A 5D Tree to Reduce the Variance of Monte Carlo Ray Tracing. In *Rendering Techniques '95*, Patrick M. Hanrahan and Werner Purgathofer (Eds.). Springer Vienna, Vienna, 11–20.

Vincent Schüßler, Johannes Hanika, Alisa Jung, and Carsten Dachsbacher. 2022. Path Guiding with Vertex Triplet Distributions. *Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering)* 41, 4 (2022). https://doi.org/10.1111/cgf.14582

Daqi Lin, Markus Kettunen, Benedikt Bitterli, Jacopo Pantaleoni, Cem Yuksel, and Chris Wyman. 2022. Generalized Resampled Importance Sampling: Foundations of ReSTIR. *ACM Trans. Graph.* 41, 4, Article 75 (2022), 23 pages. https://doi.org/10.1145/3528223.3530158

Daqi Lin, Chris Wyman, and Cem Yuksel. 2021. Fast Volume Rendering with Spatiotemporal Reservoir Resampling. *ACM Trans. Graph.* 40, 6, Article 279 (2021), 18 pages. https://doi.org/10.1145/3478513.3480499

Haolin Lu, Wesley Chang, Trevor Hedstrom, and Tzu-Mao Li. 2024. Real-Time Path Guiding Using Bounding Voxel Sampling. *ACM Trans. Graph.* 43, 4, Article 125 (2024), 14 pages. https://doi.org/10.1145/3658203

Zander Majercik, Thomas Mueller, Alexander Keller, Derek Nowrouzezahrai, and Morgan McGuire. 2021. Dynamic Diffuse Global Illumination Resampling. In *ACM SIGGRAPH 2021 Talks (SIGGRAPH '21)*. Association for Computing Machinery, New York, NY, USA, Article 24, 2 pages. https://doi.org/10.1145/3450623.3464635

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.* 41, 4, Article 102 (2022), 15 pages. https://doi.org/10.1145/3528223.3530127

Thomas Müller, Markus Gross, and Jan Novák. 2017. Practical Path Guiding for Efficient Light-Transport Simulation. *Computer Graphics Forum (Proceedings of EGSR)* 36, 4 (2017), 91–100. https://doi.org/10.1111/cgf.13227

Thomas Müller, Brian McWilliams, Fabrice Rousselle, Markus Gross, and Jan Novák. 2019. Neural Importance Sampling. *ACM Trans. Graph.* 38, 5, Article 145 (2019), 19 pages. https://doi.org/10.1145/3341156

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time neural radiance caching for path tracing. *ACM Trans. Graph.* 40, 4, Article 36 (2021), 16 pages. https://doi.org/10.1145/3450626.3459812

Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. 2018. Monte Carlo Methods for Volumetric Light Transport Simulation. *Computer Graphics Forum (Eurographics State of the Art Reports)* 37, 2 (2018), 1–26. https://doi.org/10.1111/cgf.13383

NVIDIA. 2018. NVIDIA Turing GPU Architecture. (2018).

Y. Ouyang, S. Liu, M. Kettunen, M. Pharr, and J. Pantaleoni. 2021. ReSTIR GI: Path Resampling for Real-Time Path Tracing. *Computer Graphics Forum* 40, 8 (2021), 17–29. https://doi.org/10.1111/cgf.14378

Alexander Rath, Pascal Grittmann, Sebastian Herholz, Petr V évoda, Philipp Slusallek, and Jaroslav Křivánek. 2020. Variance-Aware Path Guiding. *ACM Trans. Graph.* 39, 4, Article 151 (2020), 12 pages. https://doi.org/10.1145/3386569.3392441

Florian Reibold, Johannes Hanika, Alisa Jung, and Carsten Dachsbacher. 2018. Selective Guided Sampling with Complete Light Transport Paths. *ACM Trans. Graph.* 37, 6, Article 223 (2018), 14 pages. https://doi.org/10.1145/3272127.3275030

Lukas Ruppert, Sebastian Herholz, and Hendrik P. A. Lensch. 2020. Robust Fitting of Parallax-Aware Mixtures for Path Guiding. *ACM Trans. Graph.* 39, 4, Article 147 (2020), 15 pages. https://doi.org/10.1145/3386569.3392421

Rohan Sawhney, Daqi Lin, Markus Kettunen, Benedikt Bitterli, Ravi Ramamoorthi, Chris Wyman, and Matt Pharr. 2023. Decorrelating ReSTIR Samplers via MCMC Mutations. *ACM Trans. Graph.* (2023). https://doi.org/10.1145/3629166 Just Accepted.

Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R. Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvi. 2017. Spatiotemporal Variance-Guided Filtering: Real-Time Reconstruction for Path-Traced Global Illumination. In *Proceedings of High Performance Graphics (HPG '17)*. Association for Computing Machinery, New York, NY, USA, Article 2, 12 pages. https://doi.org/10.1145/3105762.3105770

Justin Talbot, David Cline, and Parris Egbert. 2005. Importance Resampling for Global Illumination. In *Eurographics Symposium on Rendering (2005)*. The Eurographics Association. https://doi.org/10.2312/EGWR/EGSR05/139-146

Yusuke Tokuyoshi. 2025. *A Numerically Stable Implementation of the von Mises–Fisher Distribution on $S^2$*. Technical Report 24-12-5053. AMD.

Jiří Vorba, Johannes Hanika, Sebastian Herholz, Thomas Müller, Jaroslav Křivánek, and Alexander Keller. 2019. Path Guiding in Production. In *ACM SIGGRAPH 2019 Courses (SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 18, 77 pages. https://doi.org/10.1145/3305366.3328091

Jiří Vorba, Ondřej Karlík, Martin Šik, Tobias Ritschel, and Jaroslav Křivánek. 2014. On-Line Learning of Parametric Mixture Models for Light Transport Simulation. *ACM Trans. Graph.* 33, 4, Article 101 (2014), 11 pages. https://doi.org/10.1145/2601097.2601203

Rex West, Iliyan Georgiev, Adrien Gruson, and Toshiya Hachisuka. 2020. Continuous Multiple Importance Sampling. *ACM Trans. Graph.* 39, 4, Article 136 (2020), 12 pages. https://doi.org/10.1145/3386569.3392436

Quan Zheng and Matthias Zwicker. 2019. Learning to Importance Sample in Primary Sample Space. *Computer Graphics Forum* 38, 2 (2019), 169–179. https://doi.org/10.1111/cgf.13628