ReSTIR PT Enhanced: Algorithmic Advances for Faster and More Robust ReSTIR Path Tracing

DAQI LIN, NVIDIA, USA
MARKUS KETTUNEN, NVIDIA, Finland
CHRIS WYMAN, NVIDIA, USA

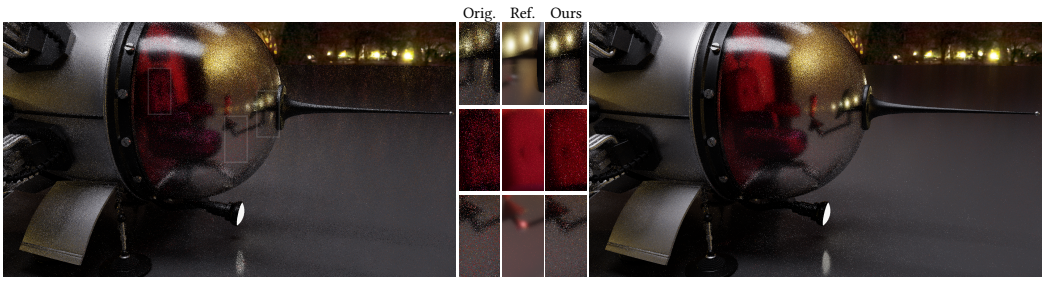


Fig. 1. *We enhance the performance and robustness of ReSTIR PT [Lin et al. 2022] with a set of novel algorithms. Our techniques provide a significant speedup, reduce correlation artifacts (first inset row), reduce color noise (third inset row), and improve the quality of glossy reflections (e.g., on the ground plane) and refractions (second inset row), as seen in this Spaceship scene.*

Algorithms leveraging ReSTIR-style spatiotemporal reuse have recently proliferated, hugely increasing effective sample count for light transport in real-time ray and path tracers. Many papers have explored novel theoretical improvements, but algorithmic improvements and engineering insights toward optimal implementation have largely been neglected. We demonstrate enhancements to ReSTIR PT that make it 2–3× faster, decrease both visual and numerical error, and improve its robustness, making it closer to production-ready. We halve the spatial reuse cost by reciprocal neighbor selection, robustify shift mappings with new footprint-based reconnection criteria, and reduce spatiotemporal correlation with duplication maps. We further improve both performance and quality by extensive optimization, unifying direct and global illumination into the same reservoirs, and utilizing existing techniques for color noise and disocclusion noise reduction.

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

Additional Key Words and Phrases: Optimization, Sampling, Light Transport, Real Time

ACM Reference Format:
Daqi Lin, Markus Kettunen, and Chris Wyman. 2026. ReSTIR PT Enhanced: Algorithmic Advances for Faster and More Robust ReSTIR Path Tracing. *Proc. ACM Comput. Graph. Interact. Tech.* 9, 1, Article 13 (May 2026), 19 pages. https://doi.org/10.1145/3804494

Authors' addresses: Daqi Lin, daqil@nvidia.com, NVIDIA, USA; Markus Kettunen, mkettunen@nvidia.com, NVIDIA, Finland; Chris Wyman, chris.wyman@acm.org, NVIDIA, USA.

ACM 2577-6193/2026/5-ART13
https://doi.org/10.1145/3804494

1 INTRODUCTION

ReSTIR Path Tracing [Lin et al. 2022] greatly improves path quality by spatiotemporally reusing path samples. But their prototype has high compute and memory overheads, yielding sluggish performance even on high-end GPUs (originally 40-80 ms per frame on an RTX 3090).

As published, ReSTIR PT exhibits robustness problems: it suffers spatiotemporal correlations, variance near disocclusions, and color noise. The hybrid shift, which enables ReSTIR PT to handle specular light transport, relies on simple roughness and distance thresholds to determine reuse strategy; these thresholds often benefit from adjustment per scene. While further research proposes partial solutions [Kettunen et al. 2023; Sawhney et al. 2024; Zhang et al. 2024], they do not fix the poor performance. In this paper, we propose techniques that greatly optimize ReSTIR PT's performance and improve its robustness and quality, reducing the gap to productization.

Our specific contributions include:

- Halving shift mapping costs in spatial reuse by reciprocal neighbor selection* (Section 3);
- New ray footprint thresholds that adapt to scene and materials* (Section 4);
- Reducing correlation artifacts by sample duplication maps* (Section 5);
- Improving quality and cost by unifying ReSTIR for direct and indirect light (Section 6); and
- Other optimizations that boost performance and improve robustness by reducing color and disocclusion noise. (Section 6)

2 BACKGROUND

Recent advances in hardware and denoising, like SVGF [Schied et al. 2017], NVIDIA NRD [NVIDIA 2020], and DLSS Ray Reconstruction [NVIDIA 2025], have made real-time ray-traced global illumination (GI) practical. Many real-time solutions today rely on radiosity-like recursive gathering with discretization, caching, and interpolation (e.g., Lumen [Wright et al. 2022], DDGI [Majercik et al. 2019], and Surfel GI [Halen et al. 2021]); this limits accuracy and responsiveness. Brute force path tracing is accurate but slow, requiring many samples for a denoisable image.

More recently, spatiotemporal reservoir resampling (ReSTIR) [Bitterli et al. 2020] aggressively reuses Monte Carlo samples from other pixels and frames in an unbiased way. When applied to path samples, ReSTIR amortizes the cost of tracing paths over space and time. This preserves path tracing's accuracy but delivers tens to hundreds more samples at the same cost. ReSTIR DI [Bitterli et al. 2020] enables real-time, many-light rendering and ReSTIR GI [Ouyang et al. 2021] targets diffuse/mid-glossy surfaces. Our work speeds and improves robustness of ReSTIR PT [Lin et al. 2022], improving the unbiased rendering of general paths, including multi-bounce specular reflection and caustics. Below, we review the resampling theory behind these ReSTIR algorithms.

2.1 RIS and GRIS

Resampled importance sampling (RIS) [Talbot et al. 2005] aims to sample from a low-variance target PDF $\bar{p} \propto \hat{p} \approx f$ to estimate an integral $\int_\Omega f(x)\,\mathrm{d}x$. As $\bar{p}$ is often difficult to sample directly, RIS draws M i.i.d. candidates $X_i \sim p$ (a suboptimal source PDF), selecting one Y proportional to resampling weights $w_i = (1/M)\hat{p}(X_i)/p(X_i)$. As $M \to \infty$, Y's distribution converges to $\bar{p} = \hat{p}/\int_\Omega \hat{p}(x)\,\mathrm{d}x$. Talbot et al. [2005] construct $\hat{p}$ by dropping the visibility term in direct illumination to improve sampling efficiency. The expression $f(Y)W_Y$ unbiasedly estimates integral $\int_\Omega f(x)\,\mathrm{d}x$, where $W_Y = (1/\hat{p}(Y))\sum_{i=1}^{M} w_i$ estimates $1/p_Y(Y)$, the reciprocal of Y's unknown PDF.

Generalized resampled importance sampling (GRIS) [Lin et al. 2022] allows random variables like Y to be used as candidates in another resampling pass. This chained resampling forms the

*Asterisked contributions apply to other spatiotemporal reuse algorithms, e.g., Bauszat et al. [2017]; Ouyang et al. [2021]

basis of ReSTIR. In GRIS, candidate samples can have different distributions, come from different domains Ω_i, and be correlated. The samples are mapped to the target domain Ω by bijective shift mappings $T_i : \Omega_i \to \Omega$ to compute $Y_i = T_i(X_i)$, which are used in the resampling weights

$$w_i = m_i(Y_i)\hat{p}(Y_i)W_{X_i}\left|\frac{\partial T_i}{\partial X_i}\right|, \tag{1}$$

with the resampling output Y sampled proportionally to the w_i. Here, W_{X_i} are *unbiased contribution weights*, random variables that satisfy $\mathbb{E}[W_{X_i}|X_i] = 1/p_{X_i}(X_i)$, usually computed from prior RIS or GRIS passes, $|\partial T_i/\partial X_i|$ is the Jacobian determinant of the shift mapping, and m_i are *resampling MIS weights* that normalize the coverage of the target domain. Convergence to $\bar{p}$ happens when $\mathrm{Var}[\sum_{i=1}^{M} w_i] \to 0$. Unbiased integration of f using Y is typically guaranteed by including a *canonical sample* that completely covers the support of $\hat{p}$ as one of the inputs to GRIS.

2.2 ReSTIR

Reservoir-based spatiotemporal importance resampling (ReSTIR) applies chained GRIS passes across frames to progressively improve the sample distribution, greatly improving real-time image quality. Each pixel maintains a reservoir, which is essentially a tuple (X, W_X, c) carrying a sample, its unbiased contribution weight, and a confidence weight (originally called an "effective sample count") used to weigh domains/techniques when computing resampling MIS. In a GRIS pass, the current pixel's reservoir provides the canonical sample, and other pixels contribute neighboring samples. The reservoir is then updated to the selected output sample Y, whose confidence weight accumulates those of all input samples.

A ReSTIR frame starts by generating per-pixel initial candidates and using RIS to select an initial sample Y. An efficient streaming implementation uses weighted reservoir sampling [Chao 1982] and initializes a reservoir with $c = 1$. This is followed by temporal and spatial reuse passes, which are GRIS passes including neighbor samples as candidates. Finally, reservoirs are used to shade the current frame using estimator $f(X)W_X$ and these reservoirs get passed to the next frame to serve as temporal neighbors. Below are key details of temporal and spatial reuse.

Temporal Reuse. In the temporal GRIS, a neighbor is determined using temporal reprojection following the shading point's motion. The confidence weight c_{temp} of the temporal reservoir is capped by a confidence cap c_{Cap} giving a new confidence after resampling of $\min(c_{\text{Cap}}, c_{\text{temp}}) + 1$. This avoids slow sample turnover in dynamic scenes and unlimited build ups of correlation.

Spatial Reuse. In spatial GRIS, $M - 1$ spatial neighbors are randomly drawn from a disk with a set radius (often 30 pixels). Neighbors whose G-Buffer attributes (e.g., normal, depth) differ too much are rejected. After spatial reuse, the confidence weight sums all accepted neighbors' confidences.

2.3 ReSTIR PT

ReSTIR PT [Lin et al. 2022] aims to reuse all types of paths produced by a path tracer. For a length-d path $\bar{\mathbf{x}} = [\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_d]$, ReSTIR GI [Ouyang et al. 2021] reuses $\mathbf{x}_2$ and later vertices using the same *reconnection shift* as ReSTIR DI [Bitterli et al. 2020]. But this fails when $\mathbf{x}_1$ or $\mathbf{x}_2$ are specular. ReSTIR PT introduced a *hybrid shift* that postpones reconnection until vertices satisfy certain reconnection criteria; until reconnection, the new path is traced by reusing the same random numbers. ReSTIR PT only evaluates a single BSDF lobe at each vertex, allowing shifts to each BSDF lobe separately; here, we omit lobe indices for clarity and defer details to our supplemental.

Path space definition. The path space integral $\sum_{d=1}^{\infty} \int_{\Omega_d} f(\bar{\mathbf{x}})\, \mathrm{d}\bar{\mathbf{x}}$ spans paths of all lengths, where $f(\bar{\mathbf{x}})$ contains BSDF, geometry, sensor, and emitter terms. Starting from primary hit $\mathbf{x}_1$, the path

tracer samples each vertex's BSDF for a path continuation direction and each vertex also performs next event estimation (NEE). To enable different shift mappings in these cases, ReSTIR PT attaches a technique index to $\bar{\mathbf{x}}$ to indicate if the final path vertex is sampled by NEE or a BSDF-sampled continuation ray; this requires splitting the integrand and using MIS weights.

Hybrid Shift. To shift a path $\bar{\mathbf{x}}$ to $\bar{\mathbf{y}} = T(\bar{\mathbf{x}})$, prior gradient-domain rendering work (e.g. Kettunen et al. [2015]) requires the reconnection vertex $\mathbf{x}_k$ and the preceding vertices $\mathbf{x}_{k-1}$ and $\mathbf{y}_{k-1}$ to be all "rough". To avoid storing the base path vertices or replaying the base path, ReSTIR PT precomputes a reconnection vertex $\mathbf{x}_k$ on $\bar{\mathbf{x}}$ by finding the first pair $(\mathbf{x}_{k-1}, \mathbf{x}_k)$ that satisfies two reconnection criteria: a roughness threshold $\min(\alpha_{\mathbf{x}_{k-1}}, \alpha_{\mathbf{x}_k}) \geq \alpha_{\min}$ (with α from the path's sampled lobes) and a distance threshold $\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \geq d_{\min}$ that avoids geometric singularities. This predetermines the replayed path length and allows GPU optimization. Shift invertibility is checked when connecting from $\mathbf{y}_{k-1}$ to $\mathbf{x}_k$.

Jacobian. ReSTIR PT implementations often use a primary sample space (PSS) parameterization. Given path length d and sampling technique t, the path tracer turns random number sequences $\bar{\mathbf{u}}$ into paths. With $\bar{\mathbf{x}} = \mathcal{X}_{t,d}(\bar{\mathbf{u}})$ denoting the produced path, we integrate $F(\bar{\mathbf{u}}) = \omega_t(\bar{\mathbf{x}}) f(\bar{\mathbf{x}})/p_t(\bar{\mathbf{x}})$ over a unit hypercube, where the MIS weight ω_t splits the path space between techniques and $p_t(\bar{\mathbf{x}})$ is the path space PDF. We assume $\bar{\mathbf{u}}$ is associated with $\bar{\mathbf{x}}$ and $T(\bar{\mathbf{u}})$ maps it to a sequence associated with $\bar{\mathbf{y}}$, and denote quantities related to $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ via superscripts x and y. Then, the PSS Jacobian determinant for the hybrid shift is

$$\left|\frac{\partial T}{\partial \bar{\mathbf{u}}}\right| = \frac{p^y_{k-1}(\omega'_{k-1}) G(\mathbf{y}_{k-1} \to \mathbf{x}_k) p^y_k(\omega_k)}{p^x_{k-1}(\omega_{k-1}) G(\mathbf{x}_{k-1} \to \mathbf{x}_k) p^x_k(\omega_k)}, \tag{2}$$

which corresponds to reconnecting from $\mathbf{y}_{k-1}$ to $\mathbf{x}_k$; the Jacobian is 1 if no reconnection happens. Above, $p^x_k(\omega_k) \equiv p(\omega_k | \mathbf{x}_k, -\omega_{k-1})$ is the solid-angle PDF of sampling the direction $-\omega_{k-1} = \widehat{\mathbf{x}_k\mathbf{x}_{k-1}} = \overrightarrow{\mathbf{x}_k\mathbf{x}_{k-1}}/\|\overrightarrow{\mathbf{x}_k\mathbf{x}_{k-1}}\|$, $p^x_k(\omega_k)$ is replaced with 1 for $k = d$, and ω'_{k-1} is a shorthand for $\widehat{\mathbf{y}_{k-1}\mathbf{x}_k}$. The single-sided geometry term $G(\mathbf{x} \to \mathbf{y})$ is $\cos\theta/||\mathbf{x} - \mathbf{y}||^2$, where θ is the angle between $\widehat{\mathbf{y}_{k-1}\mathbf{x}_k}$ and $\mathbf{y}_{k-1}$'s normal.

Implementation. ReSTIR PT assumes a direct illumination (DI) sampling method exists and skips DI paths in its own estimator [Lin et al. 2022]. At initial resampling, a path tree is generated by the path tracer and the indirect lighting paths are fed through RIS to resample one representative path. The associated random seed, reconnection vertex $\mathbf{x}_k$, and the incident radiance and direction on $\mathbf{x}_k$ are stored in the reservoir to facilitate the shift mapping in temporal and spatial reuse.

Our *ReSTIR PT Enhanced* retains the baseline architecture [Lin et al. 2022] while halving the spatial reuse cost by neighbor pairing (Section 3), introducing scene-independent reconnection criteria (Section 4), applying adaptive c_{Cap} reduction for sample decorrelation (Section 5), and it contains various optimizations (Section 6) that enhance ReSTIR PT's performance.

3 A MORE EFFICIENT PAIRED SPATIAL REUSE

ReSTIR PT's spatial and temporal reuse rely on GRIS resampling. Neighbor reuse requires not only shifting neighbors to the current pixel but also needs to shift the current pixel to neighbors to compute good MIS weights. Even with efficient pairwise MIS [Lin et al. 2022], all spatial neighbors require two shifts, doubling the work (Figure 2, left).

For pixel A to reuse from B, we must shift both pixels' paths to the other. We observe that after A reuses from B, B can reuse from A for free, saving 50% by amortizing duplicate spatial shifts (Figure 2, right). To efficiently pair pixels A and B, we precompute a random reuse texture storing a

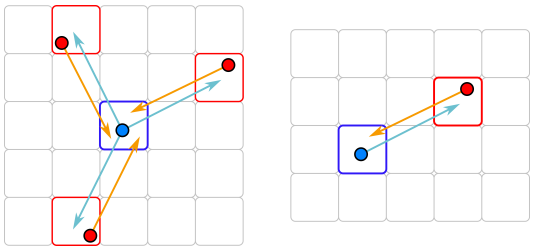


Fig. 2. ***Left****: When performing spatial reuse for the current pixel (blue), the chosen neighbors (red) must shift their paths to the current pixel, while the current pixel must shift its path to the neighbors for MIS weights.* ***Right****: Blue pixel reusing from red requires the same shift mappings as the red pixel reusing from blue. Hence, if blue reuses from red, red can reuse from blue without paying for shifts.*

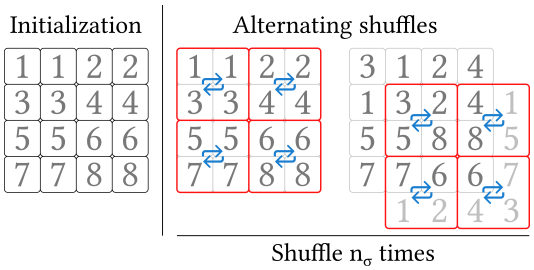


Fig. 3. ***Left****: We initialize the image with consecutive link indices.* ***Right****: We shuffle each 2×2 pixel block with a random permutation and repeat n_σ times to reach the desired standard deviation, offsetting every other shuffle diagonally by one.*

coordinate offset to each pixel's coupled neighbor; A stores the delta to B, and B stores the delta to A. To support reuse from multiple neighbors, we prepare multiple independent textures.

3.1 Computing reuse textures

Producing a reuse texture with coordinate deltas following a given distance distribution is a non-trivial problem with global constraints, and we found optimization-based approaches slow and unreliable. Instead, we permute an initial texture with random shuffles; the deltas produced by the random walks follow a normal distribution.

We assume a target standard deviation $\sigma \geq 0.8$ pixels[1]. We then choose an even image-size like[2] 254×254, and fill it with consecutive link indices (Figure 3, left). We then shuffle the link indices with n_σ tiled 2×2 shuffles, every other offset by 1 while looping over the edge (Figure 3, right). The repeat count is

$$n_\sigma = \left\lfloor \frac{\sigma^2}{2} + 1.46\sigma^{-1} + 1.76\sigma^{-2} + 0.656\sigma^{-3} + 0.5 \right\rfloor \approx \left\lfloor \frac{\sigma^2}{2} + 0.5 \right\rfloor. \tag{3}$$

The repeated shuffles move each link index roughly according to a normal distribution with standard deviation $\sigma/\sqrt{2}$. We then pair the pixels with matching link indices. With correlation decreasing at each shuffle, the coordinate delta roughly follows a normal distribution with standard deviation $\sqrt{\sigma^2/2 + \sigma^2/2} = \sigma$. The negative powers in Equation 3 are function fit correction for small σ.

[1]Smallest supported σ is 0.8, given by applying one shuffle iteration.
[2]Resolutions 256×256 require 16-bit channels to represent coordinate deltas 128 and -128 and past.

The links are now contained within the reuse texture, and links near edges can be long. To make the reuse texture tileable, we break long links by subtracting the width of the square texture from coordinate deltas greater than half the width, and adding it to coordinate deltas less than minus half the width. Finally, we pack these x and y directional deltas into a 16-bit luminance texture.

Finding links. To find coordinate deltas, each pixel locates the other pixel with its link index. Our CUDA implementation constructs an *index table* of size $N/2 \times 2$, where N is the pixel count of the reuse texture. First, every pixel reads its own link index, and writes its pixel location into the index table at location (link index, 0). This is a parallel write with a data race, so only half the pixels succeed. Next, each pixel whose location is not at (link index, 0), writes it to (link index, 1). This enables every pixel to find its linked neighbor in the index table.

3.2 Using reuse textures

Assuming spatial reuse with N neighbors, we preload N reuse textures. To avoid correlation artifacts from texture repeats, we pick a different size for each, e.g., 254, 230 and 210 for three neighbors[3]. Our reuse texture is self-inverting: A links to B, and B links to A. Hence, the reuse texture should be changed every frame. Rather than generating many reuse textures, we randomly flip, mirror, transpose, and offset each reuse texture each frame. Empirically, this completely solves the problem.

3.3 Discussion

Bekaert et al. [2002] split 16×16 pixel blocks into groups of 16 pixels in an N-rooks pattern and perform all-to-all reuse within each group. This also avoids the additional work for MIS weights, but produces structured artifacts at low sample counts. We avoid such structure by overlaying large different-sized Gaussian-permuted reuse textures; each pixel follows a different, random reuse pattern with no block borders. Our reuse textures can also be used in classical path reuse.

4 SCENE-INDEPENDENT RECONNECTION CRITERIA

During ReSTIR's sample reuse, neighbor samples are shifted into the current domain prior to reuse. Ideal shift mapping preserves the target PDF value of the path [Wyman et al. 2023], i.e. $\bar{p}_j(T(\bar{\mathbf{u}}))\left|\frac{\partial T}{\partial \bar{\mathbf{u}}}\right| \approx \bar{p}_i(\bar{\mathbf{u}})$. Without loss of generality, we analyze the reconnection criteria for ReSTIR PT's hybrid shift in primary sample space; we assume $\bar{p}_j(T(\bar{\mathbf{u}})) \approx \bar{p}_i(\bar{\mathbf{u}})$ [4] and focus on $\left|\frac{\partial T}{\partial \bar{\mathbf{u}}}\right|$, which is only affected by the choice of reconnection vertex. A good shift mapping should have $\left|\frac{\partial T}{\partial \bar{\mathbf{u}}}\right| \approx 1$.

The original hybrid shift [Lin et al. 2022] selects reconnection vertex $\mathbf{x}_k$ using a distance threshold $||\mathbf{x}_k - \mathbf{x}_{k-1}|| \geq d_{\min}$ that forbids short path reconnections (i.e., near corners) and a roughness threshold $\min(\alpha_{\mathbf{x}_{k-1}}, \alpha_{\mathbf{x}_k}) \geq \alpha_{\min}$ to forbid connections on highly glossy surfaces. These thresholds can be suboptimal and often require scene-specific tuning, since suitable reconnection distances and roughness levels are interdependent.

We introduce a *dual ray footprint threshold* and a new *single-vertex roughness threshold*, giving reconnection constraints that are more robust in the presence of specular materials, do not require scene-specific tuning, and enable earlier (and hence cheaper) reconnection opportunities.

The intuition for the new footprint thresholds comes from considering the geometric meaning of $\left|\frac{\partial T}{\partial \bar{\mathbf{u}}}\right| \approx 1$. We first express $\left|\frac{\partial T}{\partial \bar{\mathbf{u}}}\right|$ as the product of two ratios,

$$\frac{p^y_{k-1}(\omega'_{k-1})G(\mathbf{y}_{k-1} \rightarrow \mathbf{x}_k)}{p^x_{k-1}(\omega_{k-1})G(\mathbf{x}_{k-1} \rightarrow \mathbf{x}_k)} \quad \text{(4a)} \qquad \text{and} \qquad \frac{p^y_k(\omega_k)}{p^x_k(\omega_k)}. \quad \text{(4b)}$$

[3] Our heuristic algorithm tries to minimize near-periods within 3840 pixels, i.e., cases where a multiple of a texture's width is close to a multiple of another texture's width.

[4] Geometry and microfacet NDF terms are cancelled in $\omega_t(\bar{\mathbf{x}})f(\bar{\mathbf{x}})/p_t(\bar{\mathbf{x}})$, which avoids extreme values in the shift.

The first one is between the area densities of vertex $\mathbf{x}_k$ when hypothetically traced from $\mathbf{y}_{k-1}$ and when originally traced from $\mathbf{x}_{k-1}$. The second one is between the solid angle densities of ω_k when hypothetically sampled with outgoing direction $\widehat{\mathbf{y}_{k-1}\mathbf{x}_k}$ and when originally sampled with outgoing direction $\widehat{\mathbf{x}_{k-1}\mathbf{x}_k}$. Our threshold aims to bound the change of these densities.

The reciprocal of area probability density of a vertex on a surface has the geometric meaning of the footprint, i.e., the area represented by the sample. Intuitively, a longer reconnection distance or rougher material at $\mathbf{x}_{k-1}$ enlarges the ray footprints and reduces the density change at shift mapping, making a shift reconnecting *from* $\mathbf{x}_{k-1}$ more useful. To bound Equation 4b, we look at *inverse ray footprint* $(p^x_k(\omega_k)G(\mathbf{x}_k \to \mathbf{x}_{k-1}))^{-1}$, which is the reciprocal area density of $\mathbf{x}_{k-1}$ produced by tracing from $\mathbf{x}_k$. Our analysis shows that thresholding this footprint equivalently bounds the change of solid angle density in Equation 4b.

Specifically, our new criterion for reconnection is then defined by a dual ray footprint threshold:

$$\min((p^x_{k-1}(\omega_{k-1})G(\mathbf{x}_{k-1} \to \mathbf{x}_k))^{-1}, (p^x_k(\omega_k)G(\mathbf{x}_k \to \mathbf{x}_{k-1}))^{-1}) \geq \frac{c}{100}\frac{||\mathbf{x}_0 - \mathbf{x}_1||^2}{\left\langle n_{\mathbf{x}_1}, \widehat{\mathbf{x}_1\mathbf{x}_0}\right\rangle/(4\pi)}, \quad (5)$$

where the RHS is a constant multiple of the primary ray footprint that measures the distance between the nearby primary hits. We found the constant multiplier $c = 0.02$ (i.e., $c/100 = 0.0002$) to work well in diverse scenes. We also enforce a single-vertex threshold ($\alpha_{\mathbf{x}_{k-1}} \geq \alpha_{\min}$) at vertex $\mathbf{x}_{k-1}$ to cheaply avoid trying to reconnect in some tricky edge cases.

4.1 Explanation of the dual footprint thresholds

We provide a brief explanation of how our dual footprint thresholds achieve the bounding and provide a full derivation in the supplemental material. Consider shifting a path $\bar{\mathbf{x}}$ in one pixel to $\bar{\mathbf{y}} = T(\bar{\mathbf{x}})$ in another pixel. For a candidate reconnection vertex $\mathbf{x}_k$ on $\bar{\mathbf{x}}$, connecting from $\mathbf{y}_{k-1}$ instead of $\mathbf{x}_{k-1}$ perturbs the *path-traced area density* of $\mathbf{x}_k$ and the angular density of ω_k. We want these relative changes to be small: for small ϵ,

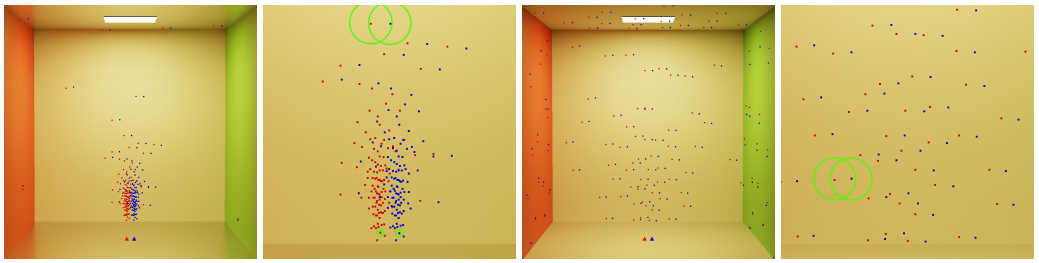
$$\left|\frac{p^y_{k-1}(\omega'_{k-1})G(\mathbf{y}_{k-1} \to \mathbf{x}_k)}{p^x_{k-1}(\omega_{k-1})G(\mathbf{x}_{k-1} \to \mathbf{x}_k)} - 1\right| < \epsilon \quad (6a) \qquad \text{and} \qquad \left|\frac{p^y_k(\omega_k)}{p^x_k(\omega_k)} - 1\right| < \epsilon, \quad (6b)$$

which imply $(1-\epsilon)^2 < \left|\frac{\partial T}{\partial \bar{\mathbf{u}}}\right| < (1+\epsilon)^2$.

Ray footprint threshold. Equation 6a requires bounding the change in area density at $\mathbf{x}_k$ when reconnecting from $\mathbf{y}_{k-1}$ instead of $\mathbf{x}_{k-1}$. Assuming random replay preserves local area density (Figure 4), we analyze how the area density of $\mathbf{x}_k$ changes under a spatial displacement $\Delta\mathbf{x}_k = \mathbf{y}_k - \mathbf{x}_k$, where $\mathbf{y}_k$ is obtained by replaying from $\mathbf{y}_{k-1}$.

We assume $\|\Delta\mathbf{x}_k\|$ is bounded by a constant multiple of the *primary ray footprint* [Müller et al. 2021], $\|\Delta\mathbf{x}_k\| < c_1 R^{\bar{\mathbf{x}}}_{\text{pri}} = c_1\sqrt{\frac{\|\mathbf{x}_0-\mathbf{x}_1\|^2}{\langle n_{\mathbf{x}_1}, \widehat{\mathbf{x}_1\mathbf{x}_0}\rangle/(4\pi)}}$. Further assuming that relative density variation within a footprint of radius $\sqrt{c_2/(p^x_{k-1}(\omega_{k-1})G(\mathbf{x}_{k-1}\to\mathbf{x}_k))}$ is bounded by ϵ, a sufficient condition for Equation 6a is $(p^x_{k-1}(\omega_{k-1})G(\mathbf{x}_{k-1} \to \mathbf{x}_k))^{-1} > \frac{c_1^2}{c_2}(R^{\bar{\mathbf{x}}}_{\text{pri}})^2$. This *ray footprint* threshold avoids reconnections near density peaks caused by glossy BSDFs or geometric singularities. In practice, we collapse the constants into a single parameter c in Equation 5; larger c makes reconnection more conservative, i.e., favor postponing.

Inverse ray footprint threshold. The ray footprint bounds changes in area density at $\mathbf{x}_k$ but does not capture changes in the BSDF sampling PDF $p^x_k(\omega_k)$ caused by altering the outgoing direction $-\omega_{k-1}$, which can be significant for low-roughness glossy materials.



(a) *GGX roughness* 0.3 **(b)** *(zoomed in)* **(c)** *GGX roughness* 0.7 **(d)** *(zoomed in)*

Fig. 4. *Visualizing secondary hits (red and blue points) sampled for two pixels (red and blue triangles on the floor) by BSDF sampling with the same random numbers reveals a near-exact spatial shift in the sample populations and thus sample density. Green circles show radii proportional to sample footprints, i.e., reciprocal probability densities. At a fixed surface point, larger footprints generally imply a smaller density change between the red and blue populations and thus safer reconnection. Studying different GGX roughnesses (a-b, c-d) shows larger roughnesses generally allow safer reconnection.*

We therefore define the *inverse ray footprint* $(p_k^x(\omega_k)G(\mathbf{x}_k \to \mathbf{x}_{k-1}))^{-1}$. Assuming approximate BSDF PDF reciprocity for glossy materials,[5] it can be interpreted as the area per sample of a reverse-traced ray (as if the path is traced from the light). Under the same smoothness assumptions as above, and with $G(\mathbf{x}_k \to \mathbf{x}_{k-1}) \approx G(\mathbf{x}_k \to \mathbf{y}_{k-1})$ for distant reconnections, a sufficient condition for Equation 6b is $(p_k^x(\omega_k)G(\mathbf{x}_k \to \mathbf{x}_{k-1}))^{-1} > \frac{c_1^2}{c_2}(R_{\text{pri}}^{\bar{\mathbf{x}}})^2$. We reuse the same parameter c for both thresholds for simplicity, yielding the combined test in Equation 5. This enables safe reconnection to distant glossy surfaces without enforcing a roughness threshold at $\mathbf{x}_k$.[6]

4.2 Single-vertex roughness threshold

We retain a roughness threshold at $\mathbf{x}_{k-1}$ to guard against cases where parallax, curvature, or very low roughness can make footprint-based bounds unreliable. These cases would otherwise require excessively large geometric bounds or overly large footprint tolerances.

The roughness threshold also handles reconnection to environment lights, where $\Delta\mathbf{x}_k$ is unbounded and sharp changes in angular density may occur. With this safeguard, we can use a tighter footprint parameter (c) that performs well in typical scenes. For non-parametric materials, we discuss using the BSDF sampling PDF as a roughness proxy in the supplemental material.

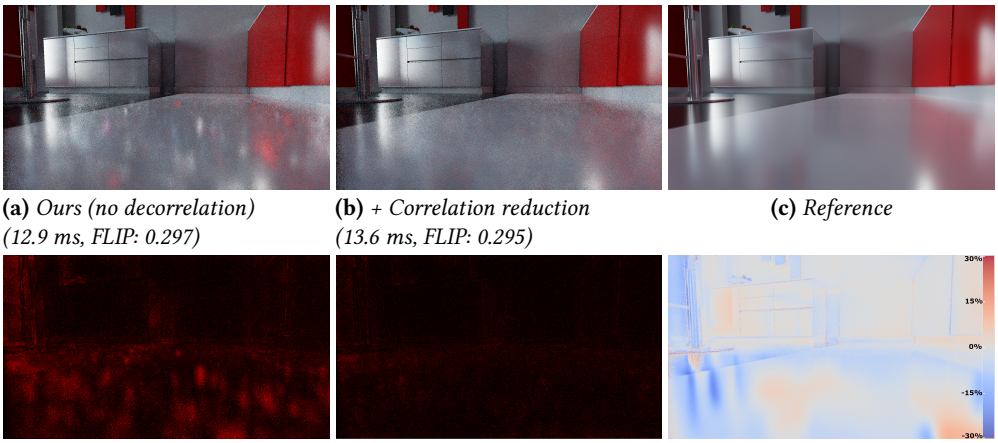
5 CORRELATION REDUCTION WITH DUPLICATION MAPS

One of the greatest challenges to ReSTIR's stability is sample correlation. The main sources for sample correlation are

- Low-probability, high-energy fireflies from initial sampling, and
- Imperfect shift mappings that artificially shift samples into high-energy fireflies.

These fireflies spread to neighbor pixels via spatiotemporal reuse, forming correlation blobs (e.g., disks or streaks) sometimes occupying numerous pixels and persisting over many frames. Poorly chosen parameters exacerbate visual artifacts (e.g., a small spatial reuse radius).

[5]We assume $p_k^x(\omega_k) = p(\omega_k|\mathbf{x}_k, -\omega_{k-1}) \approx p(-\omega_{k-1}|\mathbf{x}_k, \omega_k)$, which is exact for NDF sampling of microfacet glossy material and approximate for VNDF sampling [Heitz 2018].
[6]For diffuse or emissive $\mathbf{x}_k$, reconnection does not affect $p_k^x(\omega_k)$ and the inverse ray footprint test is skipped.



(a) *Ours (no decorrelation) (12.9 ms, FLIP: 0.297)* **(b)** *+ Correlation reduction (13.6 ms, FLIP: 0.295)* **(c)** *Reference*

(d) *Duplication map of (a)* **(e)** *Duplication map of (b)* **(f)** *Bias visualization*

Fig. 5. *Evaluating our de-duplication method (b) in* KITCHEN*, where camera motion along the ground introduces many correlations (a) due to reuse of suboptimal temporal neighbors. We compute a duplication map (d, e), counting duplicate samples in each 17×17 neighborhood (black to red represents 0–20% duplication in the visualization) and use this to adaptively decrease c_{Cap}. This introduces some bias, appearing as energy loss. Our average absolute relative bias ($mean(|bias|/ref.)$) is 3.25% in (b). We also visualize this bias per-pixel (f), where more glossy surfaces exhibit larger bias.*

Correlation artifacts make denoising ReSTIR challenging, as artifacts get detected as signal. Previous work fights sample correlation by partially reusing paths while keeping prefixes independent [Kettunen et al. 2023] or mutating samples to reduce impoverishment [Sawhney et al. 2024]. But these methods are expensive and only reduce certain types of correlation.

We introduce a simple but effective solution based on c_{Cap} reduction, essentially limiting the temporal confidence weights adaptively. The motivation is simple—how high we weight temporal samples directly impacts potential correlation strength. Highly weighting temporal reuse provides more chances for fireflies to spread spatially and makes them more temporally persistent.

Our solution measures prior frame's correlations around each pixel and reduces c_{Cap} in highly correlated areas. Our correlation measure simply counts pixels in a spatial neighborhood that share the current pixel's sample, by which we mean the samples are shifted copies from the same initial candidate. We detect this by comparing the random seeds already stored for (potential) random replay. We implement our method as follows:

(1) After finishing each frame, compute a *sample duplication map*. Each pixel counts how many reservoirs in the surrounding 17×17 region share its random seed. This count is divided by 288 to compute a duplication score $D \in [0, 1]$.
(2) When doing temporal resampling, look up the duplication score corresponding to the temporal reservoir used (i.e., backprojecting the current pixel into prior frame).
(3) We compute a modified c_{Cap} for temporal reuse as $c_{\text{Cap}} = \text{lerp}(c_{\text{Cap}}^{\text{Default}}, c_{\text{Cap}}^{\text{min}}, D^{\alpha})$. Where $c_{\text{Cap}}^{\text{Default}}$ is the default (e.g., 20 in Lin et al. [2022]) and $c_{\text{Cap}}^{\text{min}}$ is the minimum c_{Cap} used when encountering $D = 1$. α sets the sensitivity of the reduction. $\alpha = 1$ gives a linear reduction, while α closer to 0 gives a quick reduction as D increases.

In our tests, we found $c_{\text{Cap}}^{\text{min}} = 1$ and $\alpha = 0.1$ yield good results. Our adaptive c_{Cap} dramatically reduces correlation artifacts caused by high-energy samples produced by sampling or resampling.

Because this approach adjusts confidence weights based on specific samples, the partition of unity of MIS weights m_i is violated, introducing a small bias. This bias only occurs in correlated areas where we modify c_{Cap}; essentially our approach trades correlation for bias. Figure 5 shows our method in a very difficult scene, where it averages 3.25% absolute relative bias.

6 IMPLEMENTATION IMPROVEMENTS

Beyond our algorithmic contributions, we introduce implementation-focused improvements that further enhance performance and robustness. We reduce redundancy by combining separate ReSTIR direct and indirect lighting passes into one unified algorithm (Section 6.1), apply GPU optimizations to reduce thread divergence (Section 6.2), and adopt existing techniques to reduce color (Section 6.3) and disocclusion noise (Section 6.4). The quality improvements these bring can be seen aggregated in Figure 1 and Figure 13, but more individual ablations are provided in the supplemental.

6.1 Unifying Direct Lighting and Global Illumination

Real-time renderers often separately compute direct and indirect lighting, with indirect light often approximated more aggressively. But using ReSTIR, we can efficiently sample high-quality indirect illumination. Rather than running separate direct and indirect ReSTIR passes, as in Lin et al. [2022], we combine both using a single reservoir.

The idea is simple. Lin et al. [2022] start by initially sampling a *path tree* by emanating NEE rays from BSDF-sampled vertices $[\mathbf{x}_2, ..., \mathbf{x}_n]$ where n is limited by the maximum allowed bounces. Initial resampling selects and stores a single *path* in the reservoir, i.e., $\bar{\mathbf{x}} = [\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_d]$ with a *single* NEE/BSDF ray connecting to a light at $\mathbf{x}_d$ (see Section 2.3) with $d \geq 3$ originally. Direct light can be handled by simply tracing another NEE ray from $\mathbf{x}_1$ when generating the path tree, and giving the (unified) initial resampling a chance to select a shorter direct lighting path ($d = 2$) from the tree. The selected path is thus sampled from the full path space.

This removes the cost and storage of separate ReSTIR passes for direct and indirect light. Somewhat surprisingly, it also improves quality as direct lighting now benefits from ReSTIR PT's built-in shift mapping (especially on glossy highlights). Implementation details for modifying the MIS weights and source PDFs in primary sample space are provided in the supplemental material.

For scenes with many lights, we optionally apply RIS when generating NEE samples at $\mathbf{x}_1$. As in ReSTIR DI, visibility is excluded from the target function and tested only for the selected light. We reuse the presampled light-tile scheme from Wyman and Panteleev [2021] to improve cache coherence: each frame precomputes 128 tiles with 1024 lights, from which each 8×8 screen tile draws a light tile to generate candidates per bounce.

6.2 Improving GPU Performance by Reducing Divergence

Monte Carlo path tracing incurs substantial GPU divergence as neighboring paths often traverse very different geometry and materials. ReSTIR PT further increases divergence due to additional control flow and intermediate data, e.g., reservoirs. We apply a set of optimizations that reduce both code and data divergence, substantially improving GPU efficiency.

6.2.1 Low-level Code Optimization. A naive ReSTIR PT implementation contains many branches arising from reservoir sampling and heterogeneous path cases. We reduce divergence by replacing most branches with conditional moves and simplifying computations while preserving equivalent results. We also reduce reservoir storage from 88 bytes to 64 bytes by removing unnecessary fields and applying lossy compression to selected quantities (details in the supplemental material).

6.2.2 Stream Compaction for Random Replay. In many scenes, only a few pixel–neighbor pairs require random replay, yet warp-level execution forces all threads to wait. We parallelize over

pixel–neighbor pairs, applying stream compaction to discard pairs that do not require replay. This reduces warp divergence and the number of active warps, yielding a substantial speedup.

6.2.3 Forced NEE Light Reconnection. During random replay, paths ending at NEE-sampled light vertices incur expensive light sampling that other path types avoid. Since replayed random numbers usually select the same light (e.g., with power-based sampling), we force reconnection to such vertices if no earlier reconnection is found. This removes light sampling from random replay, improving performance. Any potential variance increase is mitigated by path MIS weights, which already downweight these cases (as such surfaces are typically glossy).

6.2.4 Russian Roulette. Initial resampling is often the most expensive stage due to long, divergent paths. We apply Russian roulette to reduce average path length at initial sampling, but remove it from random replay to avoid killing valid paths during reuse. This effectively removes the roulette dimensions from the path parametrization and applies an external roulette at initial path sampling, modifying the sampling PDF in primary sample space (see supplemental). Although this slightly increases noise in initial samples, spatiotemporal reuse largely suppresses its impact in the final result, and the roulette failures at shift mapping are completely avoided.

6.3 Reducing Color Noise

ReSTIR inherently suffers from color noise because resampling operates on a scalar target function $\hat{p}$, while the integrand $\boldsymbol{F}(y)$ is RGB-valued. GRIS therefore samples primarily according to luminance, leaving chroma poorly importance-sampled.

Prior work [Kettunen et al. 2023; Lin et al. 2022] noted that the estimate $\boldsymbol{F}(Y)W_Y$ can be improved by marginalizing over the random index choice. In ReSTIR PT, this improvement comes at almost no extra cost: since $\hat{p} = |\boldsymbol{F}|$, $\boldsymbol{F}(Y_i)$ is already evaluated during spatial reuse. We accumulate vector-valued resampling weights $\boldsymbol{w}_i = m_i(Y_i)\boldsymbol{F}(Y_i)W_{X_i}\,|\partial T/\partial X_i|$ and use $\sum_{i=1}^{M} \boldsymbol{w}_i$ for shading. As spatial neighbors typically contain uncorrelated chroma noise[7], spatial reuse naturally averages it out. This decouples resampling and shading: scalar weights drive future resampling, while vector weights are used for shading. Unlike similar techniques for direct lighting [Wyman and Panteleev 2021], our method incurs no additional computations.

6.4 Reducing Disocclusion Noise with Dual Motion Vectors

Disocclusion causes severe noise in ReSTIR as newly visible surfaces lack any temporal history. Dual motion vectors [Zeng et al. 2021] address this by reprojecting pixels using an alternative motion vector that assumes consistent relative motion between occluding and disoccluded surfaces.

We apply this directly to ReSTIR's temporal resampling and observe a substantial reduction in disocclusion noise. While prior work avoids copy-paste artifacts by storing incident radiance fields, this issue does not arise in ReSTIR PT: unbiased path resampling prevents pattern cloning. The remaining correlated noise is mild and can be further reduced using our decorrelation strategy (Section 5).

7 EVALUATION

Our evaluation uses a PC with NVIDIA RTX 5880 Ada graphics card and AMD Ryzen Threadripper PRO 3975WX 32-Cores CPU. We use the Falcor [Benty et al. 2020] real-time rendering framework and start optimization from the original ReSTIR PT source code [Lin et al. 2022]. All images are rendered in 1920×1080. We use the HDR version of FLIP [Andersson et al. 2021] as the metric for

[7] Although spatiotemporal reuse can also create spatially correlated color noise, such correlation is usually not among the randomly sampled spatial neighbors in the new frame.

Table 1. *Frame and pass costs (in milliseconds) averaged over four scenes (Veach Ajar, Carousel, Opera House, and Bathroom) of varying complexity. Methods under the dashed line incur added cost.*

Method	Total frame	Initial sampling	Temporal reuse	Spatial reuse	ReSTIR DI and others
Baseline	35.73	10.59	5.30	14.79	5.05
+Code microop (Section 6.2.1)	32.98	10.70	4.16	13.06	5.07
+Forced NEE reconnect (Section 6.2.3)	29.75	11.44	3.40	9.73	5.19
+Replay compaction (Section 6.2.2)	26.81	11.79	2.66	6.99	5.37
+Paired spatial reuse (Section 3)	25.02	12.56	2.80	4.11	5.55
+Russian roulette (Section 6.2.4)	16.52	5.21	2.24	3.83	5.24
+Unify DI & GI (Section 6.1)	13.04	6.47	2.14	3.43	1.00
+New thresholds (Section 4)	14.51	6.68	2.92	3.60	1.31
+All improvements (Section 5, 6.3, 6.4)	15.53	6.77	3.20	3.73	1.83

image error, as it closely aligns with the perceptual difference when flipping to compare images. We borrow ReSTIR PT's default temporal and spatial parameters, namely $c_{\text{Cap}} = 20$ for temporal resampling and use of 3 random spatial neighbors in a 30-pixel radius.

For paired spatial reuse, we compute 3 pairing textures that realize Gaussian distributions with the same average sample radial distance for spatial resampling. As the mean sample distance of an isotropic Gaussian with standard deviation σ is $\sigma\sqrt{\pi/2}$, and the mean sample distance of a uniform disk of radius R is $2R/3$, we propose choosing $\sigma = \sqrt{8/(9\pi)}R$ to match prior work's empirically chosen R. For the default $R = 30$ in ReSTIR, this gives $\sigma = 16.0$. Our paired spatial reuse does not fully halve costs, due to overhead from splitting spatial reuse in two: a pre-pass shifts each path to all paired neighbors, and a second pass performs resampling.

We use 1 spp path tracing (one path tree per pixel) to produce the initial samples. For RIS of NEE samples (Section 6.1), we draw 32 light samples from the light tiles for NEE candidates at the primary hit, following Wyman and Panteleev [2021]. We reduce the number of NEE light candidates at deeper bounces using an inverse-square decay schedule, allocating $32/B^2$ samples (clamped to a minimum of 1) at bounce index B.

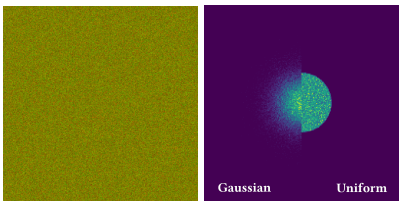
7.1 Performance

Table 1 shows performance of our techniques, with each row adding one new feature/optimization on top of a baseline of Lin et al.'s [2022] public source code. We first measure the speedup from our cost-reduction techniques, which provide an average 2.74× speedup across the four tested scenes. These scenes were chosen to reflect a range of geometry and material complexity. Results for individual scenes are provided in the supplemental material. To provide further insight into the effect of our low-level GPU optimizations, we profiled Opera House using NSight Graphics. The profiler data indicate that the optimizations in Section 6.2.1–6.2.3 reduce thread divergence and improve GPU computation efficiency. Specifically, SM warp occupancy increases from 22.4% to 31.1%, active threads per warp increase from 15.3 to 19.9, and warp latency decreases from 347k to 241k cycles, all without changing the sampler behavior. Applying Russian roulette (Section 6.2.4) further improves these metrics to 34.9%, 20.6, and 82k cycles, respectively.

Under the dashed line we show the added cost brought by our techniques from Section 4, 5, and 6. *New thresholds* reflects the cost of our scene-independent reconnection criteria that improves shift mapping quality. The *all improvements* row applies all other techniques (decorrelation, color noise and disocclusion noise reduction). These increase average cost by 19% versus our fastest version. But even with these quality improvements, we are still 2.30× faster than Lin et al. [2022]. Our performance also scales better with resolution, as reported in the supplemental document.

(a) *Original image* **(b)** *Swap once* **(c)** *Swap twice*

Fig. 6. *Our pairing texture enables an invertible stochastic Gaussian "blur." Swapping twice recovers the original image. Here we use a pairing texture with* $\sigma = 16.0$.



(a) *Pairing texture* **(b)** *Histogram*

Fig. 7. *Visualizing* $\sigma = 16.0$*, which has the same average sample distance as* $R = 30$ *uniform disk sampling.*

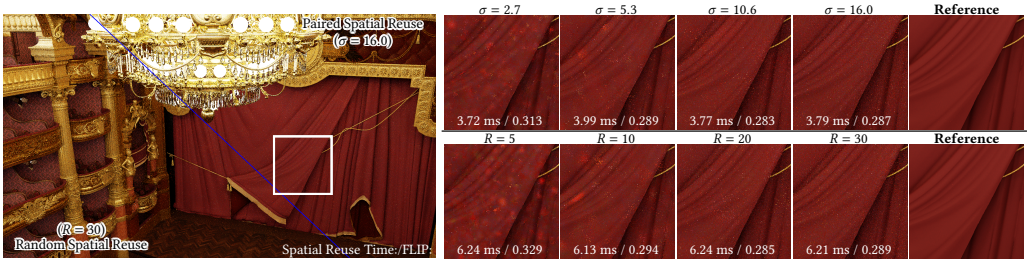


Fig. 8. *Comparing our paired spatial reuse (top) with traditional random spatial reuse from a uniform disk (bottom) in the* Opera House*. In each column, we choose a* σ *value that produces the same average sample distance as a uniformly-sampled disk of radius* R.

Storage. Our method also reduces storage relative to the baseline through two changes: compressing the ReSTIR PT reservoir and unifying the reservoirs for direct and indirect lighting. Because each ReSTIR pass requires two sets of reservoirs to support temporal reuse, these changes reduce per-pixel storage from $2 \times (88 + 16)$ bytes in the baseline implementation (which uses 16-byte reservoirs for ReSTIR DI) to 2×64 bytes. With a 1920×1080 render resolution, this lowers memory consumption from 431 MB to 265 MB.

7.2 Ablation Studies

We study our paired spatial reuse (Section 3), new reconnection criteria (Section 4), and decorrelation technique (Section 5) more closely here. Isolated studies of techniques in Section 6 are included in the supplemental document.

Evaluation of paired spatial reuse. We encode pairing textures as two-channel tilable images, storing integer offsets $(\Delta x, \Delta y) \in [-127, 127]^2$ per pixel (wrapped by tiling). Figure 7a visualizes a 254×254 pairing texture. In Figure 6 we validate our texture by swapping each pixel with its paired neighbor; one swap gives a blur with Gaussian noise while a second recovers the original image.

Figure 8 compares our paired spatial reuse against traditional uniform disk samples with disk radii $R = 5, 10, 20, 30$, matched to Gaussian kernels with equal average sample distances. Paired reuse speeds ReSTIR's spatial reuse by 1.63× on average by halving shift mapping costs with only a modest overhead. It also lowers FLIP error, likely because a Gaussian distribution concentrates samples closer to the center (i.e., Figure 7b), increasing the probability to select compatible neighbors.

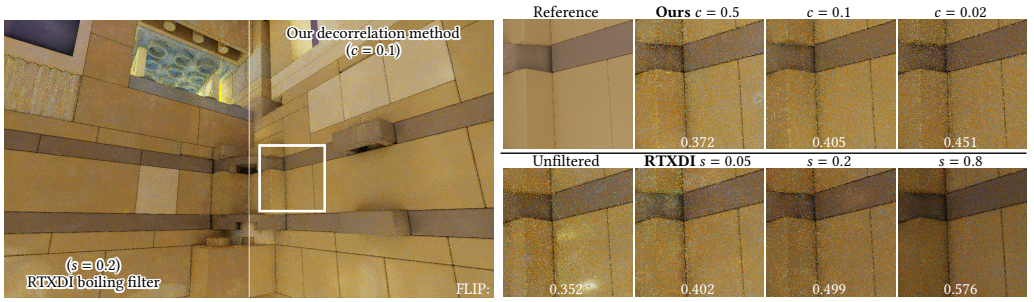


Fig. 9. *Comparing our duplication-based decorrelation (top) and RTXDI's boiling filter (bottom) in the* Tower Bridge *scene, which exhibits strong correlations (see* unfiltered *inset) due to poor initial sample quality. The parameters reflect weak, medium (default), and strong correlation reduction in both methods. Increasing FLIP errors reflect the darkening bias the methods introduce.*

Evaluation of the new reconnection criteria. We comprehensively evaluated our new reconnection criteria. For the single-vertex roughness threshold, we use ReSTIR PT's default $\alpha = 0.2$. For dual footprint thresholds, we empirically found $c = 0.02$ (see Equation 5) to be near-optimal across a wide range of scenes. Ablation results for different c values are provided in the supplemental.

Figures 10, 11, and 12 compare our criteria with Lin et al.'s [2022] reconnection rules, which use a world-space distance threshold and a two-vertex roughness test. We set the distance threshold to 2% of the shortest scene dimension, typical to prior work. Across variations in viewing distance, material roughness, and lighting conditions, our thresholds consistently deliver higher-quality resampling with improved robustness.

Our ray footprint thresholds resemble the path footprint criteria of Bekaert et al. [2003] and Müller et al. [2021]. However, we found path footprints alone often fail to eliminate unconnectable paths because they do not consider material properties at the reconnection vertex. Additional comparisons are included in our supplemental.

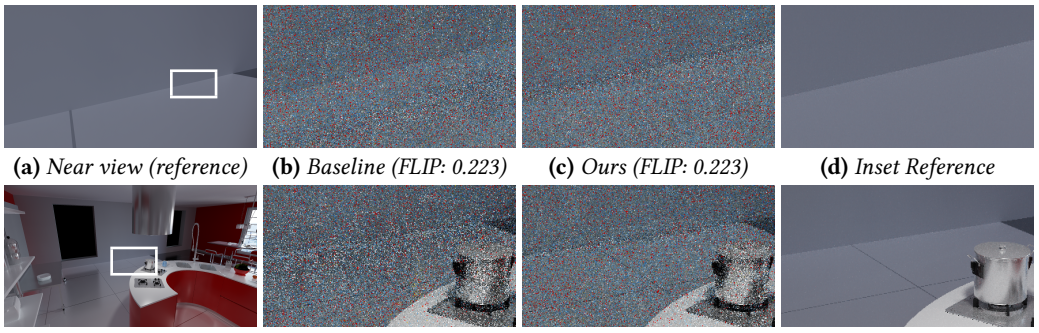
Ablation study of duplication-based decorrelation. In Figure 9, we compare our duplication-based correlation reduction method to the boiling filter in RTXDI SDK [NVIDIA 2021]. This filter computes an average resampling weight $\bar{w}$ over each warp's pixels and clears any reservoir with weight $w_j > a\bar{w}$ where $a = -9 + 10/s$ is controlled by *strength factor* $s \in (0, 1]$. Note that RTXDI's boiling filter causes substantial energy loss to remove most artifacts.

7.3 Overall quality evaluation

Figure 1 and Figure 13 show the overall performance and quality improvement with our enhanced ReSTIR PT. Compared to Lin et al. [2022], our enhanced ReSTIR PT reduces correlation artifacts, color and disocclusion noise. It exhibits better quality thanks to our new reconnection criteria (e.g., see the surfaces inside the glass in Figure 1) and the unification of direct and indirect lighting, yet runs significantly faster, reaching 2.08×-3.05× speedup over Lin et al. [2022].

7.4 Bias and Convergence

Our decorrelation technique (Section 5) introduces bias, but disabling it restores unbiasedness while retaining all remaining benefits of ReSTIR PT Enhanced (see Figure 14 and additional comparisons at the end of the supplemental document). In practice, this bias is usually small relative to the noise level. As shown in Figure 15, the biased and unbiased variants of our method have nearly overlapping convergence curves under both MSE and FLIP over the time range relevant to real-time



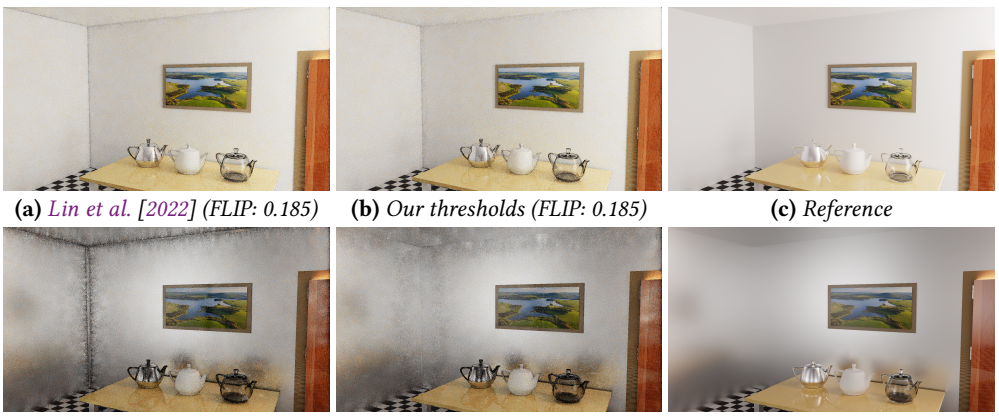
(a) *Near view (reference)* **(b)** *Baseline (FLIP: 0.223)* **(c)** *Ours (FLIP: 0.223)* **(d)** *Inset Reference*

(e) *Far view (reference)* **(f)** *Baseline (FLIP: 0.296)* **(g)** *Ours (FLIP: 0.291)* **(h)** *Inset Reference*

Fig. 10. *Our new reconnection thresholds adapt to varying viewing distances, improving consistency in the* Kitchen *scene. In the baseline [Lin et al. 2022], thresholds good for the near view behave poorly for the far view.*

(a) *Lin et al. [2022] (FLIP: 0.339)* **(b)** *Our thresholds (FLIP: 0.322)* **(c)** *Reference*

Fig. 11. *A failure case shown by Lin et al. [2022] in* San Miguel*. Previously, the hybrid shift sampled caustics poorly (pointed by the arrow) from near-delta distant highlights (the lamp reflection on the far window). Our new threshold effectively adjusts "roughness" by distance, here allowing reconnection to the window.*



(a) *Lin et al. [2022] (FLIP: 0.185)* **(b)** *Our thresholds (FLIP: 0.185)* **(c)** *Reference*

(d) *Lin et al. [2022] (FLIP: 0.271)* **(e)** *Our thresholds (FLIP: 0.252)* **(f)** *Reference*

Fig. 12. *Lin et al.'s [2022] reconnections perform similarly to our work in diffuse scenes like* Veach Ajar*. With metallic walls (roughness of 0.3), their baseline shows significantly more noise. Our new thresholds automatically enlarge reconnection distances, giving more consistent noise.*

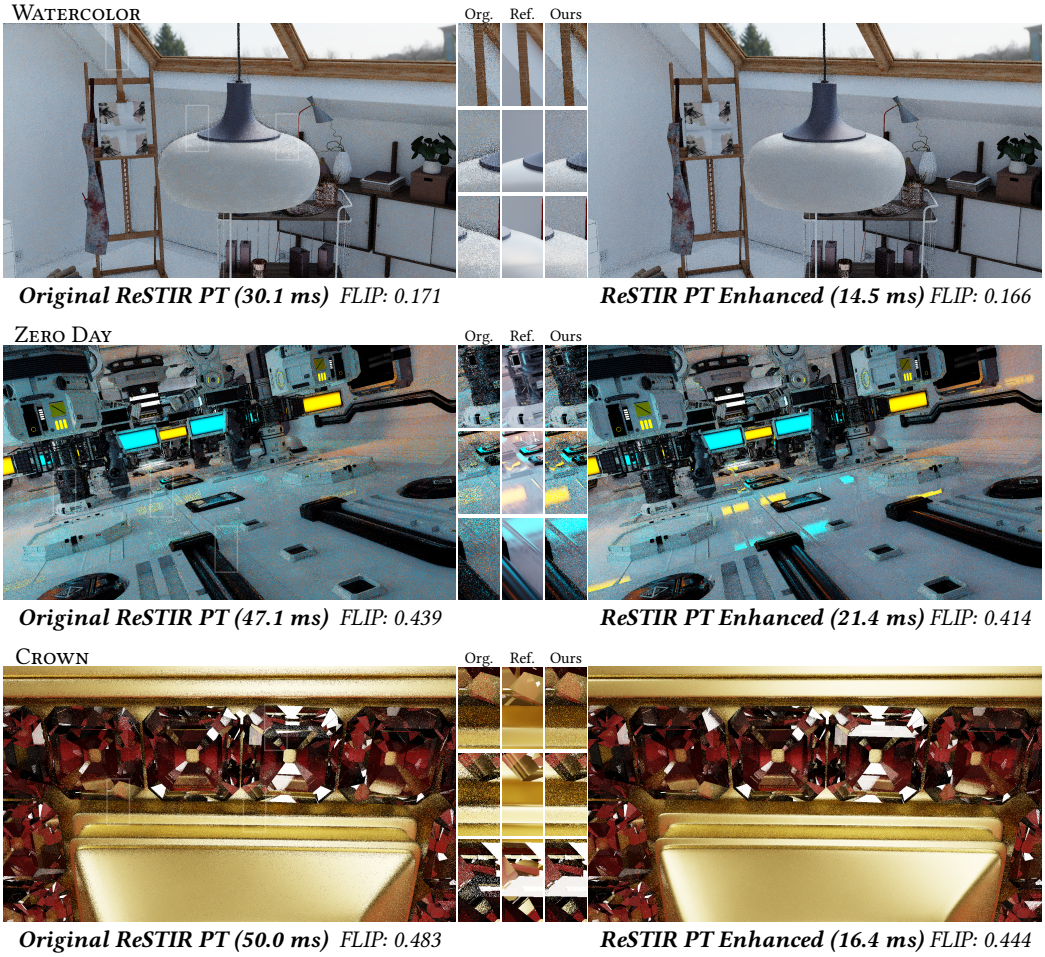
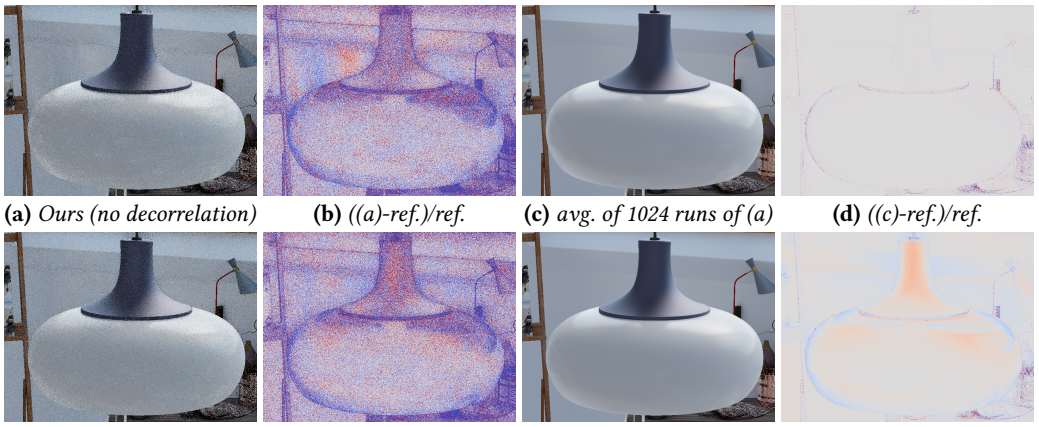


Fig. 13. *Comparing Lin et al. [2022] to our enhancements in the Watercolor, Zero Day, and Crown scenes. We sample glossy surfaces noticeably better, thanks to improved MIS on primary hits and better reconnection thresholds. We also reduce color and disocclusion noise with fewer correlation artifacts.*

rendering (< 0.1 s). Our default biased version even achieves lower MSE initially as it prevents the spread of outliers. With longer accumulation, however, its convergence curve gradually flattens as the bias emerges. In more typical scenes such as Watercolor, where correlation is less severe than in Kitchen (Section 5), the convergence curve flattens more slowly with an overall lower bias level. Overall, we find the default decorrelated, biased variant more preferable for real-time image quality. When a noise-free unbiased image is desired, accumulating it with temporal reuse disabled is typically more efficient [Lin et al. 2022].

8 CONCLUSION AND FUTURE WORK

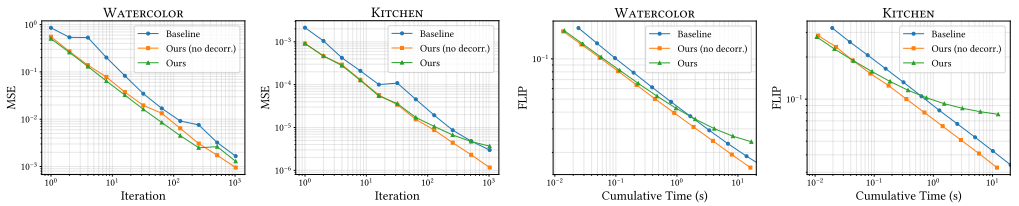
We presented a suite of new ideas to enhance ReSTIR PT, improving both performance and robustness. Optimizations range from low-level GPU code improvements to larger algorithmic changes, showing up to a 3× speedup. We improve robustness via better shift mapping, correlation reduction, and reductions in color and disocclusion noise, providing consistently higher visual quality.



(a) *Ours (no decorrelation)* **(b)** *((a)-ref.)/ref.* **(c)** *avg. of 1024 runs of (a)* **(d)** *((c)-ref.)/ref.*

(e) *Ours* **(f)** *((e)-ref.)/ref.* **(g)** *avg. of 1024 runs of (e)* **(h)** *((g)-ref.)/ref.*

Fig. 14. *Our method without decorrelation (a) exhibits more correlation artifacts in* Watercolor *under an upward-moving camera, including noise clusters on the lamp and correlated dark and bright trails in the disocclusion region. After averaging 1024 independent runs, our default version reveals systematic error (due to bias) as can be seen in (h), shown using the same color scale as in Figure 5.*



(a) *MSE vs. iteration count for two scenes* **(b)** *FLIP vs. cumulative time for two scenes*

Fig. 15. *Convergence plots for* Watercolor *(Figure 13, 14) and* Kitchen *(Figure 5), comparing baseline ReSTIR PT, our method without decorrelation (Ours no decorr.), and our default method with decorrelation (Ours). Our default method exhibits more bias in* Kitchen *due to heavier correlation reduction. For each scene and method, we run the same camera animation 1024 times with different random seeds for sampling, capture the same view during camera motion, and average the results across runs. (a) shows mean squared error versus iteration count. (b) remaps iteration count of the data points in (a) to cumulative rendering time of the captured frames and reports FLIP error. FLIP is less sensitive to fireflies and correlation artifacts, but still sensitive to regional brightness differences.*

Several contributions generalize to other reuse algorithms: paired spatial reuse, footprint-based reconnection thresholds, and temporal history control via duplication maps. We further found that a unified ReSTIR, sampling direct and indirect lighting together, improves both speed and quality.

Despite these advances, ReSTIR PT faces open challenges motivating future work. High-frequency features remain sensitive to temporal noise during motion, and overall quality is bounded by the effectiveness of initial sampling, especially for low-probability light transport effects such as caustics. Some issues are partially addressed by recent work such as Area ReSTIR [Zhang et al. 2024], suggesting exploring smooth transitions from ReSTIR PT Enhanced to an enhanced Area ReSTIR within a shared framework. Additional opportunities also remain for further algorithmic and hardware-aware optimizations on newer GPU architectures.

ACKNOWLEDGMENTS

We thank Miika Aittala for useful discussions on pairwise reuse textures, Aaron Lefohn for general support, and the anonymous reviewers for their constructive feedback.

REFERENCES

Pontus Andersson, Jim Nilsson, Peter Shirley, and Tomas Akenine-Möller. 2021. Visualizing errors in rendered high dynamic range images. In *Eurographics-Short Papers*. Eurographics-European Association for Computer Graphics, 25–28.

Pablo Bauszat, Victor Petitjean, and Elmar Eisemann. 2017. Gradient-domain path reusing. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–9.

Philippe Bekaert, Mateu Sbert, and John H Halton. 2002. Accelerating Path Tracing by Re-Using Paths. *Rendering Techniques* 2, 2002 (2002), 125–134.

Philippe Bekaert, Philipp Slusallek, Ronald Cools, Vlastimil Havran, and Hans-Peter Seidel. 2003. A custom designed density estimation method for light transport. (2003).

Nir Benty, Kai-Hwa Yao, Petrik Clarberg, Lucy Chen, Simon Kallweit, Tim Foley, Matthew Oakes, Conor Lavelle, and Chris Wyman. 2020. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor

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 Transactions on Graphics (Proceedings of SIGGRAPH)* 39, 4 (July 2020).

Min-Te Chao. 1982. A general purpose unequal probability sampling plan. *Biometrika* 69, 3 (1982), 653–656.

Henrik Halen, Andreas Brinck, Kyle Hayward, and Xiangshun Bei. 2021. Global Illumination Based on Surfels. Talk at ACM SIGGRAPH 2021, Advances in Real-Time Rendering in Games. https://www.ea.com/seed/news/siggraph21-global-illumination-surfels

Eric Heitz. 2018. Sampling the ggx distribution of visible normals. *Journal of Computer Graphics Techniques (JCGT)* 7, 4 (2018), 1–13.

Markus Kettunen, Daqi Lin, Ravi Ramamoorthi, Thomas Bashford-Rogers, and Chris Wyman. 2023. Conditional resampled importance sampling and ReSTIR. In *SIGGRAPH Asia 2023 Conference Papers*. 1–11.

Markus Kettunen, Marco Manzi, Miika Aittala, Jaakko Lehtinen, Frédo Durand, and Matthias Zwicker. 2015. Gradient-domain path tracing. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–13.

Daqi Lin, Markus Kettunen, Benedikt Bitterli, Jacopo Pantaleoni, Cem Yuksel, and Chris Wyman. 2022. Generalized Resampled Importance Sampling: Foundations of ReSTIR. 41, 4 (2022), 75:1–75:23. https://doi.org/10.1145/3528223.3530158

Zander Majercik, Jean-Philippe Guertin, Derek Nowrouzezahrai, and Morgan McGuire. 2019. Dynamic Diffuse Global Illumination with Ray-Traced Irradiance Fields. *Journal of Computer Graphics Techniques (JCGT)* 8, 2 (5 June 2019), 1–30.

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372* (2021).

NVIDIA. 2020. NVIDIA Real-Time Denoiser Delivers Best-in-Class Denoising in Ubisoft's Watch Dogs Legion. https://developer.nvidia.com/blog/nvidia-real-time-denoiser-delivers-best-in-class-denoising-in-watch-dogs-legion/

NVIDIA. 2021. RTXDI SDK – Light Resampling In Practice. https://developer.nvidia.com/blog/new-video-light-resampling-in-practice-with-rtxdi/. Accessed: 2025-11-29.

NVIDIA. 2025. DLSS 4: Transforming Real-Time Graphics with AI. https://research.nvidia.com/labs/adlr/DLSS4/. Technical Report.

Yaobin Ouyang, Shiqiu Liu, Markus Kettunen, Matt Pharr, and Jacopo Pantaleoni. 2021. ReSTIR GI: Path Resampling for Real-Time Path Tracing. *Computer Graphics Forum* (2021).

Rohan Sawhney, Daqi Lin, Markus Kettunen, Benedikt Bitterli, Ravi Ramamoorthi, Chris Wyman, and Matt Pharr. 2024. Decorrelating restir samplers via mcmc mutations. *ACM Transactions on Graphics* 43, 1 (2024), 1–15.

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*. 1–12.

Justin Talbot, David Cline, and Parris Egbert. 2005. Importance Resampling for Global Illumination. In *Eurographics Symposium on Rendering (2005)*, Kavita Bala and Philip Dutre (Eds.). The Eurographics Association.

Daniel Wright, Krzysztof Narkowicz, and Patrick Kelly. 2022. Lumen: Real-time Global Illumination in Unreal Engine 5. Talk at ACM SIGGRAPH 2022, Advances in Real-Time Rendering in Games. https://advances.realtimerendering.com/s2022/SIGGRAPH2022-Advances-Lumen-Wright%20et%20al.pdf

Chris Wyman, Markus Kettunen, Daqi Lin, Benedikt Bitterli, Cem Yuksel, Wojciech Jarosz, and Pawel Kozlowski. 2023. A gentle introduction to restir path reuse in real-time. In *ACM SIGGRAPH 2023 Courses*. 1–38.

Chris Wyman and Alexey Panteleev. 2021. Rearchitecting Spatiotemporal Resampling for Production. In *High-Performance Graphics - Symposium Papers*.

Zheng Zeng, Shiqiu Liu, Jinglei Yang, Lu Wang, and Ling-Qi Yan. 2021. Temporally Reliable Motion Vectors for Real-time Ray Tracing. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 79–90.

Song Zhang, Daqi Lin, Markus Kettunen, Cem Yuksel, and Chris Wyman. 2024. Area ReSTIR: Resampling for Real-Time Defocus and Antialiasing. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–13.