Decorrelating ReSTIR Samplers via MCMC Mutations

ROHAN SAWHNEY, Carnegie Mellon University and NVIDIA, USA
DAQI LIN, NVIDIA, USA
MARKUS KETTUNEN, NVIDIA, Finland
BENEDIKT BITTERLI, NVIDIA, USA
RAVI RAMAMOORTHI, NVIDIA and UC San Diego, USA
CHRIS WYMAN, NVIDIA, USA
MATT PHARR, NVIDIA, USA

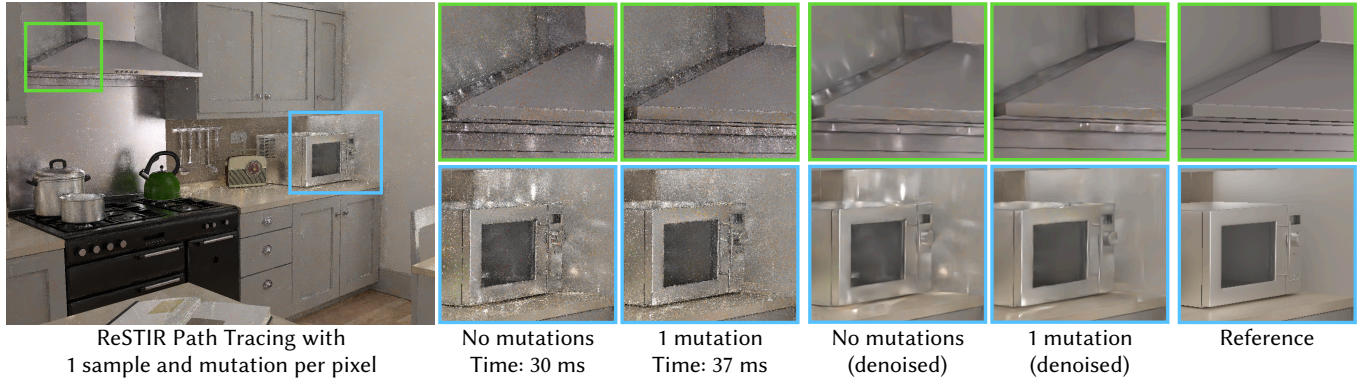| ReSTIR Path Tracing with 1 sample and mutation per pixel | No mutations Time: 30 ms | 1 mutation Time: 37 ms | No mutations (denoised) | 1 mutation (denoised) | Reference |

Fig. 1. A single sample per pixel (spp) comparison of indirect illumination rendered using ReSTIR Path Tracing (PT) [Lin et al. 2022] with and without our sample mutations. By performing even a single mutation per sample, our approach can suppress correlation artifacts that may arise within ReSTIR samplers due to spatiotemporal reuse. Mutations improve visual fidelity of both rendered and denoised results (with the OptiX denoiser [NVIDIA 2017]) while leaving mean squared error unchanged.

Monte Carlo rendering algorithms often utilize correlations between pixels to improve efficiency and enhance image quality. For real-time applications in particular, repeated *reservoir resampling* offers a powerful framework to reuse samples both spatially in an image and temporally across multiple frames. While such techniques achieve equal-error up to 100× faster for real-time direct lighting [Bitterli et al. 2020] and global illumination [Ouyang et al. 2021; Lin et al. 2021], they are still far from optimal. For instance, spatiotemporal resampling often introduces noticeable correlation artifacts, while reservoirs holding more than one sample suffer from impoverishment in the form of duplicate samples. We demonstrate how interleaving *Markov Chain Monte Carlo (MCMC)* mutations with reservoir resampling helps alleviate these issues, especially in scenes with glossy materials and difficult-to-sample lighting. Moreover, our approach does not introduce any bias, and in practice we find considerable improvement in image quality with just a single mutation per reservoir sample in each frame.

Authors' addresses: Rohan Sawhney, rsawhney@nvidia.com, Carnegie Mellon University and NVIDIA, USA; Daqi Lin, daqil@nvidia.com, NVIDIA, USA; Markus Kettunen, mkettunen@nvidia.com, NVIDIA, Finland; Benedikt Bitterli, bbitterli@nvidia.com, NVIDIA, USA; Ravi Ramamoorthi, ravir@cs.ucsd.edu, NVIDIA and UC San Diego, USA; Chris Wyman, cwyman@nvidia.com, NVIDIA, USA; Matt Pharr, mpharr@nvidia.com, NVIDIA, USA.

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

Additional Key Words and Phrases: real-time rendering, resampled importance sampling, weighted reservoir sampling, Markov chain Monte Carlo

1 INTRODUCTION

The efficiency of rendering algorithms often hinges on their ability to effectively evaluate similar integrals by reusing samples across pixels [Ward et al. 1988; Jensen 1996; Veach and Guibas 1997; Keller 1997]. In real-time path tracing, sample reuse becomes more critical since tracing rays is computationally intensive even on high-end consumer GPUs [Kilgariff et al. 2018]. Moreover, while existing denoisers drastically improve image quality even at low sample counts [Chaitanya et al. 2017; Schied et al. 2017, 2018; Kozlowski and Cheblokov 2021; NVIDIA 2022], they are unable to reconstruct features missing from their input samples. Thus, sample reuse across pixels is often the only means to improve sampling quality given limited computational budgets. Compared to methods that generate independent samples, reuse is also at times the only practical approach to render challenging scenes with caustics and tricky lighting [Hachisuka and Jensen 2009; Veach and Guibas 1997].
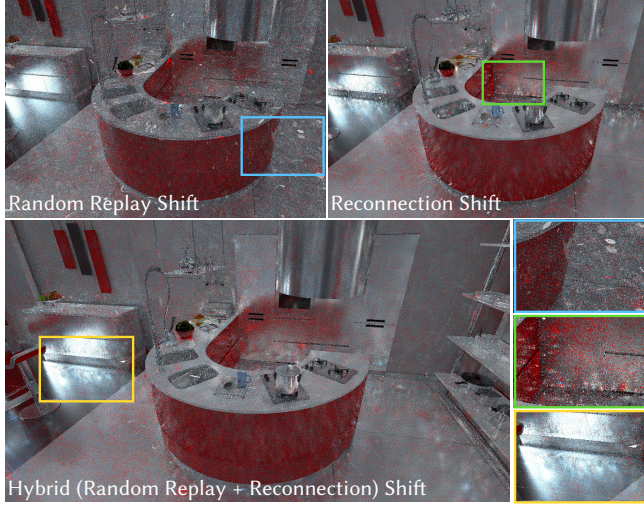
Fig. 2. Glossy scenes with difficult-to-sample lighting rendered using ReSTIR PT often contain correlation artifacts irrespective of the selected shift mapping strategy (reviewed in Section 4.3 and Lin et al. [2022, Section 7]). Artifacts result from suboptimal importance sampling and over-enthusiastically sharing a few high-contribution samples between pixels.
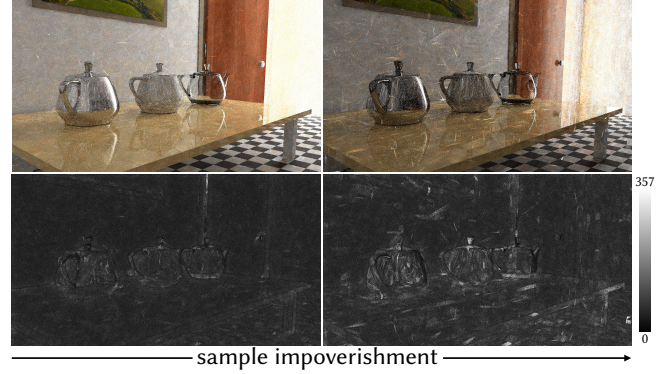

Fig. 3. Reservoir resampling suffers from sample impoverishment as it becomes more difficult to sample light-carrying paths. *Top row, left to right:* The Veach Ajar scene rendered using ReSTIR PT (random replay shift) at 1 spp with the door's angle decreasing. *Bottom row:* Heat maps visualize duplicate samples in 20×20 pixel neighborhoods. Black represents no duplicates, while white indicates the number of identical samples in a neighborhood.

Recent sampling algorithms for real-time ray tracing achieve massive speedups in scenes with complex illumination by sharing samples spatially within an image and temporally across frames [Bitterli et al. 2020; Ouyang et al. 2021; Lin et al. 2021, 2022]. These so-called *ReSTIR*[1] based techniques select N high-contribution samples from a larger streamed candidate pool of size M. They do so by reformulating *resampled importance sampling (RIS)* [Talbot et al. 2005] in terms of *weighted reservoir sampling (WRS)* [Chao 1982]. While RIS effectively produces samples in proportion to an *arbitrary* target function (*e.g.*, the integrand of the rendering equation), WRS makes resampling efficient by reducing storage costs from $O(M)$ to $O(N)$. Repeated resampling across pixels then helps distribute important samples over several frames for estimation.

Though ReSTIR derives impressive efficiency gains from correlated sampling, the benefits of repeated resampling are not indefinite. When only a few high-contribution samples have been identified, iterative spatial reuse creates blotchy artifacts as several pixels reuse the same sample (Figure 12, *top row*). Such undersampling artifacts eventually fade away with temporal reuse over several frames, using a user-specified parameter to balance pixel error with correlations from sample reuse (Figure 4). Unfortunately, simply emphasizing error reduction via greater reuse adds lag under camera movement with dynamically changing lighting and geometry (Section 2.4), and introduces distracting low-frequency artifacts (Figures 2 and 3) akin to those in photon mapping [Hachisuka and Jensen 2009], Metropolis Light Transport (MLT) [Veach and Guibas 1997] and Virtual Point Light (VPL) methods [Dachsbacher et al. 2014].

As spatiotemporal correlations are difficult to quantify, resolving artifacts is challenging. For instance, popular denoisers that compute first- and second-order moments (*e.g.*, Schied et al. [2017]) are

less effective given imprecise variance estimates with correlated samples. For ReSTIR, trying to reduce such artifacts by increasing the reservoir size N is also ineffective, as resampling *with replacement* [Chao 1982] produces duplicate samples in the presence of strong correlations (see Wyman and Panteleev [2021, Figure 19]).

Inspired by work on *Sequential Monte Carlo (SMC)* [Doucet et al. 2001] and *Population Monte Carlo (PMC)* [Cappé et al. 2004], we demonstrate that interleaving MCMC mutations with reservoir resampling (Section 3) helps alleviate correlations and impoverishment, especially in scenes with glossy materials and difficult lighting. Unlike MLT where mutations drive information sharing across pixels, our mutations instead primarily *mitigate* artifacts caused by spatiotemporal reuse, with little-to-no visual impact in scenes where artifacts do not arise (Figure 13). Our approach highlights the complementary strengths of resampling and mutations for real-time rendering: resampling identifies samples with large contributions proportional to a pixel's target distribution, while mutations diversify the resampled population by locally perturbing samples in proportion to the same target distribution. Furthermore, like Veach and Guibas [1997]'s bias elimination strategy for MLT, we show that resampling eliminates the need for *any* burn-in period with *Metropolis–Hastings (MH)* mutations [Metropolis et al. 1953; Hastings 1970] (Section 2.5, Appendix A). This drives considerable image quality improvements from even a single mutation per frame for each reservoir sample (Figures 1, 7, 9 and 11).

From an implementation perspective, our approach requires only simple additions to existing ReSTIR algorithms (see Algorithm 3)—we mutate reservoir samples using Metropolis–Hastings and an appropriate target function every frame after temporal reuse. This is immediately followed by an adjustment to each mutated sample's *contribution weight* to maintain *detailed balance* and ensure unbiased estimation. Overall, our contributions include:

- Demonstrating how to incorporate MCMC mutations within ReSTIR samplers to lessen correlation artifacts from spatiotemporal resampling.

---

[1]acronym for **Re**servoir-based **S**patio-**T**emporal **I**mportance **R**esampling

- Showing how to adjust the RIS contribution weight of mutated samples in an unbiased fashion for further resampling.
- Situating ReSTIR in the broader family of techniques that jointly apply resampling and mutations to sampling problems, such as MLT, SMC and PMC (see Table 2).

Though mutations can help produce images with better visual fidelity, we observe that, similar to blue-noise sampling [Mitchell 1987; Georgiev and Fajardo 2016; Heitz and Belcour 2019], they do not necessarily reduce error (Figure 9). This is true both in scenes with difficult-to-sample lighting (Figure 12, *top row*), as well as in scenes with ample lighting and diffuse materials where resampling easily finds important samples (Figure 13). Moreover, despite the effectiveness of mutations in scenes with strong correlations, we show in the supplemental document that with our current approach, even infinite mutations cannot eliminate correlations entirely.

We start with the key building blocks of our approach in the next section, and postpone discussion about related work to Section 6 for better context when comparing with our method.

2 BACKGROUND

The rendering equation [Kajiya 1986] gives the outgoing radiance L_{out} leaving a point y in the direction ω. Expressed as an integral over directions, it is

$$L_{\text{out}}(y, \omega) = L_{\text{e}}(y, \omega) + \int_{S^2} L_{\text{in}}(y, \omega_i)\, \rho(y, \omega, \omega_i)\, |\cos \theta_i|\, d\omega_i. \quad (1)$$

Here L_{e} is the emitted radiance, $L_{\text{in}}(y, \omega_i)$ is the incoming radiance from the direction ω_i, $\rho(y, \omega, \omega_i)$ is the BSDF and θ_i is the angle between ω_i and the surface normal at y. Absent participating media, the incident radiance L_{in} is defined recursively as $L_{\text{in}}(y, \omega_i) = L_{\text{out}}(t(y, \omega_i), -\omega_i)$; the function $t(y, \omega_i)$ returns the point on the closest surface from y in direction ω_i. Integrating over the sphere of directions S^2 then gives the total radiance scattered towards ω; the rendering equation can be estimated with Monte Carlo as

$$L_{\text{out}}(y, \omega) \approx L_{\text{e}}(y, \omega) + \frac{1}{N} \sum_{i=1}^{N} \frac{L_{\text{in}}(y, \omega_i)\, \rho(y, \omega, \omega_i)\, |\cos \theta_i|}{p(\omega_i)}, \quad (2)$$

where $p(\omega_i)$ is the probability density function (PDF) with respect to solid angle used to sample incident directions ω_i.

As in Kajiya's formulation, sometimes it is more convenient to reformulate Equation 1 over surfaces. To keep the discussion independent of the choice of formulation, we use $\int_\Omega f(x)\, dx$ to generically represent the integral we want to evaluate with Ω as its domain. This integral can likewise be estimated using

$$\widehat{I}_{\text{MC}} := \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}, \quad (3)$$

where x_i are independent random samples drawn from any *source* PDF p that is non-zero on the support of f. In rendering, one often draws samples proportional to individual terms of the rendering equation to reduce variance (*e.g.*, the BSDF ρ). To perform even better importance sampling, ReSTIR instead uses RIS to draw samples *approximately* proportional to the product of multiple terms in the integrand (*e.g.*, $L_{\text{in}} \cdot \rho \cdot |\cos \theta|$).

We review RIS and generalized RIS next (Sections 2.1 and 2.2); Section 2.3 discusses a streaming RIS implementation via reservoir sampling. Section 2.4 then describes how correlations arise within ReSTIR due to resampling. Section 2.5 reviews the Metropolis–Hastings algorithm we use in Section 3 to resolve correlation artifacts.

2.1 Resampled Importance Sampling (RIS)

RIS [Talbot et al. 2005; Lin et al. 2022] enables *unbiased* estimation and sample generation from a non-negative target function $\hat{p}$ with an unknown normalization factor $\int_\Omega \hat{p}(y)\, dy$. It does so by rewriting the standard Monte Carlo estimator from Equation 3 as

$$\frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{\hat{p}(x_i)} \left(\int_\Omega \hat{p}(y)\, dy \right). \quad (4)$$

The normalization factor is estimated by generating $M \geq 1$ candidate samples $\mathbf{y} = \{y_1, \ldots, y_M\}$ from a source PDF q that may be suboptimal but easy to sample from (*e.g.*, $q \propto \rho$), yielding

$$\frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{\hat{p}(x_i)} \left(\frac{1}{M} \sum_{j=1}^{M} \frac{\hat{p}(y_j)}{q(y_j)} \right). \quad (5)$$

The samples $\mathbf{x} = \{x_1, \ldots, x_N\}$ in turn are selected by randomly choosing an index $j \in \{1, \ldots, M\}$, N times, from the candidate pool $\mathbf{y}$ with discrete probabilities:

$$\mathbb{P}(j \mid \mathbf{y}) = \frac{w(y_j)}{\sum_{k=1}^{M} w(y_k)}. \quad (6)$$

Here the *resampling weight* w for each candidate y_j is given by

$$w(y_j) = \frac{1}{M} \hat{p}(y_j) W(y_j), \quad (7)$$

where $W := 1/q(y_j)$ is called the *(unbiased) contribution weight* for y_j. The selected samples x_i are likewise given contribution weights

$$\widehat{W}(x_i) := \frac{1}{\hat{p}(x_i)} \left(\sum_{j=1}^{M} w(y_j) \right) \quad (8)$$

that assume the role of a reciprocal PDF, though these weights are only unbiased estimates for elements of the resampled set $\mathbf{x}$. This is because the parenthesized term for the normalization factor of $\hat{p}$ is itself an estimator that has variance. Each $x_i \in \mathbf{x}$ is also distributed only approximately in proportion to $\hat{p}$ (*i.e.*, $\hat{p}$ is sampled perfectly only in the limit as $M \to \infty$). Since we resample *with replacement*, the set $\mathbf{x}$ can contain duplicate samples, which reflects that samples are selected in proportion to $\hat{p}$. With this setup, Talbot [2005] shows that the RIS estimator

$$\widehat{I}_{\text{RIS}} := \frac{1}{N} \sum_{i=1}^{N} f(x_i) \widehat{W}(x_i) \quad (9)$$

is unbiased as long as $\hat{p}$ and q are non-zero on the support of f, *i.e.*,

$$\mathbb{E}[\widehat{I}_{\text{RIS}}] = \int_\Omega f(x)\, dx. \quad (10)$$

Combining with Multiple Importance Sampling (MIS). There are often several reasonable sampling strategies available in rendering, *e.g.*, BSDF or light sampling. MIS [Veach and Guibas 1995b] allows multiple strategies to be combined robustly within RIS [Talbot 2005]. When each candidate y_j has its own source PDF q_j, then MIS weights generalize the parenthesized term in Equation 5 with

$$\sum_{j=1}^{M} m_j(y_j) \frac{\hat{p}(y_j)}{q_j(y_j)}. \tag{11}$$

Here, $m_j \geq 0$ is the MIS weight for the jth sampling technique. These weights must form a partition of unity, *i.e.*, $\sum_{j=1}^{M} m_j(y) = 1$. A common choice is the *balance heuristic* $m_j(y) = q_j(y)/\sum_{k=1}^{M} q_k(y)$ [Veach and Guibas 1995b]. With MIS, the resampling weight in Equation 7 becomes:

$$w(y_j) = m_j(y_j)\hat{p}(y_j)W(y_j), \quad \text{where} \quad W(y_j) = \frac{1}{q_j(y_j)}. \tag{12}$$

Notice we recover $m_j = 1/M$ when source PDFs are the same for each sample y_j. MIS weights play an important role in ReSTIR—beyond reducing noise in the resampling weights, they also remove bias when the supports of the source and target distributions do not match integrand f's support (see Section 4 in Bitterli et al. [2020] and Section 5 in Lin et al. [2022] for further details).

In practice, using RIS with the balance heuristic is costly, as all sampling strategies (*i.e.*, the source PDFs) must be evaluated for each candidate sample y_j. Bitterli [2022, Chapter 9.1.3] provides a similarly robust but more efficient heuristic called *Pairwise MIS*, which only requires $O(M)$ PDF evaluations over the entire candidate pool. We use pairwise MIS when the number of sampling strategies M is greater than 2 (*e.g.*, during spatial resampling in ReSTIR; see Section 2.4); otherwise we use the balance heuristic.

2.2 Generalized Resampled Importance Sampling (GRIS)

So far we assumed the resampling inputs $y_j \sim q_j$ share a common integration domain Ω with integrand f. This assumption may no longer hold when reusing spatially or temporally across an image (as in ReSTIR), and depends on the integral formulation used for the rendering equation. For instance, ReSTIR applied to global illumination [Ouyang et al. 2021; Lin et al. 2022] generates samples from PDFs with respect to solid angle. Reuse across pixels therefore requires a change of integration domain, necessitating a correction term in the resampling weights [Ouyang et al. 2021, Equation 11]. ReSTIR for direct lighting [Bitterli et al. 2020] instead integrates over the surface of all lights, ensuring Ω is fixed across samples.

Recent work by Lin et al. [2022] generalizes RIS to use candidate samples y_j originating from different domains Ω_j. It achieves this via *shift mapping*, *i.e.*, a bijective transformation of samples from one pixel to corresponding samples on another pixel [Lehtinen et al. 2013]. In particular, if Ω denotes the domain of integration for f, and $S_j : \Omega_j \rightarrow \Omega$ are shifts that map $y_j \in \Omega_j$ to the modified sample $y'_j \in \Omega$, then the resampling weight for y_j becomes

$$w(y_j) = m_j(y'_j)\hat{p}(y'_j)W(y_j) \cdot \left| \frac{\partial y'_j}{\partial y_j} \right|, \tag{13}$$

ALGORITHM 1: Weighted reservoir sampling ($N = 1$)

```
 1:  class Reservoir
 2:      x ← ∅                                  ▷ output sample
 3:      w_sum ← 0                   ▷ sum of resampling weights
 4:      M ← 0                    ▷ number of samples seen so far
 5:      Ŵ ← 0          ▷ contribution weight (set in Algorithm 2)
 6:      function UPDATE(y, w)
 7:          w_sum ← w_sum + w
 8:          M ← M + 1
 9:          if RAND() < (w/w_sum) then
10:              x ← y
```

where the Jacobian determinant $|\partial y'_j/\partial y_j|$ accounts for the change of integration domain from Ω_j to Ω. (Jacobians also appear in MIS weights m_j; see Appendix B). The rest of the RIS procedure in Section 2.1 remains unchanged—substituting these resampling weights to Equation 8 provides the contribution weight for the selected y'_j.

Various shift mappings have been proposed to maximize the similarity between y'_j and y_j such that $|\partial y'_j/\partial y_j| \approx 1$ [Hua et al. 2019, Section 3]. We describe the shift mappings we use in Section 4.

2.3 Weighted Reservoir Sampling (WRS)

WRS [Chao 1982] facilitates efficient RIS implementations using a single pass over elements in a stream $\{y_1, \ldots, y_M\}$ to select a random sample. As in Section 2.1, each stream element has an associated resampling weight w. The basic idea is to process the stream one element at a time, and to select—from the $m < M$ elements processed so far—a sample y_j with probability $w(y_j)/\sum_{k=1}^{m} w(y_k)$. The next stream element y_{m+1} then replaces y_j with probability $w(y_{m+1})/\sum_{k=1}^{m+1} w(y_k)$. The stream length M need not be known ahead of time, and WRS can be used to select $N > 1$ samples if needed [Wyman 2021, Chapter 22.6].

WRS reduces the storage needed for resampling to $O(N)$. A lightweight data structure called a *reservoir* is typically used to process the stream and store the selected samples, the stream length M and the weight sum $\sum_{j=1}^{M} w(y_j)$; see Algorithm 1.

2.4 Reservoir-based Spatiotemporal Resampling

ReSTIR applies RIS and WRS in a *chained* fashion within and across pixels of an image. The first key idea is to approximately importance sample multiple terms in the rendering equation's integrand through a per-pixel target function $\hat{p}$. The second is to reuse samples from neighboring pixels to exploit the similarity between their target functions. The algorithm performs four steps every frame:

(1) *(Initial resampling)* Select N samples from a candidate pool of M samples at each pixel. Equations 12 and 8 provide the resampling and contribution weights for the candidate and selected samples respectively. A reservoir stores the selected samples and their *estimated* contribution weights.

(2) *(Temporal resampling)* Use Algorithm 2 to reuse samples across two corresponding pixels in consecutive frames t and $t-1$. The resampling weight for each sample is computed using the contribution weight already stored in its reservoir.

ALGORITHM 2: Combining two reservoirs for temporal reuse ($N = 1$)

Input: Reservoirs r_i and r_j for pixels i and j from frames t and $t - 1$ (*resp.*), and a cap for the sample count in r_j

Output: A combined reservoir s for frame t

1: **function** COMBINETEMPORALRESERVOIRS(i, j, r_i, r_j, M_{cap})
2:     ▷*Cap confidence weight for r_j*
3:     $r_j.M \leftarrow \min(r_j.M, M_{cap})$
4:     ▷*Compute resampling weight for sample in r_i*
5:     $x_i \leftarrow r_i.x$
6:     $m_i \leftarrow$ COMPUTEMIS(x_i, $\hat{p}_i$, $r_i.M$, $\hat{p}_j$, $r_j.M$)   ▷*Equation 31*
7:     $w_i \leftarrow m_i \cdot \hat{p}_i(x_i) \cdot r_i.\widehat{W}$ ▷*Equation 12*
8:     ▷*Shift sample in r_j to pixel i and compute its resampling weight*
9:     $x'_j, |\partial x'_j / \partial x_j| \leftarrow$ SHIFTMAP($r_j.x$, j, i)   ▷*Section 4*
10:    $m_j \leftarrow$ COMPUTEMIS(x'_j, $\hat{p}_i$, $r_j.M$, $\hat{p}_i$, $r_i.M$)   ▷*Equation 32*
11:    $w_j \leftarrow m_j \cdot \hat{p}_i(x'_j) \cdot r_j.\widehat{W} \cdot |\partial x'_j / \partial x_j|$ ▷*Equation 13*
12:    ▷*Combine weighted samples into a single reservoir*
13:    Reservoir s
14:    s.UPDATE(x_i, w_i)
15:    s.UPDATE(x'_j, w_j)
16:    $s.M \leftarrow r_i.M + r_j.M$
17:    $s.\widehat{W} \leftarrow \frac{1}{\hat{p}_i(s.x)} s.w_{sum}$ ▷*Equation 8*
18:    **return** s

---

(3) *(Spatial resampling)* For each pixel, select K random reservoirs from a small spatial neighborhood and merge them into the pixel's reservoir. This is similar to Algorithm 2 and can be repeated multiple times; for reference see Bitterli et al. [2020, Algorithm 4] and Ouyang et al. [2021, Algorithm 2].

(4) *(Final shading)* Use Equation 9 to compute each pixel's color.

Spatiotemporal reuse gives each pixel access to a large population of samples from its local neighborhood. As a result, ReSTIR quickly finds samples that make large contributions to pixels, using MIS weights and shift mappings to ensure unbiased estimation within the pixel where samples are reused. Nonetheless, gains from sharing samples are not indefinite, and correlation artifacts may arise from undersampling, imperfect shift mappings, and wrongly set parameters. For instance, performing multiple rounds of spatial resampling with too small a pixel radius can lead to blotchy artifacts. This happens when RIS identifies too few samples to effectively importance sample the integrand, *e.g.*, due to difficult-to-sample lighting. Likewise, inadequately designed shift mappings may introduce geometric singularities into a sample's resampling weight via the Jacobian determinant, causing the sample to be widely reused.

During temporal resampling, one must cap the stream length M of a temporally reused sample (Algorithm 2, *line 3*) to guarantee convergence—not doing so results in convergence to a wrong result [Lin et al. 2022, Section 6.4]. Unfortunately, the ideal M_{cap} cannot always be determined in a scene-agnostic way—small caps inadequately utilize the temporal history and result in higher variance (Lin et al. [2022, Figure 9]), while large caps increase correlation. In particular, increasing M_{cap} decreases the relative weight and hence selection probability of newly proposed samples, while artificially inflating a reservoir sample's importance. As a result, an outlier reservoir sample's estimated contribution weight must first decay to match a pixel's actual value. Unfortunately, the outlier may be
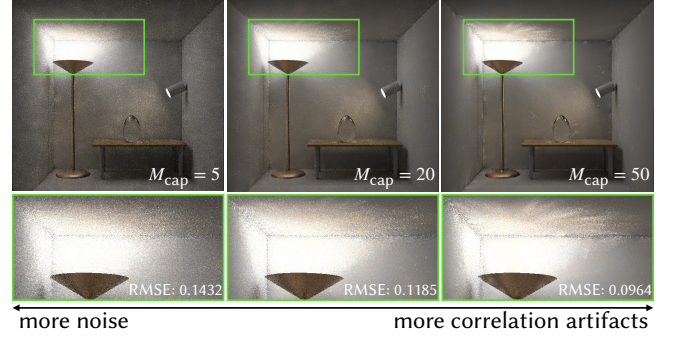

Fig. 4. Parameters for ReSTIR sample reuse can be difficult to set in a scene agnostic way. For instance, a small M_{cap} inadequately exploits prior samples, leading to noise (*left*), while a large M_{cap} value introduces correlations (*right*). Our approach offers greater leeway in setting parameter values that trade noise for correlation (see Figures 11 and 12).

spread between neighboring pixels before it is replaced. This can lead to visible correlation artifacts and sample impoverishment over multiple frames (see Figure 4). We use the Metropolis-Hastings algorithm, described next, to address these issues in ReSTIR.

2.5 Metropolis–Hastings (MH)

Like RIS, the MH [Metropolis et al. 1953; Hastings 1970] algorithm generates a set of samples distributed proportionally to a non-negative and possibly unnormalized target function $\hat{p}$. While RIS uses resampling to achieve this goal, MH instead constructs a *Markov chain* that has a stationary distribution proportional to $\hat{p}$. In more detail, given an initial sample $x^0 \in \Omega$, MH incrementally constructs a sequence of random samples $x^0, x^1, x^2, \ldots$ as follows:

(1) For $k \geq 0$, generate a candidate sample z^k by applying a random *mutation* to the current sample x^k in the chain, *i.e.*, sample z^k from a *proposal density* $T(x^k \rightarrow z^k)$.

(2) Compute an acceptance probability for the candidate z^k:

$$a(x^k \rightarrow z^k) := \min\left(1, \frac{\hat{p}(z^k) \, T(z^k \rightarrow x^k)}{\hat{p}(x^k) \, T(x^k \rightarrow z^k)}\right). \quad (14)$$

(3) Set $x^{k+1} = z^k$ with probability a; otherwise set $x^{k+1} = x^k$.

The acceptance probability $a(x^k \rightarrow z^k)$ ensures that samples are distributed proportional to the target function $\hat{p}$. The *detailed balance* condition guarantees the existence of the Markov chain's stationary distribution by requiring the transition density between any two sample values to be equal:

$$\hat{p}(x^k)T(x^k \rightarrow z^k)a(x^k \rightarrow z^k) = \hat{p}(z^k)T(z^k \rightarrow x^k)a(z^k \rightarrow x^k). \quad (15)$$

To generate the correct distribution from all inputs, Markov chains must be *ergodic*. This can be guaranteed easily with mutations that always propose candidate samples over the entire support of $\hat{p}$, i.e., $T(x^k \rightarrow z^k) > 0$ for all x^k and z^k where $\hat{p}(x^k) > 0$ and $\hat{p}(z^k) > 0$. Even with this constraint, there is still much freedom in choosing mutation strategies—Section 4 describes the strategies we use.
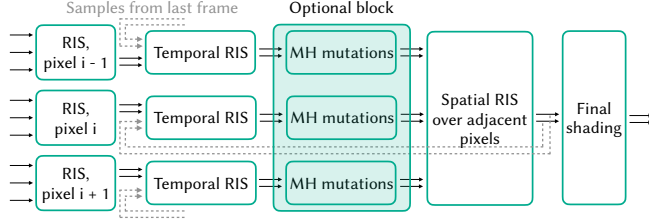
Fig. 5. Our approach introduces Metropolis-Hastings mutations as an additional block into the larger ReSTIR algorithm for spatiotemporal sample reuse. Samples are mutated within each pixel after temporal resampling (Algorithm 2) to mitigate correlation artifacts and sample impoverishment.

Unlike RIS, MH does not estimate the value of integrals. It does however produce valid samples from its target function which can be used by a secondary estimator such as RIS for estimation (Section 3).

Eliminating start-up bias. MH assumes the initial sample x^0 is generated with probability density proportional to $\hat{p}$; using a sample not from this distribution results in *start-up bias*. A typical solution runs the Markov chain for several iterations until the initial state is "forgotten", *i.e.*, discarding several early samples generated by MH. Sadly, the length of this *burn-in* period is tricky to determine as it depends on the initial sample value and its actual distribution. Veach [1998, Chapter 11.3.1] instead proposed resampling x^0 from M candidate samples $\mathbf{y} = \{y_1, \ldots, y_M\}$ generated using an easy-to-sample source PDF (much like Section 2.1). Equations 6 and 7 then provide the discrete probabilities and resampling weights (*resp.*) needed to select a candidate, *i.e.*, $x^0 = y_j$ for some $j \in \{1, \ldots, M\}$. Contributions of mutated samples initialized from x^0 are weighted by Equation 8 to guarantee unbiasedness. Our mutations likewise leverage ReSTIR's built-in resampling to avoid start-up bias.

3 METHOD

Sample selection with RIS from a target distribution improves with larger populations of candidate samples. ReSTIR provides access to a sizable candidate pool for resampling through spatiotemporal reuse, enabling it to quickly identify high-contribution samples via RIS. However, at times ReSTIR extensively reuses a few samples over multiple frames due to imperfect importance sampling and suboptimal parameters, with no mechanism to easily diversify an existing population of high-contribution samples.

Inspired by Sequential and Population Monte Carlo techniques (Section 6), we interleave reservoir resampling with MCMC mutations to mitigate correlations and sample impoverishment caused by spatiotemporal reuse. Our key observation is that mutating reservoir samples with the *same* per-pixel target function as RIS helps to quickly decorrelate the resampled population, especially when it contains outliers. In Algorithm 3, we use Metropolis-Hastings to locally perturb temporal reservoir samples selected by Algorithm 2; interleaving with resampling then diversifies the samples ReSTIR shares between pixels. We discuss key aspects of our work next, starting with how to modify mutated samples' contribution weights to guarantee unbiased results.

ALGORITHM 3: Mutate sample via Metropolis-Hastings

Input: Pixel i, reservoir r_i from Algorithm 2, and iteration count
Output: Reservoir r_i with its sample mutated in proportion to $\hat{p}_i$
1: **function** MUTATESAMPLE(i, r_i, iters)
2:      $z \leftarrow$ METROPOLISHASTINGS($r_i.x$, $\hat{p}_i$, iters)   ▷*Section 2.5*
3:      $r_i.\widehat{W} \leftarrow \frac{\hat{p}_i(r_i.x)}{\hat{p}_i(z)} \cdot r_i.\widehat{W}$ ▷*Equation 16*
4:      $r_i.x \leftarrow z$
5:      **return** r_i

Modified contribution weights. A contribution weight $\widehat{W}$ (Equation 8) estimates the reciprocal value of the target PDF $\hat{p}/\int_\Omega \hat{p}$ that a sample is approximately distributed according to. $\widehat{W}$ is needed to compute resampling weights for combining reservoirs (Algorithm 2, *lines 7* and *11*) and to estimate per-pixel shading (Equation 9).

Contribution weights are sample dependent. Thus, a sample that undergoes mutation cannot reuse the weight associated with its original state, *i.e.*, a mutated sample's contribution weight should provide an unbiased estimate for the sample's reciprocal target PDF. Our key contribution is to show that the unbiased contribution weight for any mutated sample x^k, from a Markov chain $x^0, \ldots, x^k, \ldots$, can be computed via the relation

$$\widehat{W}(x^k) = \frac{\hat{p}(x^0)}{\hat{p}(x^k)} \widehat{W}(x^0). \tag{16}$$

Equation 16 does not depend on samples between x^0 and x^k in the Markov chain and imposes no constraints on computing $\widehat{W}(x^0)$, which can arise from prior resampling, runs of MH, or a mix of the two. This provides flexibility in where and when to mutate samples during ReSTIR (as long as mutations are confined to a given pixel).

One can get an intuitive feel for Equation 16 by substituting in the expression for $\widehat{W}(x^0)$ from Equation 8:

$$\widehat{W}(x^k) = \frac{\cancel{\hat{p}(x^0)}}{\hat{p}(x^k)} \cdot \frac{1}{\cancel{\hat{p}(x^0)}} \left(\sum_{j=1}^{M} w(y_j) \right) = \frac{1}{\hat{p}(x^k)} \left(\sum_{j=1}^{M} w(y_j) \right). \tag{17}$$

Notice that the estimated normalization factor for $\hat{p}$, *i.e.*, the sum of weights w, remains unchanged for both the initial and mutated samples x^0 and x^k. This normalization factor arises via RIS *prior to* performing mutations (*e.g.*, Algorithm 2, *lines 14-15*). Meanwhile, MH treats the resampling weights as fixed, simply redistributing a reservoir's sample population proportionally to the per-pixel target function $\hat{p}$. Equation 16 then encodes any required correction to a sample's contribution weight to account for the sample mutation—unlike temporal and spatial resampling which reuse samples across different pixels, this equation does not contain any MIS weight or shift mapping as there is no change of integration domain, with samples only mutated within $\hat{p}$'s support.

Start-up bias. Algorithm 3 does not require a burn-in period for mutations, even though the samples used to initialize MH are not distributed exactly according to $\hat{p}$. This is because we use the unbiased contribution weights of mutated samples for subsequent steps in ReSTIR, including when computing shading and resampling weights for further reuse. This approach eliminates start-up bias completely

for any mutated sample x^k and function f by ensuring

$$\mathbb{E}[f(x^k)\widehat{W}(x^k)] = \int_\Omega f(x)\, dx. \tag{18}$$

Appendix A provides a formal proof. Note that avoiding start-up bias does not imply samples generated using MH are well-distributed according to $\hat{p}$. However, since we initialize MH using reservoir samples that are already distributed roughly proportional to the target function from resampling, our method does not rely on MH to find important samples (see Figure 13)—rather it decorrelates and diversifies outlier samples by mutating them locally in proportion to $\hat{p}$ and adjusting their estimated contribution weight accordingly.

When to perform mutations? Temporal reservoirs often contain stale samples, as ReSTIR assigns higher relative importance to existing samples. We therefore mutate samples output by Algorithm 2 within each pixel (Figure 5), using the same per-pixel target function as RIS for the current frame. Mutating samples randomly after temporal resampling diversifies the inputs to spatial resampling, protecting against possibly escalating amounts of sample impoverishment caused by repeated reuse.

Applying Algorithm 3 within each pixel to mutate samples after the initial or spatial resampling steps in ReSTIR (Section 2.4) is possible but not required. Like mutations, initial resampling serves to rejuvenate the sample population every frame (by introducing new independent samples into the population). Samples from spatial resampling are stored for future reuse; mutating them proportional to the current target function would cause them to lag by one frame.

Finally, Algorithm 3 places no restrictions on MH iteration count. To improve runtime performance, one could adaptively specify mutation counts per pixel (including no mutations) using, for instance, local correlation estimates. We leave development of such heuristics to future work and use a fixed, user-specified number of iterations.

4 IMPLEMENTATION DETAILS

We perform mutations for both direct and indirect illumination in ReSTIR using Kelemen et al. [2002]'s *primary sample space (PSS)* parameterization. This conveniently allows applying mutations directly to random number sequences used to generate light-carrying paths, while constraining path vertices to remain on the scene manifold. Moreover, it simplifies use of certain shift mappings in ReSTIR PT, *e.g.*, the *random replay* shift [Lin et al. 2022, Section 7.2].

In this section, we represent samples with a path vertex notation $\bar{\mathbf{x}} = [\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_k] \in \Omega^k(\mathcal{M})$, with $\Omega^k(\mathcal{M})$ the space of all paths of length k on the scene manifold $\mathcal{M}$ (*e.g.*, $k = 2$ for direct lighting). Each path $\bar{\mathbf{x}}$ is uniquely determined[2] by a vector of random numbers $\bar{\mathbf{u}} = [u_0, u_1 \ldots] \in [0, 1]^{O(k)}$. We use S to denote a shift mapping from a base path $\bar{\mathbf{x}}$ in one pixel to an offset path $\bar{\mathbf{y}}$ in another pixel, *i.e.*, $S([\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_k]) = [\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_k]$. Mutated paths and random numbers are represented using $\bar{\mathbf{z}}$ and $\bar{\mathbf{v}}$ (*resp.*).
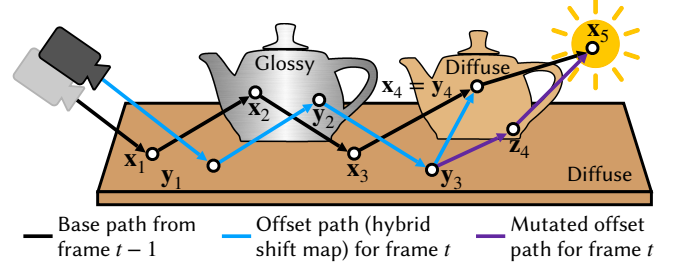

Fig. 6. During temporal resampling, the hybrid shift in ReSTIR PT connects the offset path for frame t to the base path from frame $t - 1$ when it encounters two consecutive diffuse vertices $\mathbf{x}_3, \mathbf{x}_4$; prior to that it reuses random numbers from the base path to trace rays. Our reconnection vertex mutation then perturbs the reconnection vertex $\mathbf{y}_4$ in the offset path.

4.1 Primary sample space

The PSS parameterization reformulates the acceptance probability in Equation 14 in terms of a *contribution function C* as follows:

$$a(\bar{\mathbf{u}} \rightarrow \bar{\mathbf{v}}) := \min\left(1, \frac{C(\bar{\mathbf{v}})\, T(\bar{\mathbf{v}} \rightarrow \bar{\mathbf{u}})}{C(\bar{\mathbf{u}})\, T(\bar{\mathbf{u}} \rightarrow \bar{\mathbf{v}})}\right). \tag{19}$$

For us $C(\bar{\mathbf{u}}) := \hat{p}(\bar{\mathbf{y}}(\bar{\mathbf{u}}))/q(\bar{\mathbf{y}}(\bar{\mathbf{u}}))$, where $\hat{p}$ is the per-pixel target function (also used for resampling) and q is the sampling PDF for generating $\bar{\mathbf{y}}$ from the random numbers $\bar{\mathbf{u}}$ [3] (with mutated path $\bar{\mathbf{z}}$ likewise generated from $\bar{\mathbf{v}}$). As suggested by Kelemen et al. [2002], we compute $\bar{\mathbf{v}}$ by perturbing each element of $\bar{\mathbf{u}}$ with Gaussian noise. We use $s = s_2 \exp(-\log(s_2/s_1)U)$ as our perturbation amount with $U \sim [0, 1)$ and $s \in (s_1, s_2)$.

4.2 Direct Lighting

Our ReSTIR DI mutations perturb the directions of reservoir samples via their random numbers. For direct lighting, path $\bar{\mathbf{y}} = [\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2]$ and its PDF $q(\bar{\mathbf{y}})$ equals $p_\rho(\omega)|\cos\theta|/|\mathbf{y}_2 - \mathbf{y}_1|^2$, where p_ρ is the PDF for importance sampling the BSDF ρ, ω is the unit vector from $\mathbf{y}_1$ to $\mathbf{y}_2$, and θ is the angle between ω and the geometric surface normal at $\mathbf{y}_2$. The PDF $q(\bar{\mathbf{z}})$ is defined analogously as $p_\rho(v)|\cos\phi|/|\mathbf{z}_2 - \mathbf{y}_1|^2$ with direction v pointing from $\mathbf{y}_1$ to mutated vertex $\mathbf{z}_2$; ϕ is the angle between v and the surface normal at $\mathbf{z}_2$. Random numbers for the starting MH sample $\mathbf{y}_2$ are recovered by inverting the sampling procedure for direction ω [Bitterli et al. 2017]. Since this mutation is symmetric, the transition kernels in Equation 19 cancel.

4.3 Indirect Illumination

For ReSTIR PT, our mutation strategies build on shift maps. Unlike a mutation, a shift mapping deterministically perturbs a base path $\bar{\mathbf{x}}$ through one pixel into an offset path $\bar{\mathbf{y}}$ through another pixel *during* resampling (*e.g.*, Algorithm 2, *line 9*). For instance, a *random replay (RR)* shift reuses the random numbers that generate $\bar{\mathbf{x}}$ to trace $\bar{\mathbf{y}}$. Since tracing a full path is expensive, a reconnection is often used to connect the offset path to the base path at a given index i, *i.e.*, $\mathbf{y}_j = \mathbf{x}_j$ for $j \geq i$. Connecting paths immediately with $i = 2$ is called the

---

[2] As in Bitterli et al. [2017], we bijectively map between paths and their random numbers by padding paths with extra dimensions.

[3] The starting unmutated path $\bar{\mathbf{y}}$ for MH could have been generated in ReSTIR from one of many sampling schemes (*e.g.*, light or BSDF sampling), or over multiple rounds of resampling. Here, we do not require the random numbers $\bar{\mathbf{u}}$ that originally generated $\bar{\mathbf{y}}$; Sections 4.2 and 4.3 discuss the $\bar{\mathbf{u}}$ we use for mutations.

reconnection (R) shift. Compared to random replay, reconnections are often better at producing paths with similar contributions for diffuse surfaces. But reconnecting y_{i-2}, y_{i-1} to x_i on a glossy surface can introduce paths with near-zero throughput, or introduce geometric singularities when y_{i-1} and x_i are too close.

We use Lin et al.'s [2022] *hybrid (H)* shift strategy (see Figure 6) to evaluate mutations in ReSTIR PT for all our results except Figure 8 where we use random replay. This shift mapping postpones reconnection using random replay until certain connectability conditions are met (*e.g.*, surface roughness and distance between vertices).

Mutation strategies. As with direct lighting, one way to mutate a path is to perturb the random numbers used to generate it. Like a random replay shift, this approach expensively requires tracing a full path for each proposed mutation (which may be rejected). We refer to this mutation as a *full path (FP)* mutation.

A more computationally efficient approach mutates the offset path with random replay up to the reconnection vertex $y_i = x_i$, and then connects to the base path starting at x_{i+1} instead (the full path is mutated if a reconnection is not possible). We observe that this *partial path (PP)* mutation strategy is not only faster, but also has higher acceptance (70% vs. 40% on the scene from Figure 1) as it minimizes changes to the geometry of high-contribution paths selected via resampling. Moreover, its paths have similar contributions to the offset paths it mutates. Note that mutating path vertices with random replay until the reconnection to x_{i+1} can cause connectability conditions for the hybrid shift to fail. We reject such mutated samples by defining their transition PDF to be 0.

Taking a step further, our final strategy mutates only the reconnection vertex y_i (Figure 6) while keeping the rest of the offset path unchanged, i.e., $[z_0, \ldots, z_k] = [y_0, y_1, \ldots, y_{i-1}, z_i, x_{i+1}, \ldots, x_k]$, where y_{i-1} connects to z_i with mutated random numbers. We found this *reconnection vertex (RV)* mutation only slightly less effective at reducing correlations. It is, however, significantly faster when performing multiple mutations, as only rays from y_{i-1} to z_i and z_i to x_{i+1} need to be traced. We use this mutation to generate results in Section 5, unless noted otherwise. Figure 10 compares the effectiveness of these mutation strategies.

Finally, note that the transition kernels $T(\bar{v} \rightarrow \bar{u})$ and $T(\bar{u} \rightarrow \bar{v})$ are no longer symmetric when offset paths contain a reconnection vertex. In Appendix C, we show that their ratio equals:

$$\frac{T(\bar{v} \rightarrow \bar{u})}{T(\bar{u} \rightarrow \bar{v})} = \frac{|\cos \phi|}{|\cos \theta|} \frac{|y_{i+1} - y_i|^2}{|y_{i+1} - z_i|^2} \frac{p(v_{i-1}, v_i)}{p(\omega_{i-1}, \omega_i)} \frac{p(v_i, \omega_{i+1})}{p(\omega_i, \omega_{i+1})}, \quad (20)$$

where ω_{i-1}, ω_i and ω_{i+1} are unit vectors from y_{i-1} to y_i, y_i to $y_{i+1} (= x_{i+1})$ and y_{i+1} to $y_{i+2} (= x_{i+2})$ respectively, v_{i-1} and v_i are unit vectors from y_{i-1} to z_i and z_i to y_{i+1}, θ is the angle between ω_i and the surface normal at y_{i+1}, ϕ is the angle between v_i and the surface normal at y_{i+1}, and p is the solid angle PDF used to sample an outgoing direction. Any mutations applied to random numbers for the subpath $[y_0, y_1, \ldots, y_{i-1}]$ do not factor in the ratio as they are symmetric.

Reservoir storage. Lin et al. [2022, Section 8.2] note ReSTIR PT stores additional data in the reservoir from Algorithm 1, specifically a seed for random replay and the resampled path's reconnection vertex. For the full and partial path mutation strategies, we need
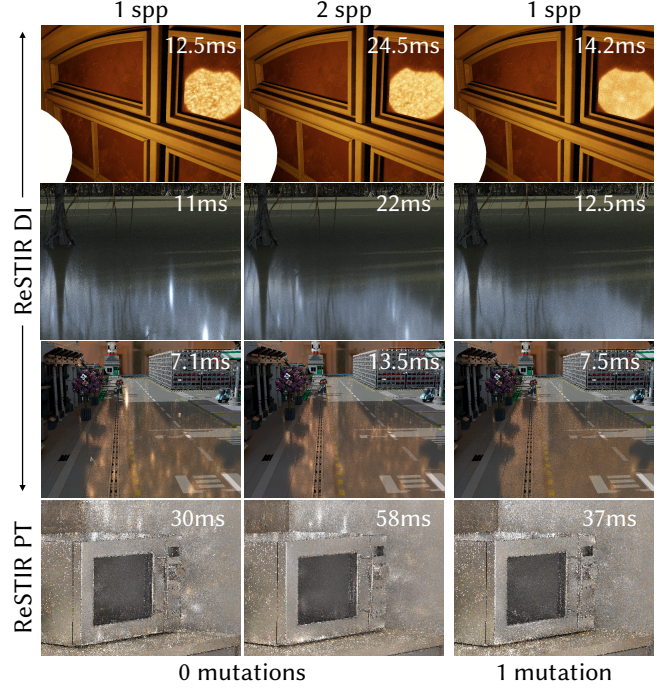

Fig. 7. Correlation artifacts often do not disappear simply by using more samples, justifying the overhead of performing mutations.

the path's entire random number sequence since PSS mutations transform this sequence—as a result, the sequence cannot be regenerated from its original seed. This increases the reservoir size as path length grows. Luckily, the reconnection vertex mutation avoids this overhead, only mutating random numbers that sample z_i from the fixed offset vertex y_{i-1}. As in ReSTIR DI, we recover random numbers for y_i by inverting the sampling of direction $y_i - y_{i-1}$. The only additional information we store is the offset vertex y_{i+1} (which connects to mutated vertex z_i).

5 RESULTS AND DISCUSSION

We prototyped our method in the open-source Falcor rendering framework [Kallweit et al. 2022]. All results use a GeForce RTX 3090 GPU at 1920×1080 resolution. Our direct lighting implementation uses the same settings as Bitterli et al. [2020], *i.e.*, initial candidate samples $M = 32$, spatial reuse radius of 30 pixels from the current pixel, and $M_{cap} = 20$. For indirect illumination, we set $M = 32$ and the spatial reuse radius to 20 similar to Lin et al. [2022], but use a longer temporal history with $M_{cap} = 50$ (unless noted otherwise). Our supplementary video shows 1 spp results for all our scenes; Table 1 gives single frame timings.

As we show in Figure 1 and Figures 7–12, short-range correlation artifacts are noticeably reduced in scenes with glossy materials and difficult lighting with just 1–5 mutations; further mutations have diminishing returns in improving image quality (Figures 9 and 10). Mutation cost overhead is generally less than simply increasing sample count (Figure 7), and recent denoisers [NVIDIA 2017] provide
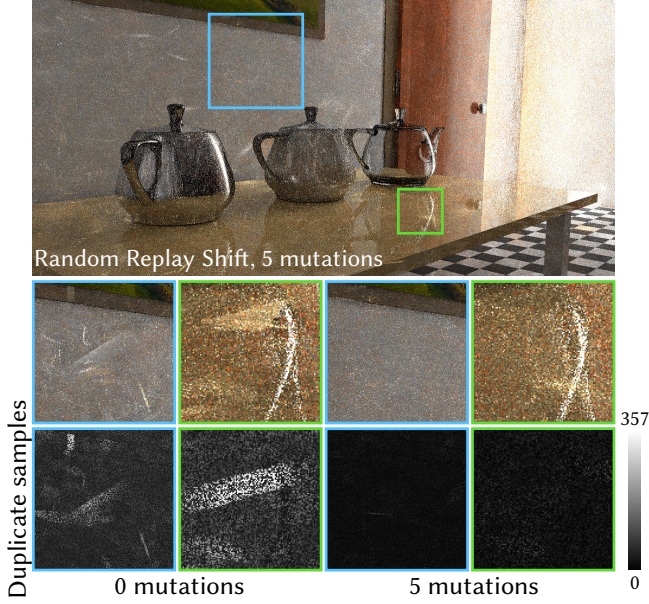
Fig. 8. Mutations mitigate sample impoverishment in ReSTIR by diversifying the sample population. The bottom row visualizes duplicate samples in 20×20 pixel neighborhoods on the scene from Figure 3. The supplemental video shows improvements in an animation as the door's angle is decreased.
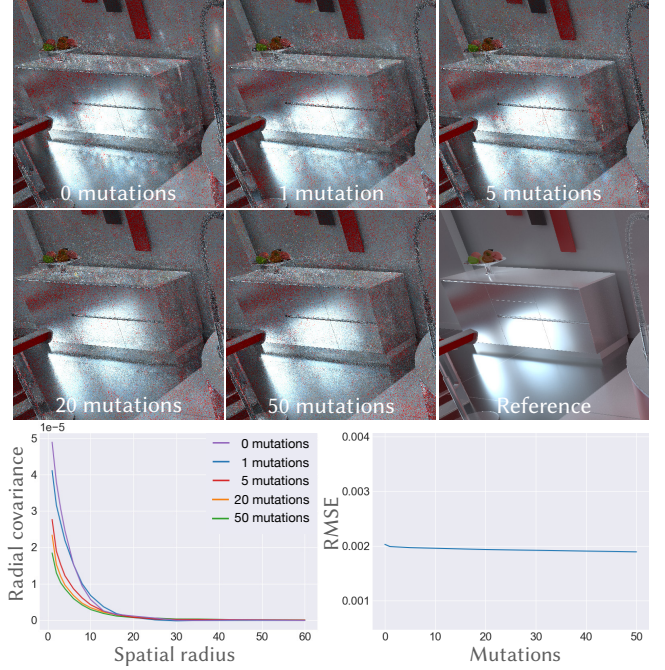

Fig. 9. Sample mutations reduce short range correlation artifacts produced by ReSTIR, with even 1-5 mutations providing noticeable improvements in image quality (measured in the bottom left using average radial covariance). Mutations typically have little impact on mean squared error (shown in the bottom right at equal spp), as we perturb samples only within each pixel.
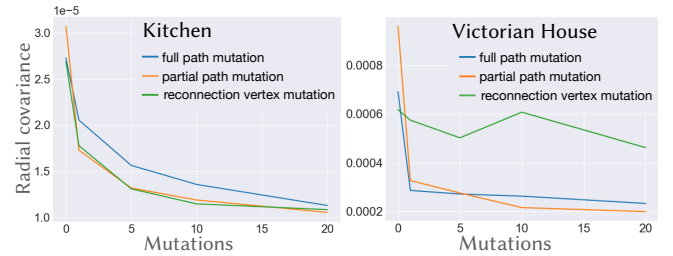

Fig. 10. Reduction in covariance depends on the mutation strategy, as in any MCMC technique. *Left:* On the Kitchen (Figure 9), we get smaller covariance using the hybrid shift with partial path or reconnection vertex mutations as they minimize changes to paths selected by resampling. *Right:* In the Victorian house (Figure 12, *bottom row*), reconnection vertex mutations are less effective as fewer paths are reconnected due to a lack of consecutive diffuse vertices, whereas partial path mutations just perturb the entire path.

considerably better results with our decorrelated samples (Figure 1). Figure 8 shows mutations greatly reduce sample impoverishment, with fewer reservoirs sharing the exact same sample realizations.

Compared to standard path tracing, ReSTIR is much faster at achieving equal-error via correlated sampling for real-time direct [Bitterli et al. 2020, Figure 8] and global illumination [Lin et al. 2022, Figure 13]. Mutations however provide only marginal improvements in mean squared error in ReSTIR samplers (Figures 9 and 13), without ever negatively impacting results. Akin to blue-noise dithering [Georgiev and Fajardo 2016; Heitz and Belcour 2019], our image quality improves despite errors having similar magnitudes. The reason is mutating within a pixel leaves the sum of resampling weight unchanged in Equation 17, and these weights ultimately control RIS estimator variance (Equation 9). Mutations do slightly reduce variance, as they indirectly alter resampling weights of *future* samples thanks to spatiotemporal reuse of the new, more diverse sample population; the supplementary document has more details. In Figures 11 and 12 we also ablate M_{cap} values to show the greater leeway our approach offers for this parameter, allowing use of larger values to trade noise for correlation.

Since ReSTIR often suffers from correlation artifacts, we quantify improvements in correlation by computing sample covariance between pixels, which naturally generalizes sample variance. This metric measures the joint variability of two random variables (*e.g.*, whether error in two pixels varies similarly). For pixels i and j in image I, sample covariance $\text{Cov}(i, j)$ between i and j is given by

$$\text{Cov}(i, j) = \frac{1}{K-1} \sum_{k=1}^{K} \left(I_{ki} - \bar{I}_i \right) \left(I_{kj} - \bar{I}_j \right), \qquad (21)$$

where K is the number of images used to estimate covariance (we use $K = 100$), and $\bar{I}$ is the average of K images. To capture the joint variability of a pixel with its local neighborhood, in our experiments we average covariance estimates over boxes of a given radius centered at each pixel. We then further average over the entire image to get a single number. Figure 9 (bottom left) shows average radial covariance decreases with increasing spatial radius. This is expected
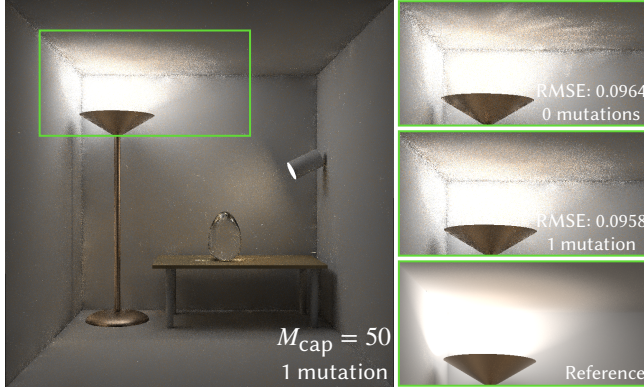
Fig. 11. By reducing correlation artifacts, mutations allow for use of larger M_{cap} values in ReSTIR to trade noise for correlation, yielding lower error in scenes with difficult to sample light-carrying paths (see Figure 4 for results with smaller M_{cap} values).

Fig. 12. ReSTIR PT excessively reuses a few paths and suffers from "boiling" in scenes with difficult-to-sample indirect lighting such as the Victorian house where paths must be reflected in through the windows (*top left*). Our mutations do not resolve this boiling by finding better paths (*top right*), but increasing M_{cap} does due to greater temporal reuse (*bottom left*). Mutations mitigate correlation artifacts caused by a large M_{cap} (*bottom right*).

Fig. 13. Mutations typically do not reduce mean squared error, shown here on the Veach Ajar scene rendered using ReSTIR PT with the hybrid shift. This suggests that unlike Metropolis Light Transport, resampling (and not mutations) finds important light-carrying paths in ReSTIR. Compared to the random replay shift in Figure 8, resampling with the superior hybrid shift also introduces fewer correlations in this scene for mutations to resolve.

as ReSTIR only reuses samples in local neighborhoods (so small-scale correlation artifacts are more pronounced); mutations reduce covariance in these short ranges.

Table 1 lists the reduction in average covariance (with pixel radius equal to 8) and the FLIP weighted median score [Andersson et al. 2020] we observe on our scenes—FLIP tends to be marginally more sensitive to short range correlations than mean squared error. As correlations are typically localized, the reduction is even larger for the image insets in our figures compared to the results in Table 1. Ineffective shift mappings in ReSTIR often result in increased correlations; mutations compensate for this shortcoming. For instance, mutations typically have fewer correlation artifacts to resolve with a hybrid shift in ReSTIR PT compared to, *e.g.*, random replay (Figures 13 and 8 respectively), which highlights the benefit of using good shift mappings. In contrast, mutations provide greater covariance reduction in the scenes rendered with ReSTIR DI (Figure 7), where higher covariance stems from vertex reconnections failing to preserve path contributions for low roughness surfaces.

Why mutations help? The supplemental document details why mutations reduce covariance, simplifying down to the following, somewhat unintuitive, phenomenon: without mutations, covariance between pixels i and j stems from the mismatch between the distributions of input samples and the target functions at i and j (Equation 16 in the supplemental). However, in the limit of infinite mutations, covariance is determined by each sample's mismatch with its own pixel's target function (Equation 10 in the supplemental) due to the ratio $\hat{p}(x^0)/\hat{p}(x^k)$ in the mutated contribution weight (Equation 16); this mismatch tends to be smaller. Though our analysis predicts that covariance does not vanish completely even with infinite mutations, our results show covariance is often reduced with just one mutation.

6 RELATED WORK

Our method builds directly on the recent ReSTIR family of algorithms for real-time direct [Bitterli et al. 2020] and global illumination [Ouyang et al. 2021; Lin et al. 2021, 2022]. We augment
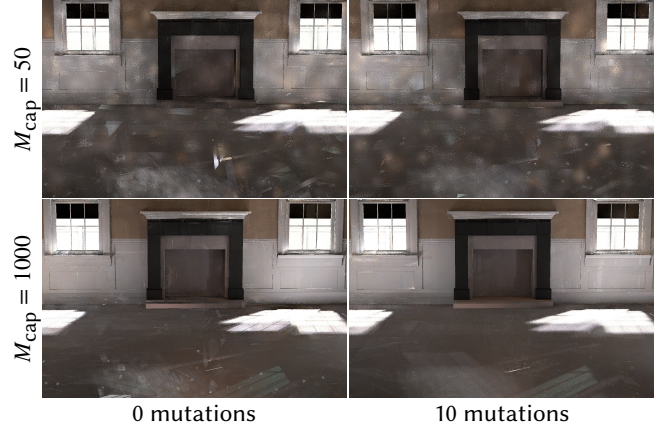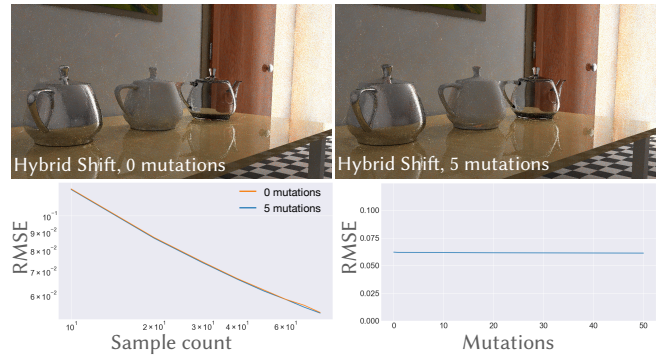
spatiotemporal reservoir resampling in ReSTIR with sample mutations, and demonstrate the complementary strengths of resampling and mutations in this framework. In graphics, our approach is most closely related to Metropolis Light Transport (MLT) [Veach and Guibas 1997] and associated techniques [Kelemen et al. 2002; Jakob and Marschner 2012; Lehtinen et al. 2013; Hachisuka et al. 2014; Otsu et al. 2018; Cline et al. 2005; Lai et al. 2007, 2009; Bashford-Rogers et al. 2021]. In the broader Monte Carlo landscape, our approach belongs to the class of algorithms that jointly use resampling and mutations for sampling problems, such as Sequential Monte Carlo (SMC) [Doucet et al. 2001] and Population Monte Carlo (PMC) [Cappé et al. 2004]. We discuss the relation to MLT, SMC and PMC in more detail next; Table 2 provides a summary. We refer the reader

Scene	Method ReSTIR/Shift/Mutation	Relative Cov/FLIP 1 vs. 0 mutations	Relative Cov/FLIP 5 vs. 0 mutations	Time (ms) 0 mutations	Time (ms) 1 mutation	Time (ms) 5 mutations
Bistro Exterior (Fig. 7)	DI / - /BSDF	0.386 / 0.928	0.258 / 0.894	12.5	14.2	18.5
Forest (Fig. 7)	DI / - / BSDF	0.060 / 0.909	0.017 / 0.871	11	12.5	16
Lego city (Fig. 7)	DI / - / BSDF	0.039 / 0.919	0.027 / 0.905	7.1	7.5	8.6
Kitchen (Fig. 1)	PT / H / RV	0.824 / 0.956	0.733 / 0.925	30	37	43
Kitchen (Fig. 2)	PT / H / RV	0.764 / 0.977	0.697 / 0.968	32	39	47
Veach Egg (Fig. 10)	PT / H / RV	0.737 / 0.886	0.668 / 0.874	11	15	17
Veach Ajar (Fig. 13)	PT / H / RV	0.922 / 0.988	0.881 / 0.977	26	33	37
Veach Ajar (Fig. 8)	PT / RR / FP	0.692 / 0.964	0.608 / 0.953	44	64	90
Victorian House (Fig. 12)	PT / H / RV	0.934 / 0.953	0.817 / 0.930	34	40	46
Victorian House (Fig. 12)	PT / H / PP	0.340 / 0.879	0.287 / 0.848	34	58	78

Table 1. Relative reduction in covariance and FLIP scores from mutations at 1 spp. We first average covariances over boxes of pixel radius 8, and then over the entire image. The ReSTIR DI scenes use $M_{\text{cap}} = 20$ while the ReSTIR PT scenes use $M_{\text{cap}} = 50$, except for the Victorian house where $M_{\text{cap}} = 1000$.

to Bitterli et al. [2020, Section 7] and Lin et al. [2022, Section 9.3] for comparisons between ReSTIR and other rendering algorithms that exploit path reuse and spatial correlations.

Metropolis Light Transport. MLT uses statistically correlated samples generated by Metropolis–Hastings to solve the rendering equation. Unlike algorithms using independent samples, MLT is effective at finding difficult light paths by locally exploring the path space. It reuses samples by mutating high-contribution paths over the image. Algorithmically, our method resembles MLT in various ways. Both techniques require secondary estimators, respectively RIS and bidirectional path tracing (BDPT) [Lafortune and Willems 1993; Veach and Guibas 1995a], to normalize the MH target function. Samples used by these estimators are resampled into a smaller set to initialize MH (our Section 3 and Veach [1998, Chapter 11.3.1]), and contributions of mutated samples are effectively weighted by the same weights (Equation 17) to remain unbiased (our Appendix A and Veach [1998, Appendix 11.A]).

The crucial difference between our work and MLT lies in how samples are reused across pixels. MLT latches onto high-contribution paths and mutates them over the entire image while resampling just to eliminate start-up bias. Thus, MLT results often contain correlation artifacts caused by mutations, applying MH to both find important samples and redistribute them between pixels. In contrast, ReSTIR derives spatiotemporal reuse *exclusively* from resampling; in this paper, we mutate samples within each pixel to mitigate correlations and sample impoverishment from spatiotemporal resampling. As a result, our method does not require numerous MH iterations, as the primary purpose of mutations is not finding important paths (Figures 12 and 13). Further, our approach suits real-time rendering as it integrates seamlessly into ReSTIR. MLT can be adapted to mutate temporally, but unlike our work, the entire animated sequence must be available in advance [Van de Woestijne et al. 2017].

Several features have recently been added to MLT, including sample stratification [Cline et al. 2005; Gruson et al. 2020], MIS [Hachisuka et al. 2014] and enhanced mutation strategies [Jakob and Marschner 2012; Kaplanyan et al. 2014; Bitterli et al. 2017; Otsu
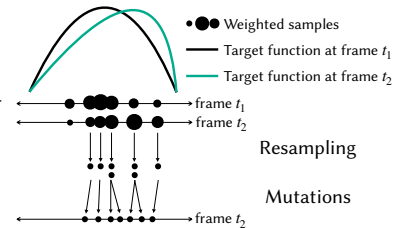
et al. 2018; Luan et al. 2020]. Though we mostly employ simple PSS mutations [Kelemen et al. 2002], many of these improvements can also be incorporated into our approach to reduce correlations faster.

Sequential Monte Carlo. SMC is a family of Monte Carlo methods used for filtering and tracking problems in Bayesian inference and signal processing [Doucet et al. 2001]. As shown in the inset, the

goal is maintaining a population of weighted samples distributed roughly proportional to an evolving target distribution (with unknown normalization factor). Sample weights are adjusted every iteration to reflect each sample's importance to the most recent distribution. Resampling discards samples with low weights and duplicates those with high weights. Mutations ensure the population does not contain identical samples. Unlike ReSTIR, which uses RIS, SMC methods use *weighted importance sampling (WIS)* to estimate correlated integrals in a chained fashion, *i.e.*, the current step's sample weights and normalization factors are defined incrementally based on corresponding quantities from earlier steps [Del Moral et al. 2006]. This allows temporally reusing samples for estimation, instead of generating new samples every frame.

SMC methods have found limited use in rendering: Hedman et al. [2016] maintain a temporally coherent distribution of VPLs for indirect illumination via a sampling method that moves as few of the VPLs between frames as possible. This work is only loosely inspired by SMC, in that it does not perform any sample mutations and generates biased results as it discards VPLs heuristically.

Ghosh et al. [2006] instead sample a sequence of per-pixel target functions for direct illumination of dynamic environment maps. Unlike ReSTIR, which derives its samples from spatiotemporally neighboring pixels, Ghosh et al. [2006] maintain a fixed sample population per pixel that is resampled and mutated to be updated for each frame. Large populations are needed for effective importance

	MLT	PMC (ERPT)	SMC	ReSTIR (Ours)
Estimator	Standard Monte Carlo	Standard Monte Carlo	Weighted Importance Sampling	Resampled Importance Sampling
Contribution weights	Estimate integrated luminance using BDPT* (value is constant over image)	Estimate integrated luminance using BDPT* (value is constant over image)	Estimate per-pixel target distribution for current frame Defined incrementally using weights from previous frame	Estimate per-pixel reciprocal target distribution for current frame Defined incrementally using weights from neighboring (spatial and temporal) pixels
Role of resampling	Eliminate start-up bias in MH	Introduce new samples into population each frame, discard low-contribution samples Eliminate start-up bias in MH	Temporal reuse of sample population within a pixel Discard low-contribution samples Eliminate start-up bias in MH	Spatiotemporal reuse of sample populations across pixels Introduce new samples into population each frame, discard low-contribution samples Eliminate start-up bias in MH
Role of mutations	Share samples spatially across the image*	Share samples spatially across the image*	Mitigate sample impoverishment due to resampling	Mitigate correlations and sample impoverishment due to resampling

Sample "reuse" derived from mutations ←————————————→ Sample reuse derived from resampling

*For animated sequences, luminance is integrated over all images, and samples are mutated spatiotemporally

Table 2. Overview of the role of resampling and mutations in MLT, PMC, SMC and ReSTIR.

sampling, as high-contribution samples are not shared spatially between pixels; in contrast, ReSTIR often stores just a single sample per reservoir. SMC methods likely require MIS weights and shift mappings (like ReSTIR) to resolve bias and correctly derive effective spatiotemporal reuse from neighbors. Similar to our work, mutations mitigate sample impoverishment but do not provide reuse.

Population Monte Carlo. PMC methods also couple resampling and mutations to distribute weighted samples in proportion to a sequence of target functions [Cappé et al. 2004]. The main added feature is they sample using parametric mixture models with simple source PDFs. Mixture probabilities are tuned for each target function using previously generated samples and their importance.

In rendering, the PMC framework has been used for direct lighting [Fan et al. 2007; Lai et al. 2015], global illumination [Lai and Dyer 2007; Lai et al. 2007] and animation [Lai et al. 2009]. Lai et al.'s [2009] work is most relevant to ours: they derive sample reuse by mutating samples spatially and temporally across the image plane using Energy Redistribution Path Tracing (ERPT) [Cline et al. 2005]. Resampling serves to select high-contribution samples while discarding those with small weights; it is also used to refresh the sample population (much like initial resampling in ReSTIR) and eliminate start-up bias from mutations. Unlike our method, they require knowing animated sequences in advance, precluding most real-time applications.

7 LIMITATIONS AND FUTURE WORK

In this paper, we provide an unbiased mechanism leveraging MCMC mutations to diversify ReSTIR's sample population. Often, just a single mutation per pixel effectively mitigates correlation artifacts

in glossy scenes with complex lighting. However, as in most MCMC schemes, we cannot accurately predict the number of Metropolis–Hastings iterations needed to reduce correlations below a given threshold. Beyond the analysis in the supplemental document, further investigation is also needed to understand how mutations address sample impoverishment in ReSTIR—not just in terms of the number of duplicate samples (Figure 8), but also the discrepancy characteristics of the resulting sample population.

Mutations in ReSTIR have a non-negligible run-time overhead. Though we demonstrate improvements on an equal-time covariance metric with simple mutation strategies in both ReSTIR DI and PT (Figure 7 and Table 1), more sophisticated mutations [Jakob and Marschner 2012; Kaplanyan et al. 2014; Bitterli et al. 2017; Otsu et al. 2018; Luan et al. 2020] could provide further gains. Our decision to mutate only after temporal (but not spatial) resampling is informed in part by run-time considerations. As mentioned in Section 3, applying mutations selectively (*i.e.*, not at each pixel every frame) based on local correlation heuristics could improve performance. As both mutations and ReSTIR's initial path candidates serve to rejuvenate the sample population, it may also be worthwhile to carefully balance costs of per-pixel mutations and new path candidates.

Our proposed sample mutations reduce correlation between nearby pixels, leading to an error distribution (likely) closer to white noise. But blue noise error distributions are often superior with respect to human perception [Mitchell 1987]; perhaps our mutations could be modified to more directly optimize for blue noise characteristics. For example, when deciding mutation acceptance, we might consider both the target function and the neighboring pixel samples, preferring mutations that introduce differing sample values. A further

improvement might apply the insights of Heitz and Belcour [2019] to optimize the image-space distribution of error rather than solely considering the sample values.

Like Metropolis Light Transport, mutating samples across pixels potentially unlocks further amortization by sharing samples over the entire image (*e.g.*, using the *expected values* technique [Veach 1998, Section 11.5]). We leave such "cross-pixel" mutations to future work as the change of integration domain from one pixel to another requires a further adjustment to a mutated sample's contribution weight beyond Equation 16, *i.e.*, as with temporal and spatial resampling, MIS weights and shift mappings are likely needed to correctly mutate a sample in proportion to the target function of a *neighboring* pixel whose support does not coincide entirely with its own. Alternatively, it is possible that the replica exchange approach adopted by Gruson et al. [2020] enables swapping of independent Markov chains from separate pixels in our framework as well.

Our mathematical presentation of resampling for ReSTIR in Section 2 is essentially unchanged from that of Lin et al. [2021] for fast volume rendering. Consequently, extending their approach to incorporate mutations from Section 3 likely requires no further modifications to the framework for volumes beyond the specific choice of mutation strategies, such as the scattering and propagation perturbations proposed by Pauly et al. [2000, Section 5.1].

More generally, by augmenting ReSTIR with mutations, our work establishes a closer correspondence between the RIS-based resampling techniques developed in graphics, and those in the broader statistics literature such as SMC and PMC. In particular, our approach stands to benefit from techniques such as annealed importance sampling [Neal 2001] used in SMC to reduce variance in the resampling weights [Ghosh et al. 2006, Section 4], as well as from adaptation strategies for mutation kernels developed in PMC to increase acceptance rates [Lai et al. 2007, Section 4.2]. Moreover, as in these fields, mutations in ReSTIR open the door not just to artifact-free integration (of the rendering equation), but also to tracking and filtering problems—for instance using well-distributed sample populations generated by our approach as training data for path guiding [Müller et al. 2017; Müller et al. 2019].

ACKNOWLEDGMENTS

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 (aug 2020), 23 pages. https://doi.org/10.1145/3406183

Thomas Bashford-Rogers, Luís Paulo Santos, Demetris Marnerides, and Kurt Debattista. 2021. Ensemble Metropolis Light Transport. *ACM Transactions on Graphics (TOG)* 41, 1 (2021), 1–15.

Benedikt Bitterli. 2022. *Correlations and Reuse for Fast and Accurate Physically Based Light Transport.* Vol. 77. Dartmouth College Ph.D Dissertations. https://digitalcommons.dartmouth.edu/dissertations/77

Benedikt Bitterli, Wenzel Jakob, Jan Novák, and Wojciech Jarosz. 2017. Reversible jump Metropolis light transport using inverse mappings. *ACM Transactions on Graphics (TOG)* 37, 1 (2017), 1–12.

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 (TOG)* 39, 4 (2020), 148–1.

Olivier Cappé, Arnaud Guillin, Jean-Michel Marin, and Christian P Robert. 2004. Population Monte Carlo. *Journal of Computational and Graphical Statistics* 13, 4 (2004), 907–929.

Chakravarty R Alla Chaitanya, Anton S Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. 2017. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–12.

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

David Cline, Justin Talbot, and Parris Egbert. 2005. Energy redistribution path tracing. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 1186–1195.

Carsten Dachsbacher, Jaroslav Křivánek, Miloš Hašan, Adam Arbree, Bruce Walter, and Jan Novák. 2014. Scalable realistic rendering with many-light methods. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 88–104.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. 2006. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68, 3 (2006), 411–436.

Arnaud Doucet, Nando De Freitas, Neil James Gordon, et al. 2001. *Sequential Monte Carlo Methods in Practice.* Vol. 1. Springer.

ShaoHua Fan, Yu-Chi Lai, Stephen Chenney, and Charles Dyer. 2007. *Population Monte Carlo Samplers for Rendering.* Technical Report. University of Wisconsin-Madison Department of Computer Sciences.

Iliyan Georgiev and Marcos Fajardo. 2016. Blue-noise dithered sampling. In *ACM SIGGRAPH 2016 Talks.* 1–1.

Abhijeet Ghosh, Arnaud Doucet, and Wolfgang Heidrich. 2006. Sequential Sampling for Dynamic Environment Map Illumination.. In *Rendering Techniques.* 115–126.

Adrien Gruson, Rex West, and Toshiya Hachisuka. 2020. Stratified Markov Chain Monte Carlo Light Transport. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 351–362.

Toshiya Hachisuka and Henrik Wann Jensen. 2009. Stochastic progressive photon mapping. In *ACM SIGGRAPH Asia 2009 papers.* 1–8.

Toshiya Hachisuka, Anton S Kaplanyan, and Carsten Dachsbacher. 2014. Multiplexed Metropolis light transport. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–10.

W Keith Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. (1970).

Peter Hedman, Tero Karras, and Jaakko Lehtinen. 2016. Sequential monte carlo instant radiosity. In *Proceedings of the 20th ACM SIGGRAPH symposium on interactive 3D graphics and games.* 121–128.

Eric Heitz and Laurent Belcour. 2019. Distributing Monte Carlo errors as a blue noise in screen space by permuting pixel seeds between frames. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 149–158.

Binh-Son Hua, Adrien Gruson, Victor Petitjean, Matthias Zwicker, Derek Nowrouzezahrai, Elmar Eisemann, and Toshiya Hachisuka. 2019. A Survey on Gradient-Domain Rendering. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 455–472.

Wenzel Jakob and Steve Marschner. 2012. Manifold exploration: A Markov chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–13.

Henrik Wann Jensen. 1996. Global illumination using photon maps. In *Eurographics workshop on Rendering techniques.* Springer, 21–30.

James T Kajiya. 1986. The rendering equation. In *Proceedings of The 13th Annual Conference on Computer Graphics and Interactive Techniques.* 143–150.

Simon Kallweit, Petrik Clarberg, Craig Kolb, Tom'aš Davidovič, Kai-Hwa Yao, Theresa Foley, Yong He, Lifan Wu, Lucy Chen, Tomas Akenine-Möller, Chris Wyman, Cyril Crassin, and Nir Benty. 2022. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor https://github.com/NVIDIAGameWorks/Falcor.

Anton S Kaplanyan, Johannes Hanika, and Carsten Dachsbacher. 2014. The natural-constraint representation of the path space for efficient light transport simulation. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–13.

Csaba Kelemen, László Szirmay-Kalos, György Antal, and Ferenc Csonka. 2002. A simple and robust mutation strategy for the Metropolis light transport algorithm. In *Computer Graphics Forum*, Vol. 21. Wiley Online Library, 531–540.

Alexander Keller. 1997. Instant Radiosity. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97).* ACM Press/Addison-Wesley Publishing Co., USA, 49–56. https://doi.org/10.1145/258734.258769

Emmett Kilgariff, Henry Moreton, Nick Stam, and Brandon Bell. 2018. NVIDIA Turing Architecture In-Depth. *URL: https://devblogs. nvidia. com/nvidia-turing-architecture-indepth/(visited on 2020-05-11)* (2018).

Pawel Kozlowski and Tim Cheblokov. 2021. ReLAX: A Denoiser Tailored to Work with the ReSTIR Algorithm. *GPU Technology Conference* (2021).

Eric P. Lafortune and Yves D. Willems. 1993. Bi-Directional Path Tracing. In *Proceedings of 3rd International Conference on Computational Graphics and Visualization Techniques.* 145–153.

Yu-Chi Lai, Hsuan-Ting Chou, Kuo-Wei Chen, and Shaohua Fan. 2015. Robust and efficient adaptive direct lighting estimation. *The Visual Computer* 31, 1 (2015), 83–91.

Yu-Chi Lai and Charles Dyer. 2007. *Population Monte Carlo Path Tracing*. Technical Report. University of Wisconsin-Madison Department of Computer Sciences.

Yu-Chi Lai, Shao Hua Fan, Stephen Chenney, and Charcle Dyer. 2007. Photorealistic image rendering with population Monte Carlo energy redistribution. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*. 287–295.

Yu-Chi Lai, Feng Liu, and Charles Dyer. 2009. *Physically-based animation rendering with Markov chain Monte Carlo*. Technical Report. University of Wisconsin-Madison Department of Computer Sciences.

Jaakko Lehtinen, Tero Karras, Samuli Laine, Miika Aittala, Frédo Durand, and Timo Aila. 2013. Gradient-domain Metropolis light transport. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–12.

Daqi Lin, Markus Kettunen, Benedikt Bitterli, Jacopo Pantaleoni, Cem Yuksel, and Chris Wyman. 2022. Generalized Resampled Importance Sampling: Foundations of ReSTIR. *ACM Transactions on Graphics (TOG)* 41, 75.

Daqi Lin, Chris Wyman, and Cem Yuksel. 2021. Fast volume rendering with spatiotemporal reservoir resampling. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–18.

Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. 2020. Langevin Monte Carlo Rendering with Gradient-based Adaptation. *ACM Trans. Graph.* 39, 4 (2020).

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. 1953. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics* 21, 6 (1953), 1087–1092.

Don P. Mitchell. 1987. Generating Antialiased Images at Low Sampling Densities. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*. Association for Computing Machinery, New York, NY, USA, 65–72. https://doi.org/10.1145/37401.37410

Thomas Müller, Markus Gross, and Jan Novák. 2017. Practical path guiding for efficient light-transport simulation. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 91–100.

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

Radford M Neal. 2001. Annealed importance sampling. *Statistics and computing* 11, 2 (2001), 125–139.

NVIDIA. 2017. NVIDIA OptiX AI-Accelerated Denoiser. https://developer.nvidia.com/optix-denoiser.

NVIDIA. 2022. NVIDIA Real-time denoisers (NRD). https://developer.nvidia.com/rtx/ray-tracing/rt-denoisers.

Hisanari Otsu, Johannes Hanika, Toshiya Hachisuka, and Carsten Dachsbacher. 2018. Geometry-aware Metropolis light transport. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–11.

Yaobin Ouyang, Shiqiu Liu, Markus Kettunen, Matt Pharr, and Jacopo Pantaleoni. 2021. ReSTIR GI: Path Resampling for Real-Time Path Tracing. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 17–29.

Mark Pauly, Thomas Kollig, and Alexander Keller. 2000. Metropolis light transport for participating media. In *Rendering Techniques 2000: Proceedings of the Eurographics Workshop in Brno, Czech Republic, June 26–28, 2000 11*. Springer, 11–22.

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.

Christoph Schied, Christoph Peters, and Carsten Dachsbacher. 2018. Gradient estimation for real-time adaptive temporal filtering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1, 2 (2018), 1–16.

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. https://doi.org/10.2312/EGWR/EGSR05/139-146

Justin F Talbot. 2005. *Importance resampling for global illumination*. Brigham Young University.

Joran Van de Woestijne, Roald Frederickx, Niels Billen, and Philip Dutré. 2017. Temporal coherence for Metropolis light transport. In *Eurographics Symposium on Rendering-Experimental Ideas & Implementations*. Eurographics Association, 55–63.

Eric Veach. 1998. *Robust Monte Carlo methods for light transport simulation*. Stanford University.

Eric Veach and Leonidas Guibas. 1995a. Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*. Springer, 145–167.

Eric Veach and Leonidas J Guibas. 1995b. Optimally combining sampling techniques for Monte Carlo rendering. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. 419–428.

Eric Veach and Leonidas J Guibas. 1997. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 65–76.

Gregory J Ward, Francis M Rubinstein, and Robert D Clear. 1988. A ray tracing solution for diffuse interreflection. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*. 85–92.

Chris Wyman. 2021. Weighted Reservoir Sampling: Randomly Sampling Streams. In *Ray Tracing Gems II*. Springer, 345–349.

Chris Wyman and Alexey Panteleev. 2021. Rearchitecting Spatiotemporal Resampling for Production. *ACM/Eurographics Symposium on High Performance Graphics* (2021).

A UNBIASED CONTRIBUTION WEIGHTS AND ELIMINATION OF STARTUP BIAS

Equation 16 shows how to update the contribution weight $\widehat{W}(x^k)$ of a mutated sample x^k from the Markov chain $x^0, ..., x^k, ...$, generated with target function $\hat{p}$. Here we prove this rule yields an unbiased contribution weight for any mutated sample x^k, *i.e.*, for any f with the same or smaller support,

$$\mathbb{E}[f(x^k)\widehat{W}(x^k)] = \int_\Omega f(x)\, \mathrm{d}x. \tag{22}$$

We assume sample x^0 initializing the chain has the same support Ω as target function $\hat{p}$. For us, this is guaranteed by chained applications of RIS with a valid shift map in Algorithm 2. Any x^0 chosen by RIS is not distributed exactly proportional to $\hat{p}$ (unless we have infinite samples), however its contribution weight $\widehat{W}(x^0)$ is unbiased and satisfies Equation 22 [Lin et al. 2022]. Next, we show access to $\widehat{W}(x^0)$ is sufficient to eliminate any startup bias with MH.

Proof. To show Equation 22 holds, we first express the update rule for contribution weight $\widehat{W}(x^k)$ (for $k > 0$) in terms of the previous sample x^{k-1} in the chain, as follows:

$$\widehat{W}(x^k) = \frac{\hat{p}(x^{k-1})}{\hat{p}(x^k)}\widehat{W}(x^{k-1}). \tag{23}$$

This is equivalent to Equation 16, shown by recursively unfolding this relationship for all prior samples x^{k-1} to x^1 in the chain. As in Section 2.5, we also assume a candidate mutation to x^{k-1} is generated using the proposal density $T(x^{k-1} \rightarrow z^{k-1})$, with acceptance probability for candidate z^{k-1} given by Equation 14:

$$a(x^{k-1} \rightarrow z^{k-1}) := \min\left(1, \frac{\hat{p}(z^{k-1})\, T(z^{k-1} \rightarrow x^{k-1})}{\hat{p}(x^{k-1})\, T(x^{k-1} \rightarrow z^{k-1})}\right). \tag{24}$$

Metropolis–Hastings sets $x^k = z^{k-1}$ with probability a; otherwise $x^k = x^{k-1}$. This lets us rewrite the expectation in Equation 22:

$$
\begin{aligned}
\mathbb{E}[f(x^k)\widehat{W}(x^k)] =& \mathbb{E}\left[f(x^k)\frac{\hat{p}(x^{k-1})}{\hat{p}(x^k)}\widehat{W}(x^{k-1})\right] \\
=& \mathbb{E}\left[f(z^{k-1})a(x^{k-1} \rightarrow z^{k-1})\frac{\hat{p}(x^{k-1})}{\hat{p}(z^{k-1})}\widehat{W}(x^{k-1})\right] \\
& + \mathbb{E}[f(x^{k-1})(1 - a(x^{k-1} \rightarrow z^{k-1}))\widehat{W}(x^{k-1})].
\end{aligned}
\tag{25}
$$

Rearranging the terms slightly yields

$$
\begin{aligned}
\mathbb{E}[f(x^k)\widehat{W}(x^k)] = &\, \mathbb{E}[f(x^{k-1})\widehat{W}(x^{k-1})] + \\
& \mathbb{E}\left[f(z^{k-1})a(x^{k-1} \rightarrow z^{k-1})\frac{\hat{p}(x^{k-1})}{\hat{p}(z^{k-1})}\widehat{W}(x^{k-1})\right] - \\
& \mathbb{E}[f(x^{k-1})a(x^{k-1} \rightarrow z^{k-1})\widehat{W}(x^{k-1})].
\end{aligned}
\tag{26}
$$

We now write each expectation as an integral. First, assume an inductive hypothesis $\mathbb{E}[f(x^{k-1})\widehat{W}(x^{k-1})] = \int_{\Omega} f(x)\,\mathrm{d}x$ for the $k-1^{\text{st}}$ MH iteration. Base case $k=1$ holds trivially, as $\widehat{W}(x^0)$ is an unbiased contribution weight. Next, note that for any integrable function $g(x^{k-1}, z^{k-1})$, its expectation $\mathbb{E}[g(x^{k-1}, z^{k-1})\widehat{W}(x^{k-1})]$ can be rewritten as a conditional expectation over candidate mutations:

$$\mathbb{E}[g(x^{k-1}, z^{k-1})\widehat{W}(x^{k-1})] = \mathbb{E}[E[g(x^{k-1}, z^{k-1}) \mid x^{k-1}]\,\widehat{W}(x^{k-1})]$$
$$= \mathbb{E}\left[\left(\int_{\Omega} g(x^{k-1}, z)T(x^{k-1} \to z)\,\mathrm{d}z\right)\widehat{W}(x^{k-1})\right], \quad (27)$$

where T is the proposal density used for mutations. This lets us expand out Equation 26 as follows:

$$\mathbb{E}[f(x^k)\widehat{W}(x^k)] = \int_{\Omega} f(x)\,\mathrm{d}x +$$
$$\int_{\Omega}\int_{\Omega} f(z)a(x \to z)\frac{\hat{p}(x)}{\hat{p}(z)}T(x \to z)\,\mathrm{d}z\,\mathrm{d}x -$$
$$\int_{\Omega}\int_{\Omega} f(x)a(x \to z)T(x \to z)\,\mathrm{d}z\,\mathrm{d}x. \quad (28)$$

Finally, we show that the two double integrals cancel each other out (resulting in $\mathbb{E}[f(x^k)\widehat{W}(x^k)] = \int_{\Omega} f(x)\,\mathrm{d}x$) by invoking the detailed balance condition from Equation 15:

$$\hat{p}(x)T(x \to z)a(x \to z) = \hat{p}(z)T(z \to x)a(z \to x) \quad (29)$$

and rewriting it as:

$$T(x \to z)a(x \to z) = \frac{\hat{p}(z)}{\hat{p}(x)}T(z \to x)a(z \to x). \quad (30)$$

Substituting for $T(x \to z)a(x \to z)$ in the third line of Equation 28 yields the same integral as the second line, but with integration variables x and z swapped. Renaming x and z and swapping the integration order in the third line allows cancellation, simplifying to $\int_{\Omega} f(x)\,\mathrm{d}x$, yielding Equation 22 and giving a proof by induction.

B MIS WEIGHTS FOR TEMPORAL REUSE

ReSTIR uses MIS weights during resampling (Equations 12 and 13) to mitigate noise and bias from reusing samples across pixels. We provide explicit expressions for the MIS weights used in Algorithm 2 here; Lin et al. [2022, Equations 37–38] provide similar expressions for Pairwise MIS weights needed for spatial resampling.

Let $S_j : \Omega_j \to \Omega_i$ denote the shift map from pixel j to pixel i. Let x_i and x_j further represent the corresponding samples for these pixels, and $S_j(x_j) = x_j'$ and $S_j^{-1}(x_i) = x_i'$ the respective shift mapped values. The MIS weights for x_i and x_j' are then given by:

$$m_i(x_i) = \frac{M_i\,\hat{p}_i(x_i)}{M_i\,\hat{p}_i(x_i) + M_j\,\hat{p}_j(x_i')\,|\partial x_i'/\partial x_i|}, \quad (31)$$

$$m_j(x_j') = \frac{M_j\,\hat{p}_j(x_j)\,|\partial x_j/\partial x_j'|}{M_j\,\hat{p}_j(x_j)\,|\partial x_j/\partial x_j'| + M_i\,\hat{p}_i(x_j')}. \quad (32)$$

We set $m_i(x_i) = 1$ and $m_j(x_j') = 0$ when valid shifts do not exist for x_i and x_j (resp.); Lin et al. [2022, Section 5.6] discusses properties of these MIS weights in detail.

C TRANSITION KERNEL FOR MUTATING A RECONNECTION VERTEX

A mutation involving a reconnection vertex $\mathbf{y}_i$ requires modifying random numbers not just for $\mathbf{y}_i$, but also for the non-mutated vertices $\mathbf{y}_{i+1}$ and $\mathbf{y}_{i+2}$. This is because the solid angle PDFs used to sample outgoing directions v_i and ω_{i+1} in Section 4.3 depend on the mutated incoming directions v_{i-1} and v_i (resp.). Here we derive Equation 20 by first noting the joint PDF for connecting the mutated reconnection vertex $\mathbf{z}_i$ to $\mathbf{z}_{i+1}$ and $\mathbf{z}_{i+1}$ to $\mathbf{z}_{i+2}$ in the surface area measure is:

$$p(\mathbf{z}_{i+2}, \mathbf{z}_{i+1} | [\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_i]) = \delta(\mathbf{z}_{i+2} - \mathbf{y}_{i+2})\delta(\mathbf{z}_{i+1} - \mathbf{y}_{i+1}).$$

This is a product of delta functions as $\mathbf{z}_{i+1} = \mathbf{y}_{i+1}$ and $\mathbf{z}_{i+2} = \mathbf{y}_{i+2}$ are the only valid vertex positions. In the PSS to path space mapping, the joint PDF for the mutated random numbers $\bar{\mathbf{v}}_{i+1}$ and $\bar{\mathbf{v}}_{i+2}$ (for vertices $\mathbf{y}_{i+1}$ and $\mathbf{y}_{i+2}$) is related to $p(\mathbf{z}_{i+2}, \mathbf{z}_{i+1} \mid [\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_i])$ via a Jacobian determinant:

$$p(\bar{\mathbf{v}}_{i+2}, \bar{\mathbf{v}}_{i+1} | [\bar{\mathbf{v}}_0, \bar{\mathbf{v}}_1, \ldots, \bar{\mathbf{v}}_i]) = p(\mathbf{z}_{i+2}, \mathbf{z}_{i+1} | [\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_i])\left|\frac{\partial \bar{\mathbf{z}}}{\partial \bar{\mathbf{v}}}\right|$$

This PDF serves as our proposal density $T(\bar{\mathbf{u}} \to \bar{\mathbf{v}})$ for mutations, which then yields:

$$\frac{T(\bar{\mathbf{v}} \to \bar{\mathbf{u}})}{T(\bar{\mathbf{u}} \to \bar{\mathbf{v}})} = \left|\frac{\partial \bar{\mathbf{v}}}{\partial \bar{\mathbf{z}}}\right|\left|\frac{\partial \bar{\mathbf{y}}}{\partial \bar{\mathbf{u}}}\right|\frac{\delta(\mathbf{y}_{i+2} - \mathbf{z}_{i+2})\delta(\mathbf{y}_{i+1} - \mathbf{z}_{i+1})}{\delta(\mathbf{z}_{i+2} - \mathbf{y}_{i+2})\delta(\mathbf{z}_{i+1} - \mathbf{y}_{i+1})}$$
$$= \left|\frac{\partial \bar{\mathbf{v}}}{\partial \bar{v}}\right|\left|\frac{\partial \bar{v}}{\partial \bar{\mathbf{z}}}\right|\left|\frac{\partial \bar{\mathbf{y}}}{\partial \bar{\omega}}\right|\left|\frac{\partial \bar{\omega}}{\partial \bar{\mathbf{u}}}\right|$$
$$= \left|\frac{\partial \bar{v}}{\partial \bar{\mathbf{z}}}\right|\left|\frac{\partial \bar{\mathbf{y}}}{\partial \bar{\omega}}\right|\frac{p(v_{i-1}, v_i)}{p(\omega_{i-1}, \omega_i)}\frac{p(v_i, \omega_{i+1})}{p(\omega_i, \omega_{i+1})}$$
$$= \frac{|\cos \phi|\;|\mathbf{y}_{i+1} - \mathbf{y}_i|^2}{|\cos \theta|\;|\mathbf{y}_{i+1} - \mathbf{z}_i|^2}\frac{p(v_{i-1}, v_i)}{p(\omega_{i-1}, \omega_i)}\frac{p(v_i, \omega_{i+1})}{p(\omega_i, \omega_{i+1})}. \quad (33)$$

The delta functions in the first line cancel since they are symmetric. In the third line, we use the fact that the Jacobian determinant of a sampling scheme is the same as its inverse PDF [Kelemen et al. 2002, Section 2]. The final step substitutes in the definition of the Jacobians relating the solid angle and area measures.