ReSTIR BDPT: Bidirectional ReSTIR Path Tracing with Caustics

TREVOR HEDSTROM, University of California San Diego, USA
MARKUS KETTUNEN, NVIDIA, Finland
DAQI LIN, NVIDIA, USA
CHRIS WYMAN, NVIDIA, USA
TZU-MAO LI, University of California San Diego, USA

Fig. 1. We combine bidirectional path tracing with a novel reuse scheme to enable ReSTIR [Wyman et al. 2023] to better find hard-to-reach light sources and even resolve caustics interactively. Only emissive filaments inside the glass blubs light this Bathroom scene. At a 1920×1080 resolution, our method achieves a mean absolute percentage error (MAPE) of 0.312 in 70ms (with 1M light subpaths), while ReSTIR PT [Lin et al. 2022] achieves a MAPE of 1.368 in 71ms.

Recent spatiotemporal resampling algorithms (ReSTIR) accelerate real-time path tracing by reusing samples between pixels and frames. However, existing methods are limited by the sampling quality of path tracing, making them inefficient for scenes with caustics and hard-to-reach lights. We develop a ReSTIR variant incorporating bidirectional path tracing that significantly improves the sampling quality in these scenes.

Combining bidirectional path tracing and ReSTIR introduces multiple challenges: the generalized resampled importance sampling (GRIS) behind ReSTIR is, by default, not aware of how a path was sampled, which complicates reuse of bidirectional paths. Light tracing is also challenging since light subpaths can contribute to all pixels. To address these challenges, we apply GRIS in a sampling technique-aware extended path space, design a bidirectional hybrid shift mapping, and introduce caustics reservoirs that can accumulate caustics across frames. Our method takes around 50ms per frame across our test scenes, and achieves significantly lower error compared to prior unidirectional ReSTIR variants running in equal time.

Authors' addresses: Trevor Hedstrom, University of California San Diego, USA, tjhedstr@ucsd.edu; Markus Kettunen, NVIDIA, Finland, mkettunen@nvidia.com; Daqi Lin, NVIDIA, USA, daqil@nvidia.com; Chris Wyman, NVIDIA, USA, chris.wyman@acm.org; Tzu-Mao Li, University of California San Diego, USA, tzli@ucsd.edu.

0730-0301/2025/0-ART0 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

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

ACM Reference Format:
Trevor Hedstrom, Markus Kettunen, Daqi Lin, Chris Wyman, and Tzu-Mao Li. 2025. ReSTIR BDPT: Bidirectional ReSTIR Path Tracing with Caustics. *ACM Trans. Graph.* 0, 0, Article 0 (2025), 17 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

ReSTIR PT [Lin et al. 2022] improves real-time rendering quality by orders of magnitude by reusing path samples between pixels and frames. However, ReSTIR PT inherits weaknesses from unidirectional path tracing for scenes with caustics or hard-to-reach lights (Figure 1). Our work aims to significantly improve ReSTIR PT in these scenes by incorporating bidirectional path tracing (BDPT).

Naïvely applying generalized resampled importance sampling (GRIS), the main engine behind ReSTIR PT, to real-time bidirectional path tracing is not feasible. When reusing across pixels or frames, GRIS does not directly account for which technique generates a path sample; and in BDPT, many sampling techniques can form each path. This makes parameterizing a path with one random seed infeasible, calling for expensive path storage. Additionally, light tracing poses additional challenges for screen-space reuse as it produces samples contributing to all pixels.

To address these challenges, we apply GRIS theory in an extended path space consisting of path–technique pairs $(\bar{x}, \tau)$ joined by technique-specific multiple importance sampling (MIS) weights. These *extended paths* allow principled reasoning about the sampling technique at path reuse without introducing bias. Our method enables a seemingly infeasible choice, using Lin et al.'s [2022] hybrid shift for camera sub-paths and random replay for light sub-paths.

We achieve interactive caustic rendering by implementing a bidirectional hybrid shift, as well as storing an additional *caustic reservoir* in each pixel. The caustic reservoir enables unbiased temporal accumulation of caustics over multiple frames in dynamic scenes. Scene changes may move caustics between pixels, but they often remain nearly converged. Accumulation is supported by merging new caustic paths into caustic reservoirs each frame. Our caustic reservoirs automatically aggregate and stratify the caustics into the pixels, allowing interactive updates and retaining high caustic quality as the scene evolves.

We further accelerate our ReSTIR BDPT by deriving a novel variant of recursive MIS weights [van Antwerpen 2011] for fast MIS weight evaluation with shift mappings. We also propose two faster alternatives, a biased variant which copies the unshifted MIS weight for cases where full unbiasedness is not required, and an unbiased *lightweight* variant which omits vertex connection techniques.

Our method runs interactively, taking around 50ms per frame in most of our test scenes (Figure 11). On difficult scenes at one sample per pixel, we achieve much lower error compared to ReSTIR PT in equal time due to the improved efficiency of bidirectional sampling.

2 BACKGROUND

We briefly review resampled importance sampling [Talbot et al. 2005], its generalization [Lin et al. 2022], and bidirectional path tracing [Veach and Guibas 1995a]. We refer readers to Wyman et al.'s [2023] course notes for a more comprehensive ReSTIR review [Bitterli et al. 2020; Lin et al. 2022], and physically-based rendering textbooks [Pharr et al. 2016] and Veach's thesis [1997] for an introduction to path space and bidirectional path tracing.

Our notation and key symbols are summarized in Table 1.

Table 1. Summary of notation.

$\bar{x}$	Full path
$\bar{y}$	Light subpath (with s vertices)
$\bar{z}$	Camera subpath (with t vertices)
X	Random variable for a path
Y	Random variable for a light subpath
Z	Random variable for a camera subpath
x_r	Reconnection vertex
x_1	Primary hit
x_2	Secondary hit
p_L	Light subpath selection probability
p	Canonical probability density function
$\overrightarrow{p}^{\sigma}_i$	"Forward" solid-angle PDF of sampling x_i from x_{i-1}
$\overleftarrow{p}^{\sigma}_i$	"Reverse" solid-angle PDF of sampling x_i from x_{i+1}
$\overrightarrow{p}_i$	$\overrightarrow{p}^{\sigma}_i$ expressed in area measure
$\overleftarrow{p}_i$	$\overleftarrow{p}^{\sigma}_i$ expressed in area measure
$\hat{p}, \hat{p}_\tau$	RIS target functions
$\omega_\tau, \omega_{s,t}$	BDPT MIS weight for technique τ or (s,t)
M	Resampling candidate count
m_i	Resampling MIS weight for candidate i
c_i	Confidence weight for reservoir i

2.1 Path integrals and notations

In rendering, a pixel's intensity I is computed as an integral over all light paths connecting it to light sources [Kajiya 1986; Veach 1997],

$$I = \int_\Omega f(\bar{x}) \, \mathrm{d}\mu(\bar{x}), \tag{1}$$

for a light path $\bar{x} = (x_0, x_1, \cdots, x_k)$ with $k+1$ vertices, where x_0 is on the camera sensor and x_k is on a light. $\Omega = \bigcup_{k=1}^{\infty} \mathcal{M}^{k+1}$ is the *path space* with the union of scene surfaces $\mathcal{M}$, and μ is the area product measure. The *measurement contribution function* f turns radiance transported along a path into pixel response:

$$f(\bar{x}) = W_\mathrm{e}(x_1 \to x_0) \cdot G(x_0 \leftrightarrow x_1) \cdot L_\mathrm{e}(x_k \to x_{k-1}) \cdot \left(\prod_{i=1}^{k-1} \rho(x_{i+1} \to x_i \to x_{i-1}) \cdot G(x_i \leftrightarrow x_{i+1}) \right), \tag{2}$$

where W_e is sensor importance of the pixel, L_e is light emission, G is the geometry term, and ρ is the bidirectional scattering distribution function (BSDF).

In standard unidirectional path tracing (PT) [Kajiya 1986], the pixel intensity (Equation 1) is given by the Monte Carlo estimator

$$\langle I \rangle_\mathrm{PT} = \frac{f(X)}{p(X)}, \tag{3}$$

where p is the probability density function (PDF) for random path X. The estimation noise is smaller when p better matches f.

2.2 Bidirectional path tracing

Bidirectional path tracing (BDPT) [Lafortune and Willems 1993; Veach and Guibas 1995a] improves sampling efficiency over unidirectional path tracing in complex scenes. It samples two subpaths: one traced from the camera and one from the light. We denote camera

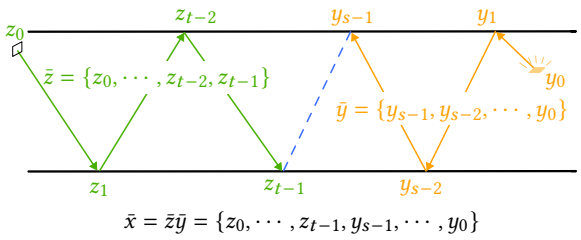

Fig. 2. A path formed by bidirectional path tracing. $\bar{z}$ is the camera subpath, $\bar{y}$ is the light subpath, and the full path is formed by appending them together: $\bar{x} = \bar{z}\bar{y}$

subpaths with t vertices as $\bar{z}_t = (z_0, z_1, \cdots, z_{t-1})$ for z_0 on the camera, and light subpaths with s vertices as $\bar{y}_s = (y_{s-1}, y_{s-2}, \cdots, y_0)$ for y_0 on the light (see Figure 2).

By connecting camera and light subpaths at different vertices, we form full paths $\bar{x}_{s,t} = \bar{z}_t\bar{y}_s$ with $t + s$ vertices. Case $t = 1$ gives *light tracing*, with an explicit connection to the camera (z_0) from the light subpath's end (y_{s-1}). Case $t = 0$ corresponds to y_{s-1} randomly selecting the sensor surface via BSDF sampling. In our implementation, we omit $t = 0$ paths, as they have zero probability for pinhole cameras and otherwise contribute very little to the final image [Veach and Guibas 1995a].

Bidirectional sampling can produce the same path by many sampling techniques (s, t), by connecting subpaths of different lengths. This can over-count the same path. To form an unbiased estimator, Veach and Guibas [1995b] combine these sampling techniques using multiple importance sampling (MIS):

$$\langle I \rangle_{\text{BDPT}} = \sum_{t>0, s\geq 0} \omega_{s,t}(X_{s,t}) \frac{f(X_{s,t})}{p_{s,t}(X_{s,t})}, \tag{4}$$

where $p_{s,t}(X)$ is the probability density of sampling path X with technique (s, t), and MIS weights $\omega_{s,t}$ form a partition of unity over the probability densities of all ways to form the same path:

$$p_{s,t}(X) = \prod_{i=1}^{t-1} \overrightarrow{p_i} \prod_{i=t}^{s+t-1} \overleftarrow{p_i} \tag{5}$$

$$\omega_{s,t}(X) = \frac{\left(p_{s,t}(X)\right)^\beta}{\sum_{t'+s'=t+s} \left(p_{s',t'}(X)\right)^\beta}. \tag{6}$$

Here, β is the power used by MIS (e.g., $\beta = 1$ for the balance heuristic, $\beta > 1$ for the power heuristic). *Forward* PDFs $\overrightarrow{p_i}$ are area-measure probability densities for sampling vertex x_i from vertex x_{i-1} during camera tracing. Similarly, *reverse* PDFs $\overleftarrow{p_i}$ represent densities for sampling vertex x_i from vertex x_{i+1} during light tracing. This follows notation in van Antwerpen [2011] and Georgiev et al. [2012]. Notably absent in Equation 5 are the densities $\overrightarrow{p_t}$ and $\overleftarrow{p_{t-1}}$ along the subpath connection; this connection is made deterministically.

By combining techniques with MIS, bidirectional path tracing excels at rendering intricate light paths, e.g., difficult-to-reach lights or complex caustics; different techniques are efficient in different situations, but due to MIS, BDPT is typically efficient as long as at least one of the techniques is efficient[1].

Recursive MIS calculations. Classical BDPT stores forward and reverse densities per vertex [Veach 1997] and loops over all techniques for a path to compute $\omega_{s,t}$. This uses storage proportional to path length and reduces compute efficiency on the GPU. By observing that subpath probabilities of sequential vertices share much of the product prefix ($\prod \overrightarrow{p_s}$ in Equation 5), van Antwerpen [2011] formulated an efficient recursive relation to incrementally compute and store data at path vertices so computing $\omega_{s,t}$ only needs data from the two connection vertices. This allows discarding other path vertices after tracing, enabling GPU-efficient algorithms without per-vertex path storage [Davidovič et al. 2014; van Antwerpen 2011].

Light Vertex Cache (LVC). Classical BDPT generates samples for a pixel by creating a pair of camera and light subpaths and evaluating all possible connections along each subpath. Many later variants [Davidovič et al. 2014; Nabata et al. 2020; Pajot et al. 2011; Popov et al. 2015] explore different ways to combine subpaths for more efficient sampling. The Light Vertex Cache (LVC) algorithm [Davidovič et al. 2014] first generates N_L light subpaths shared over all pixels, and lets per-pixel camera subpaths connect to random light subpaths. Instead of storing light subpaths, it stores connectable light subpath vertices in an array called the *light vertex cache*. During camera tracing, camera subpaths connect to random vertices in the LVC. After connecting, efficient computation of BDPT MIS weights is enabled by van Antwerpen's [2011] recursive MIS formulation. This transforms BDPT into a GPU-friendly streaming algorithm where camera subpaths connect to a stream of random light subpath vertices, instead of a single light subpath.

2.3 Resampled Importance Sampling (RIS)

Estimating the integral in Equation 1 is often done using Monte Carlo integration: $\langle I \rangle_{\text{MC}} = \frac{f(X)}{p(X)}$, where $p(X)$ is the probability density of sampling path X. Without loss of generality, we assume one sample per pixel. Ideally, we want a probability density proportional to the integrand f, but directly sampling with such a distribution is often intractable. Resampled importance sampling (RIS) [Talbot et al. 2005] approximates this by first sampling M i.i.d. candidate samples $X_1, X_2, \ldots, X_M$ from an easier-to-sample distribution p, and chooses one of them as the output Y_z proportionally to *resampling weights* w_i. Following Lin et al. [2022], we have

$$w_i = \frac{1}{M} \frac{\hat{p}(X_i)}{p(X_i)}, \tag{7}$$

where the *target function* $\hat{p}$ is proportional to the often intractable target density.

The resampling process makes the output Y_z's probability density p_{Y_z} computationally intractable, rendering the classical Monte Carlo estimator f/p inadequate. Instead, we can compute an *unbiased*

[1]Full BDPT can be less efficient than a subset of techniques due to the extra cost from evaluating all techniques, or from sub-optimal MIS weights [Grittmann et al. 2019].

contribution weight (UCW) [Lin et al. 2022] W_{Y_z} for Y_z,

$$W_{Y_z} = \frac{1}{\hat{p}(Y_z)} \sum_{i=1}^{M} w_i, \tag{8}$$

which is an unbiased estimate for the unknown reciprocal probability density $1/p_{Y_z}(Y_z)$. The classical Monte Carlo estimator $\langle I \rangle_{\text{MC}}$ is replaced with

$$\langle I \rangle_{\text{RIS}} = f(Y_z) W_{Y_Z}. \tag{9}$$

As candidate count M increases, the (intractable) output probability density approaches the *target PDF* $\bar{p} = \hat{p} / \int \hat{p}$, and W_{Y_z} approaches $1/\bar{p}(Y_z)$. Ideally, $\hat{p}$ would be f itself for asymptotically perfect importance sampling.

RIS allows candidates X_i with different source distributions p_i using multiple importance sampling [Lin et al. 2022; Talbot et al. 2005]. We simply replace the $1/M$ in Equation 7 with *resampling MIS weights* m_i:

$$w_i = m_i(X_i) \frac{\hat{p}(X_i)}{p_i(X_i)}, \tag{10}$$

where, e.g.,

$$m_i(\bar{x}) = \frac{p_i(\bar{x})}{\sum_{j=1}^{M} p_j(\bar{x})} \tag{11}$$

for the balance heuristic. If all X_i are identically distributed (meaning all p_i are identical), then the weights m_i reduce to the same $1/M$. Equations 8 and 9 remain unchanged.

2.4 Spatiotemporal reuse and ReSTIR

The idea of ReSTIR [Bitterli et al. 2020] is to recursively apply RIS to reuse samples from each pixel's spatiotemporal neighbors. Lin et al. [2022] formalized this in their generalized RIS (GRIS), defining the unbiased contribution weights and allowing them in resampling weights in place of the unknown reciprocal PDFs:

$$w_i = m_i(X_i) \hat{p}(X_i) W_{X_i}. \tag{12}$$

Since the PDFs p_i are not known, the resampling MIS weights m_i must be modified. A simple choice is the *generalized balance heuristic*, using input X_i's target function $\hat{p}_i$ as a proxy for p_i:

$$m_i(X_i) = \frac{\hat{p}_i(X_i)}{\sum_{j=1}^{M} \hat{p}_j(X_i)}. \tag{13}$$

To deal with the fact that the candidates from a pixel's spatiotemporal neighbors are from different domains, Lin et al. [2022] transfer paths between domains with *shift mappings* T_i from the gradient-domain rendering literature [Kettunen et al. 2015; Lehtinen et al. 2013]. A shift mapping T_i takes a path $\bar{x}$ from domain Ω_i and slightly modifies it into a similar path $T_i(\bar{x})$ in Ω. Not every path needs to be shiftable; a shift mapping bijectively maps paths from a subset of Ω_i into a subset of Ω. A shift may be undefined, e.g., due to occlusion or a failure to ensure bijectivity (e.g., due to total internal reflection). See Wyman et al. [2023] for details.

Reusing paths with shift mappings further modifies the resampling weights, but other formulas remain the same. The inputs X_i are mapped from the source domains into the current pixel's domain as $T_i(X_i)$, and a Jacobian determinant is added to transform the contribution weights to the current domain, giving

$$w_i = m_i(T_i(X_i))\, \hat{p}(T_i(X_i))\, W_{X_i} \left| \frac{\partial T_i}{\partial X_i} \right|. \tag{14}$$

The MIS weight m_i and target function $\hat{p}$ must be evaluated in the current domain, i.e., at $T_i(X_i)$ instead of X_i. Multiplying the UCW W_{X_i} by the Jacobian determinant technically transforms it into $W_{T_i(X_i)}$ for the shifted path. The output Y_z is then resampled from the $\{T_i(X_i)\}_i$ proportionally to the w_i, and Equations 8 and 9 remain unchanged.

The resampling MIS weights m_i require another change. Mapping a random variable X_i with a shift map T_i modifies its probability density. Lin et al. [2022] account for this in the resampling MIS weights by defining a function $\hat{p}_{\leftarrow i}$, "$\hat{p}$ from i" that shifts path $\bar{x}$ back to domain i for target function evaluation and accounts for density changes using the Jacobian determinant,

$$\hat{p}_{\leftarrow i}(\bar{x}) = \begin{cases} \hat{p}_i\left(T_i^{-1}(\bar{x})\right) \left| \frac{\partial T_i^{-1}}{\partial \bar{x}} \right| & \bar{x} \in T_i\,(\operatorname{supp} X_i) \\ 0 & \text{otherwise} \end{cases}. \tag{15}$$

Condition $\bar{x} \in T_i(\operatorname{supp} X_i)$ amounts to T_i^{-1} being defined[2]. This gives the generalized balance heuristic

$$m_i(\bar{x}) = \frac{\hat{p}_{\leftarrow i}(\bar{x})}{\sum_{j=1}^{M} \hat{p}_{\leftarrow j}(\bar{x})}. \tag{16}$$

In practice, the terms are scaled by confidence weights c_i, which effectively weight each domain based on the number of samples each domain has processed, yielding

$$m_i(\bar{x}) = \frac{c_i\, \hat{p}_{\leftarrow i}(\bar{x})}{\sum_{j=1}^{M} c_j\, \hat{p}_{\leftarrow j}(\bar{x})}. \tag{17}$$

Kettunen et al. [2023, supplemental] connect Equation 17 to Veach and Guibas [1995b]'s multi-sample MIS, if M_i is interpreted as X_i's *effective sample count*, i.e., how many simple samples it represents. For instance, if ReSTIR has improved a prior frame's sample to correspond to c_i current-frame samples, sensible temporal reuse would weight the prior sample by c_i and the current sample by 1. Effective sample counts are hard to compute, so actual sample counts with a fixed confidence cap are often used instead.

A more computationally efficient but less robust alternative to the generalized balance heuristic is the pairwise MIS weight [Bitterli et al. 2020]. We refer readers to the recent ReSTIR course for complete formulas [Wyman et al. 2023].

ReSTIR PT's [Lin et al. 2022] shift mapping, the **hybrid shift**, reuses paths by first tracing a new primary ray from the target pixel, and then combining two operations:

- **Random replay**: continue tracing the shifted path by copying the random numbers used in the original path,
- **Reconnection**: connect back to the next path vertex of the original path.

The hybrid shift applies reconnection if the current vertex on the shifted path and the next vertex on the original path are both *connectable*, i.e., follow a simple vertex roughness and distance based

[2]The condition also requires that the PDF of X_i is positive at $T_i^{-1}(\bar{x})$, but this is normally guaranteed by $\hat{p}_i(T_i^{-1}(\bar{x})) > 0$.

heuristic that guarantees the reconnection is valid (not e.g., on a mirror); random replay postpones reconnection until a connectable vertex is found. Once joined by reconnection, spatial reuse reuses the rest of the path, while temporal reuse requires re-tracing in the new scene: the reconnection vertex is moved with the geometry to follow possible animation, and the later vertices are re-traced with random replay to avoid excess vertex storage on the GPU.

To summarize, ReSTIR is the combination of initial sampling, temporal reuse with GRIS, and spatial reuse between pixels with GRIS. GRIS aggregates multiple samples into one by appropriately weighting each input (e.g., Equation 14) and choosing which one to output proportionally to these resampling weights. The output aggregates many samples over pixels and frames, and is assigned an unbiased contribution weight by Equation 8. The UCW may be used, for example, to recursively resample or integrate with Equation 9. By combining samples spatiotemporally, ReSTIR improves the sample distribution, greatly reducing noise even with just a single sample per pixel per frame.

2.5 Resampling for BDPT

RIS has previously been used to improve subpath connections within BDPT. Two-stage Resampling Nabata et al. [2020] and SPCBPT Su et al. [2022] apply RIS to sample important light subpaths for subpath connections (techniques with both $s > 1$ and $t > 1$). More recently, Liu and Gan [2023] use grid-based reservoirs to resample light subpaths over time. These methods only use RIS to select light subpaths for bidirectional connections. Since they do not modify subpaths (e.g., with reconnection), they are not applicable to light tracing techniques ($t \leq 1$), which are often most efficient for sampling effects such as caustics. They also operate in world space, requiring more storage for large or highly detailed scenes.

The term *caustic reservoir* also appears in the concurrent ReSTIR FG [Kern et al. 2024], which applies resampling to vertex merging. ReSTIR FG separates caustic light paths during resampling to reduce caustic blurring from vertex merging. We also separate caustic light paths, however this is to prevent caustic samples from degrading spatial resampling quality, since they cannot be spatially shifted to other pixels. Our caustic reservoirs are an unbiased estimate of the contribution from caustic light paths, while ReSTIR FG is biased due to the use of vertex merging.

3 OVERVIEW

We aim to produce paths with bidirectional path tracing and apply GRIS to reuse these paths spatiotemporally. Bidirectional path tracing often produces a much better sample distribution, enabling rendering of intricate light paths such as caustics, and improving the sampling quality at low sample counts.

Lin et al. [2022] apply GRIS to unidirectional path tracing, which corresponds to techniques $s = 0$ (camera subpath hits an emissive surface) and $s = 1$ (next event estimation). A main challenge in applying GRIS to bidirectional path tracing is handling the large set of sampling techniques. In particular, we often want to apply random replay as part of Lin et al.'s hybrid reconnection shift. However, we cannot use random replay for bidirectional paths unless we know which technique generated the path: a path combining two subpaths can only be reproduced by replaying the random number states for the camera and light subpaths up until the connection vertices.

Another challenge comes from light tracing techniques, i.e., $t \leq 1$, as such paths can contribute to any pixel. We must correctly weight our samples to account for this fact. Furthermore, we need to design a reuse scheme that works well for light tracing and caustics.

Finally, computing the BDPT MIS weight $\omega_{s,t}$ (Equation 6) during spatiotemporal reuse can be quite involved. We cannot directly apply van Antwerpen's [2011] efficient, recursive MIS formulation, since shift mappings may change parts of the paths.

Below, we address the aforementioned challenges:

- Section 4: We present an *extended path space* that includes information about the sampling technique, and derive the corresponding GRIS estimator for bidirectional path tracing in the extended path space,
- Section 5: We identify shift mappings for bidirectional connections and light tracing, and
- Section 6: We generalize the recursive MIS formulation to efficiently compute the MIS weight $\omega_{s,t}$ during reconnection.

Finally, in Section 7, we discuss our GPU implementation of a ReSTIR-based bidirectional path tracer.

4 BIDIRECTIONAL RESTIR WITH A PATH SPACE EXTENSION

In this section, we derive our bidirectional ReSTIR estimator by applying GRIS in an extended path space. We want to apply GRIS to resample over all bidirectional sampling techniques, using correct MIS weights ω_τ to allow unbiased spatiotemporal reuse.

Previous ReSTIR methods directly apply GRIS to the path integral (Equation 1) by sampling paths and merging them into reservoirs. As mentioned, using random replay shift mappings complicates this case: existing formulations do not allow shift mapping to depend on a path's sampling technique. Thus, when shifting via random replay, we do not know the camera and light subpath lengths.

Therefore, we apply GRIS in an *extended path space*. We pair path samples X with the technique indices τ used to generate them in *extended paths* $\hat{X} = (X, \tau)$. All extended paths of a given technique τ define the technique's path space Ω_τ and our extended path space $\hat{\Omega}$ is the union of all sample-technique pairs, $\hat{\Omega} = \bigcup_\tau \Omega_\tau$. Our extended path space integral is then:

$$\begin{aligned} I &= \int_\Omega f(\bar{x})\,\mathrm{d}\bar{x} = \sum_\tau \int_{\Omega_\tau} \omega_\tau(\bar{x}) f(\bar{x})\,\mathrm{d}\bar{x}, \\ &= \int_{\hat{\Omega}} \omega_\tau(\bar{x}) f(\bar{x})\,\mathrm{d}(\bar{x}, \tau), \end{aligned} \tag{18}$$

where $\omega_\tau(\bar{x})$ is the technique MIS weight of path $\bar{x}$ under technique τ. We weight the measurement contribution f with these MIS weights to avoid counting a path's contribution multiple times.

Now, our goal is to apply GRIS on the extended path space integral (Equation 18). To apply GRIS, we first need two items: the target function $\hat{p}$ and shift mapping T. We set the target function to be always weighted by the technique MIS weight ω_τ:

$$\hat{p}(\hat{x}) = \omega_\tau(\bar{x})\hat{q}(\bar{x}), \tag{19}$$

where $\hat{x} = (\bar{x}, \tau)$ and $\hat{q}$ is a user-defined target function. We always set $\hat{q}$ to be the luminance of the measurement contribution f. We then assume *technique-specific* shift mappings, i.e., the output path depends on the technique, but the technique is not changed in the shift:

$$T(\hat{x}) = T((\bar{x}, \tau)) = (T_\tau(\bar{x}), \tau), \tag{20}$$

where T_τ is the shift mapping defined for technique τ. We will specify T_τ in the next section. The Jacobian is simply

$$\left|\frac{\partial T}{\partial \hat{x}}\right| = \left|\frac{\partial T_\tau}{\partial \bar{x}}\right|. \tag{21}$$

Combining these, we can directly apply GRIS to estimate the extended path space integral (Equation 18). For the resampling output $\hat{Y} = (Y, \tau)$ sampled from the shifted inputs $T_i(\hat{X}_i)$ proportionally to resampling weights w_i, we get

$$\langle I \rangle = \omega_\tau(Y) f(Y) W_{\hat{Y}}. \tag{22}$$

The unbiased contribution weight for the chosen sample at GRIS resampling is

$$W_{\hat{Y}} = \frac{1}{\hat{p}(\hat{Y})} \sum_{i=1}^{M} w_i, \tag{23}$$

where $\hat{p}$ is given by Equation 19, and the candidates' resampling weights are, denoting $\hat{Y}_i = T_i(\hat{X}_i)$,

$$w_i = m_i(\hat{Y}_i)\, \hat{p}(\hat{Y}_i)\, W_{\hat{X}_i} \left|\frac{\partial T_i}{\partial \hat{X}_i}\right|. \tag{24}$$

Here, m_i is the generalized balance heuristic (Equation 17), with a substitution of our $\hat{p}$ (Equation 19) and Jacobian (Equation 21) into the definition of $\hat{p}_{\leftarrow i}$ in Equation 15.

Like Lin et al. [2022], we use the generalized balance heuristic for temporal reuse and pairwise MIS [Lin et al. 2022; Wyman et al. 2023] for spatial reuse. We use pairwise MIS for spatial reuse to avoid the quadratic cost of the balance heuristic. For temporal reuse, we use the balance heuristic since temporal reuse always has just two candidates (the current and prior frame's samples).

In initial sampling, the unbiased contribution weight $W_{\hat{X}_i}$ for a bidirectional sample-technique pair $\hat{X}_i = (X_i, \tau_i)$ from technique $\tau_i = (s, t)$ is the reciprocal of the subpath sampling PDFs:

$$W_{\hat{X}_i} = \begin{cases} \frac{1}{p(Z_i)} \frac{1}{p(Y_i)} \frac{1}{p_L(Y_i)} & s > 1, t > 1 \\ \frac{1}{p(Z_i)} \frac{1}{p(Y_i)} & \text{otherwise} \end{cases}, \tag{25}$$

where Z_i and Y_i are the camera and light subpaths, respectively, and $p_L(Y_i)$ is the probability of selecting Y_i for subpath connection.

Initial sampling generates a path for each technique within a pixel, and we use GRIS to select one. The resampling MIS weights for resampling one path-technique pair $\hat{x} = (\bar{x}, \tau)$ from the initial candidates are, with $\tau = (s, t)$,

$$m_i(\hat{x}) = \begin{cases} 1/N_L & t \le 1 \\ 1 & \text{otherwise} \end{cases}, \tag{26}$$

where N_L is the number of light subpaths. The number of samples a pixel can receive varies by technique: Light tracing techniques $t \le 1$ can contribute to any pixel, so each pixel receives N_L light tracing samples, requiring the divide by N_L to normalize. Other techniques $t \ge 2$ generate samples only for the camera subpath's pixel. With one camera subpath per pixel and each subpath vertex connected to one light subpath, all the other path-technique spaces receive one sample each per pixel.

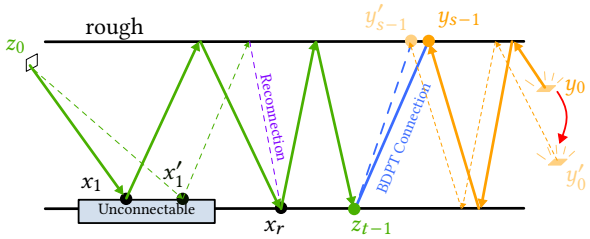

Fig. 3. **Bidirectional shift mapping**. Our shift on a path with $t = 6$ and $s = 4$. We shift the base path (solid lines) to a new primary hit x'_1. When $t > 1$, our bidirectional shift map follows the hybrid shift from Lin et al. [2022], where base path's random numbers are copied before reconnecting to the reconnection vertex x_r. We shift the light subpath with random replay, and reconnect the camera subpath to the shifted light subpath. Note that if $x_r \neq y_{s-1}$ (as in this figure) our shift involves two reconnections: once to reconnect to x_r and again for y_{s-1}.

5 SHIFT MAPPINGS FOR BIDIRECTIONAL SPATIOTEMPORAL REUSE

Now, we define our technique-specific bidirectional hybrid shift mapping T_τ (Equation 20) for $\tau = (s, t)$. We separately discuss three cases: camera tracing techniques ($t \ge 2$), light tracing techniques ($t \le 1$) where the path is non-caustic, and light tracing techniques ($t \le 1$) where the path is caustic. A path is *caustic* if the third vertex from the camera, x_2, is classified as non-rough. A key caustic path property is reconnections from the primary hit are not possible as x_2 is non-rough, leading us to use random replay instead.

Path tracing, next event estimation, and light subpath connections ($t \ge 2$). When techniques have a camera subpath, we shift the light subpath (if $s > 0$) with random replay and the camera subpath (if $t > 1$) with the hybrid shift mapping of ReSTIR PT [Lin et al. 2022]. Case $s = 0$ corresponds to path tracing from the camera and hitting an emitter; $s = 1$ corresponds to next event estimation, and $s \ge 2$ connects to a non-degenerate light subpath.

For techniques with subpath connections ($t > 1$ and $s > 0$), we always reconnect (at latest) to the first light subpath vertex. As Manzi et al. [2015], we only allow subpath connections on vertices classified as rough at sampling time. Thus, for BDPT to connect a camera and light subpath, the connected vertices must be classified rough, so a reconnection shift will also succeed. We do not impose a distance constraint for reconnection between the light and camera subpaths, as technique MIS weights w_τ naturally lower weights for short connections. Additionally, if a shift changes the classification of either vertex (e.g., a rough vertex becomes non-rough) we then force a shift failure to maintain bijectivity.

Figure 3 illustrates a temporal shift for a path with $t = 6$ and $s = 4$. We first trace a new camera subpath from the target pixel's primary hit x'_1. If we sample two consecutive rough vertices, we can reconnect to the second. Otherwise, we reuse the random numbers

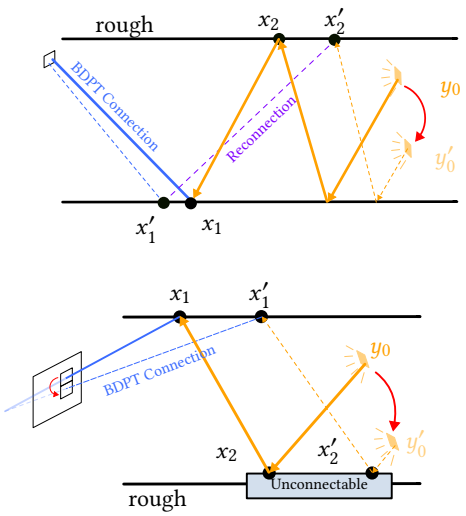

Fig. 4. **Light tracing shift mapping**. Here we illustrate our temporal shift mapping for light tracing techniques ($t \leq 1$). We apply random replay from the light until the the secondary hit x_2'. If x_2' is rough, then we can reconnect to the new primary hit x_1' (top). Otherwise, we continue with random replay (bottom), and reconnect to the camera directly. In the latter case, the primary hit (and therefore the pixel index) is determined by the random seed, which means we cannot shift to arbitrary primary hits. For this reason, we store these paths in separate *caustic reservoirs*, which do not interact with standard reservoirs used for all other paths.

that generated the original path's next direction (i.e., apply a random replay shift), and keep tracing the shifted path.

Light tracing ($t \leq 1$), non-caustic. For $t = 1$ techniques, light subpaths are connected directly to the camera. As we only perform bidirectional connections between rough vertices, this implies the primary hit x_1 is rough. If the secondary hit x_2 is also rough, then the path is non-caustic and a spatial shift is possible. We apply a reverse hybrid shift which shifts the light subpath using random replay until the secondary hit x_2', then reconnects to the new primary hit x_1' (see Figure 4, top). If the new primary hit x_1' is not rough, reconnection is not possible and the shift fails. Just like the forward hybrid shift used for camera subpaths, our reverse hybrid always reconnects the first two consecutively-rough vertices from the camera, which are always the primary and secondary hits $x_1 \leftrightarrow x_2$ as we only use this shift for non-caustic paths.

Despite replacing the last BSDF direction sampling with a reconnection to the primary hit, the technique index does not change: The shift mapping shifts the extended path $(\bar{x}, \tau)$ into another extended path $(T_\tau(\bar{x}), \tau)$ inside the same layer of the extended path space. Our shift mappings never change the technique index.

Light tracing ($t \leq 1$), caustic. For caustic paths, the secondary hit x_2 is non-rough, making reconnection impossible. Prior work defined shifts on such paths (e.g., Lehtinen et al.'s [2013] manifold shift), but they are poor fits for real-time, requiring expensive optimization procedures and per-vertex storage. We instead shift caustics temporally via random replay only (see Figure 4, bottom).

Unlike our shifts for other techniques, this changes the primary hit x_1 and thus the path's pixel filter contribution (W_e in Equation 2). To best preserve the sample's contribution, the sample should be shifted to the pixel where the pixel filter contribution is largest. Our shift mapping accomplishes this by projecting the shifted primary hit x_1' onto the screen; the shifted sample is stored at this new pixel. This may shift multiple caustic samples to the same pixel, and GRIS automatically selects one of them. This essentially forward-projects caustic paths to the next frame, using random replay instead of motion vectors, and GRIS combines the samples. We justify this mathematically in Appendix A.

5.1 Separating caustic reservoirs

As discussed above, we shift caustic $t \leq 1$ paths purely via random replay, which makes spatial shifts to arbitrary vertices impossible. Using spatially shifted reservoirs greatly degrades caustic quality as the non-caustic spatial neighbors are selected more often than low-probability caustic samples. For spatial reuse without this degradation, we store caustic samples in separate per-pixel reservoirs that are not spatially reused. These *caustic reservoirs* are identical to regular reservoirs except they can only contain caustic samples. The final image simply sums the per-pixel estimates from both reservoirs.

Mathematically, separating caustic reservoirs means having two $t \leq 1$ reservoirs, the non-caustic reservoir using $f = \hat{p} = 0$ for caustic paths and the caustic reservoir defining $f = \hat{p} = 0$ for non-caustic paths. Whether a $t \leq 1$ path is caustic is detected simply from the roughness of secondary hit x_2. Both reservoirs are populated from the same initial candidates but use different shift mappings: caustic paths use random replay only, while non-caustic paths use reconnection.

Confidence weights. ReSTIR reservoirs store confidence weights c_i that count how many samples each has processed. These weight the samples from different domains during spatiotemporal reuse (Equation 17). Importantly, confidence weights must remain independent of actualized samples for GRIS to remain unbiased; they should only count *possible* samples. Intuitively, c_i is unaffected by specific samples (if any) found each frame. For instance, we cannot update confidence weights based on if caustic samples happen to land in the current pixel, as that would correlate the weights with the samples.

We instead update c_i for caustic reservoirs using a proxy confidence c_v which is the weight of the prior frame's motion-vector-mapped caustic reservoir (as in standard temporal reuse):

$$c_i \leftarrow c_i + c_v, \tag{27}$$

where pixel v with weight c_v is found in the prior frame via (diffuse) motion vectors (e.g., $v = \text{motionVecs}[i]$). This gives unbiased results, as c_v is independent of the samples, and works well for static scenes. Diffuse motion vectors are not ideal, e.g., for animated caustics, and better quality would be achievable if we could define caustic motion vectors. Still, diffuse motion vectors effectively handle new pixels entering the image due to camera rotation. This only affects updates to c_i; the resampling MIS weight formulas remain unchanged.

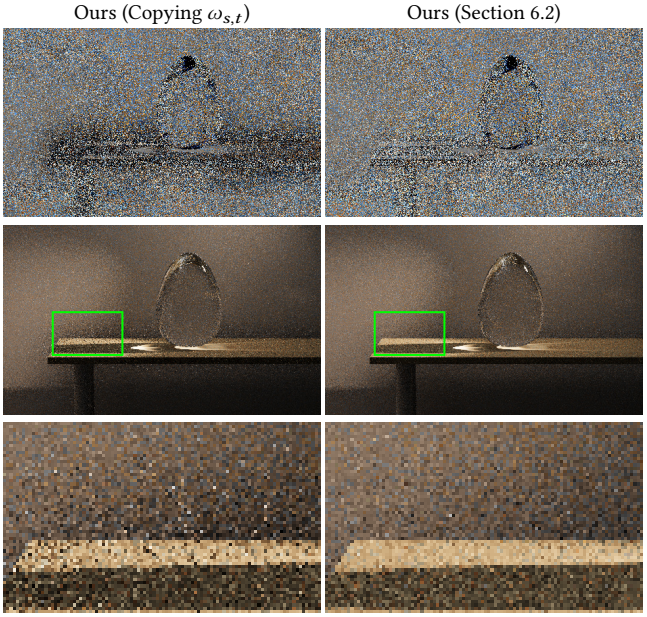

Fig. 5. Relative error offset (i.e., $0.5 + (\langle I \rangle - I)/I$) at 1 sample per pixel on Veach Bidir, comparing copying the MIS weight $\omega_{s,t}$ during spatial shifts (left) to correctly recomputing it for the shifted path (right) as in Section 6.2. Just copying the MIS weight causes bias; this is fixed by recalculating the shifted sample's MIS weight.

5.2 Shift Jacobians

Below we give formulas for our shift mapping Jacobians in the area measure. The Jacobian for a path shift is the product of Jacobians at each vertex. We denote quantities on the offset path as prime (e.g., x_1' is the shifted primary hit). For full derivations and further discussion, we refer to prior works [Hua et al. 2019; Lin et al. 2022].

For all reconnections, the Jacobian in the area measure is just 1. For random replay bounces on the camera subpath, the Jacobian is the ratio of area-measure sampling PDFs of the vertices:

$$\left|\frac{\partial x_i'}{\partial x_i}\right| = \left|\frac{\overrightarrow{p_i}}{\overrightarrow{p_i}'}\right|, \tag{28}$$

where $\overrightarrow{p_i}$ is the *forward* area-measure PDF of sampling x_i from x_{i-1}, and $\overrightarrow{p_i}'$ is the area-measure PDF of sampling x_i' from x_{i-1}'.

For the "reverse random replay" shift, which is applied to the light subpath, the Jacobians are identical but in the reverse direction:

$$\left|\frac{\partial x_i'}{\partial x_i}\right| = \left|\frac{\overleftarrow{p_i}}{\overleftarrow{p_i}'}\right|, \tag{29}$$

where $\overleftarrow{p_i}$ denotes the *reverse* area-measure PDF of sampling x_i from x_{i+1}, and $\overleftarrow{p_i}'$ is the area-measure PDF of sampling x_i' from x_{i+1}'.

6 FAST MIS WEIGHT COMPUTATION

After shifting a path, we must recompute its MIS weight ω_τ in the new domain. This is prohibitively expensive, requiring PDF evaluations at every path vertex. We could simply copy the candidate path's MIS weight, but this introduces bias (see Figure 5). Instead, we provide an efficient recursive formulation for recomputing ω_τ during reconnection. We also provide bounds on the bias associated with copying the MIS weight in Section 6.4.

We build on prior work [Georgiev 2012; van Antwerpen 2011] which optimizes the formulas for ω_τ allowing computation using only information at the connected subpath vertices y_{s-1} and z_{t-1}. Specifically, we store extra *partial MIS quantities* per vertex, and compute ω_τ from these quantities at y_{s-1} and z_{t-1}. Partials are computed incrementally, i.e., values at some vertex z_i derive from quantities at z_{i-1}, allowing efficient computation during sampling.

We aim to compute partial MIS quantities at the end of the camera subpath z_{t-1} only using information available during reconnection, without visiting the rest of the camera subpath. We store additional information with the subpath and use that to recover the partial MIS quantities needed to compute ω_τ during reconnection. Importantly, the size of the additional information is fixed and independent of the subpath length, which is key for performance.

If the scene changes or animates, these quantities must be updated. For temporal reuse, we already retrace the whole path, so we can compute the MIS weight as usual. During spatial reuse, we use our recursive MIS algorithm to recover only the information needed to compute the MIS weight, without traversing the entire path.

6.1 Recursive MIS

Here we briefly review the recursive MIS algorithm. We adopt the implementation and notation from Georgiev [2012], excluding vertex merging. We store two quantities d_i^{VC} and d_i^{p} (where d^{p} is the same as Georgiev's d^{VCM} term) at subpath vertices y_i, z_i. These quantities are initialized at the beginning of each subpath as:

$$\begin{gathered}\mathbf{y_1} : d_1^{\mathrm{p}} = \left(\frac{p_0^{\mathrm{connect}}}{p_0^{\mathrm{trace}}}\frac{1}{\overrightarrow{p_1}}\right)^\beta, \quad \mathbf{z_1} : d_1^{\mathrm{p}} = \left(\frac{p_0^{\mathrm{connect}}}{p_0^{\mathrm{trace}}}\frac{N_L}{\overrightarrow{p_1}}\right)^\beta, \\ d_1^{\mathrm{VC}} = \left(\frac{\overleftarrow{g}_0}{p_0^{\mathrm{trace}}\overrightarrow{p_1}}\right)^\beta, \qquad d_1^{\mathrm{VC}} = 0,\end{gathered} \tag{30}$$

and updated during path sampling as

$$d_i^{\mathrm{p}} = [x_{i-1}\ \text{nondelta}]\left(\frac{1}{\overrightarrow{p_i}}\right)^\beta, \tag{31}$$

$$d_i^{\mathrm{VC}} = \left(\frac{\overleftarrow{g_{i-1}}}{\overrightarrow{p_i}}\right)^\beta \left([x_{i-1}\ \text{nondelta}]\, d_{i-1}^{\mathrm{p}} + \left(\overleftarrow{p}_{i-2}^{\sigma}\right)^\beta d_{i-1}^{\mathrm{VC}}\right). \tag{32}$$

where, following the notation in Section 2, $\overrightarrow{p_i}$ is the area-measure PDF of sampling x_i from x_{i-1}, $\overleftarrow{p}_i^{\sigma}$ is the solid-angle-measure PDF of sampling x_i from x_{i+1}, $\overleftarrow{g_i}$ is the geometry term which converts $\overleftarrow{p}_i^{\sigma}$ to the area measure, and β is the power used by the MIS heuristic (i.e., $\beta = 1$ for balance heuristic or $\beta > 1$ for power heuristic). p^{connect} and p^{trace} denote the probability densities of the actual technique used to sample y_0, which depends on whether y_0 is used to start a new light subpath or connect to a camera subpath (e.g., for NEE). The bracket notation $[x_i\ \text{nondelta}]$ evaluates to 1 if the expression inside the bracket is true (i.e., if a delta BSDF was sampled at x_i), or 0 otherwise.

The full technique MIS weight is then recovered using

$$\omega_\tau = (\bar{w}_{s-1}(\bar{y}) + 1 + \bar{w}_{t-1}(\bar{z}))^{-1}, \tag{33}$$

where $\bar{w}_{s-1}$ and $\bar{w}_{s-1}$ are weights from the light and camera subpaths, computed depending on the technique as

- Vertex Connection ($s > 1, t > 1$)

$$\bar{w}_{s-1}(\bar{y}) = \left(\overleftarrow{p}_{s-1}\right)^\beta \left(d^{\mathrm{p}}_{s-1} + \left(\overleftarrow{p}^{\sigma}_{s-2}\right)^\beta d^{\mathrm{VC}}_{s-1}\right), \tag{34}$$

$$\bar{w}_{t-1}(\bar{z}) = \left(\overleftarrow{p}_{t-1}\right)^\beta \left(d^{\mathrm{p}}_{t-1} + \left(\overleftarrow{p}^{\sigma}_{t-2}\right)^\beta d^{\mathrm{VC}}_{t-1}\right), \tag{35}$$

- Camera Tracing ($s = 0$)

$$\bar{w}_{s-1}(\bar{y}) = 0, \tag{36}$$

$$\bar{w}_{t-1}(\bar{z}) = \left(p^{\mathrm{connect}}_{t-1}\right)^\beta d^{\mathrm{p}}_{t-1} + \left(p^{\mathrm{trace}}_{t-1}\right)^\beta \left(\overleftarrow{p}^{\sigma}_{t-2}\right)^\beta d^{\mathrm{VC}}_{t-1}, \tag{37}$$

- Next Event Estimation ($s = 1$)

$$\bar{w}_0(\bar{y}) = \left(\frac{\overleftarrow{p_0}}{p^{\mathrm{connect}}_0}\right)^\beta, \tag{38}$$

$$\bar{w}_{t-1}(\bar{z}) = \left(\frac{p^{\mathrm{trace}}_0(\bar{y})}{p^{\mathrm{connect}}_0(\bar{y})}\right)^\beta \left(\overleftarrow{p}_{t-1}\right)^\beta \left(d^{\mathrm{p}}_{t-1} + \left(\overleftarrow{p}^{\sigma}_{t-2}\right)^\beta d^{\mathrm{VC}}_{t-1}\right), \tag{39}$$

- Light Tracing ($t = 1$)

$$\bar{w}_{s-1}(\bar{y}) = \left(\frac{p^{\mathrm{trace}}_0(\bar{z})}{p^{\mathrm{connect}}_0(\bar{z})} \frac{\overleftarrow{p}_{s-1}}{N_L}\right)^\beta \left(d^{\mathrm{p}}_{s-1} + \left(\overleftarrow{p}_{s-1}\right)^\beta d^{\mathrm{VC}}_{s-1}\right), \tag{40}$$

$$\bar{w}_0(\bar{z}) = 0. \tag{41}$$

For further explanation and derivation, see Georgiev [2012].

6.2 Recursive Reconnection MIS

The quantities d^{VC}_{t-1} and d^{p}_{t-1} are required to compute the MIS weight for a path. However, during reconnection, we only retrace the path up to the reconnection vertex x_r. A straightforward approach is to visit the rest of the camera subpath after the reconnection vertex just to compute d^{VC}_{t-1} and d^{p}_{t-1}, however this makes the algorithm prohibitively expensive.

We instead derive a formulation for computing d^{VC}_{t-1} and d^{p}_{t-1} using information available at the reconnection vertex x_r (where $r < t - 1$), without visiting the rest of the camera subpath.

Our algorithm works by computing and storing extra quantities when the camera subpath is first sampled. These quantities are:

$$\bar{\gamma} = \left(\frac{\overleftarrow{g}_{t-2}}{\overrightarrow{p}_{t-1}}\right)^\beta, \tag{42}$$

$$\bar{\lambda}^{\mathrm{VC}} = \left(\frac{\overleftarrow{p}_r}{\overrightarrow{g}_{r+1}}\right)^\beta \prod_{i=r+1}^{t-3} \left(\frac{\overleftarrow{p}_i}{\overrightarrow{p}_{i+1}}\right)^\beta, \tag{43}$$

$$\bar{\lambda}^{\mathrm{p}} = [x_{r+1} \text{ nondelta}] \prod_{i=r+1}^{t-3} \left(\frac{\overleftarrow{p}_i}{\overrightarrow{p}_{i+1}}\right)^\beta, \tag{44}$$

$$\bar{\sigma} = \sum_{i=r+2}^{t-2} [x_i \text{ nondelta}]\, d^{\mathrm{p}}_i \prod_{j=i}^{t-3} \left(\frac{\overleftarrow{p}_j}{\overrightarrow{p}_{j+1}}\right)^\beta. \tag{45}$$

In our implementation, we compute these quantities recursively by updating them as the path is traced. This is easily done by adding d^{p}_i into $\bar{\sigma}$ at each bounce, then multiplying $\bar{\lambda}^{\mathrm{VC}}$, $\bar{\lambda}^{\mathrm{p}}$, and $\bar{\sigma}$ by the ratio of reverse and forward PDFs at each bounce. We provide an implementation of this process in Python as supplemental material.

6.3 Computing d^{VC}_{t-1}

If the reconnection vertex is the last vertex on the camera subpath (i.e., $r = t - 1$), then Equation 32 can be used without further modification.

If the reconnection vertex is the second-to-last vertex on the camera subpath (i.e., $r = t - 2$), we use a slightly modified version of Equation 32 which just has the geometry term separated from $\overrightarrow{p}^{\sigma}_{r+1}$ in the denominator, as it does not change during reconnection:

$$d^{\mathrm{VC}}_{t-1} = \left(\frac{\overleftarrow{g}_r}{\overrightarrow{p}^{\sigma}_{r+1} \overrightarrow{g}_{r+1}}\right)^\beta \left(d^{\mathrm{p}}_r + \left(\overleftarrow{p}^{\sigma}_{r-1}\right)^\beta d^{\mathrm{VC}}_r\right). \tag{46}$$

In this case, all quantities are computed during reconnection except the ratio $\overleftarrow{g}_r / \overrightarrow{g}_{r+1}$ which is computed and cached during initial sampling, and updated when the scene changes.

For the general case of $r < t - 2$ we use the additional quantities defined above:

$$d^{\mathrm{VC}}_{t-1} = \bar{\gamma} \left(\left(\frac{1}{\overrightarrow{p}^{\sigma}_{r+1}}\right)^\beta \bar{\lambda}^{\mathrm{VC}} \left(d^{\mathrm{p}}_r + \left(\overleftarrow{p}^{\sigma}_{r-1}\right)^\beta d^{\mathrm{VC}}_r\right) + \left(\frac{1}{\overrightarrow{p}^{\sigma}_{r+1}}\right)^\beta \left(\frac{1}{\overrightarrow{g}_{r+1}}\right)^\beta \bar{\lambda}^{\mathrm{p}} + \bar{\sigma} \right). \tag{47}$$

In our implementation, we also cache all other quantities needed to compute the weights $\bar{w}_{t-1}$ and $\bar{w}_{s-1}$ used to compute the final MIS weight (Equation 33).

6.4 Bias from MIS weight reuse

Faster rendering can be achieved by simply copying the MIS weight from the base path instead of recomputing it. This introduces bias, but eliminates the need for our recursive reconnection MIS algorithm. In practice, we observe a small darkening bias near corners and shadow boundaries. In our supplementary document, we show the relative bias is bounded by the MIS weight error, i.e.,

$$\frac{|\mathrm{Bias}|}{I} \leq \epsilon_\omega,$$

where ϵ_ω bounds the relative error between the correct MIS weight and the copied MIS weight from the original domain.

7 IMPLEMENTATION

We implemented bidirectional path tracing in the Falcor rendering framework [Kallweit et al. 2022] using the Slang shading language. At a high-level, our initial sampling resembles SmallVCM [Georgiev et al. 2012], but without vertex merging. We first sample a fixed number of light subpaths via Algorithm 1, parallelizing over subpaths. Initial vertices y_0 are selected according to each light's emissive power. After sampling light subpaths, we sample camera subpaths,

parallelizing over pixels, connecting them to light subpaths in Algorithm 2. Finally, pixels reuse full paths from spatiotemporal neighbors, using Algorithm 3 to shift paths between pixel domains.

7.1 Initial sampling

Light subpath sampling. In Algorithm 1, light subpaths are traced and all connectable light subpath vertices are atomically appended to the LVC (line 8). These vertices are also connected to the camera to form full paths (line 9), corresponding to the light tracing technique. These paths must be resampled into the destination pixel, however reservoir merging is not an atomic operation. To avoid race conditions from merging, we instead insert the key-value pair (k, r) into a parallel multimap which we refer to as the Light Reservoir Map (LRM), where k is the path's pixel index and r is a new reservoir containing the sampled path (line 10). Afterwards, the LRM is efficiently sorted on the GPU via prefix sum operations, following Boissé's [2021] parallel hash map (line 11).

Camera subpath sampling. After light subpath sampling, Algorithm 2 samples camera subpaths and connects them to random light subpath vertices to form full paths. For NEE (where $s=1$), we sample light primitives according to their emissive power. For subpath connection techniques (where $s>1$), we select light subpath vertices from the LVC uniformly, corresponding to $p_L = N_L/|LVC|$ in Equation 25.

Merging light tracing paths. After camera subpath sampling, Algorithm 2 merges the reservoirs from light tracing in each pixel (lines 15-20). Conceptually, light tracing reservoirs are merged into *every* pixel's reservoir, and we only use the LRM as an acceleration structure to skip zero-contribution paths within each pixel. During this step, we separate reservoirs containing caustic paths as mentioned in Section 5.1, corresponding to the check on line 17 in Algorithm 2. The full image is the sum of caustic and noncaustic contributions, which we recover by summing the estimates from our caustic and noncaustic reservoirs.

Confidence weights. The confidence weight c_i of the resulting reservoir is always 1, regardless of how many paths Algorithm 2 found. This is because confidence weights cannot depend on the random samples, and must be updated deterministically based on the sampling process. In practice, we implement this by initializing our reservoirs with $c_i = 1$, and only updating c_i during spatiotemporal reuse.

7.2 Spatiotemporal sample reuse

After initial sampling, pixels reuse samples from neighbor pixels and the prior frame. For both spatial and temporal reuse, Algorithm 3 is used to shift paths to new primary hits. For caustic samples, the primary hit is unused, and the shifted path instead contributes to an arbitrary pixel based on the reverse random replay shift. For this reason, we temporally shift caustic paths separately, merging them into the new pixels based on the random replay shift. As in initial sampling, this occurs in two passes using the LRM to efficiently sort the shifted paths into pixels.

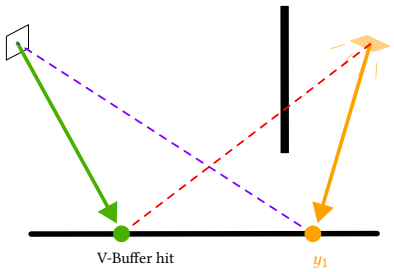

Fig. 6. **Light tracing hits and V-Buffer hits**. It is possible to connect light subpaths to the camera (purple dashed line) which cannot be shifted to the visibility buffer hit for the pixel due to differences in visibility (red dashed line).

Integration measures. Integrating with respect to surface area (as in Equation 1) introduces geometry terms $G(x_i \leftrightarrow x_{i+1})$ which convert differential areas at x_{i+1} to differential projected solid-angles at x_i. Many geometry terms cancel when evaluating f/p (Equation 3), as p also has geometry terms after converting sampling probability densities to area measure. Renderers often omit cancelled geometry terms entirely, essentially integrating with respect to solid-angle everywhere. This reduces computation and improves numerical stability where geometric inverse-squared terms go to zero.

For sample reuse, we use the unweighted integrand (f without dividing by p), so we cannot rely on this cancellation. To avoid numerically-unstable geometry terms, we integrate with respect to solid-angle at each vertex; this slightly changes our Jacobians. We give solid-angle parameterizations of shift Jacobians in Appendix B.

Stable but biased temporal reuse of light traced samples. Many ReSTIR implementations [Bitterli et al. 2020; Lin et al. 2022] store primary hits in a visibility buffer [Burns and Hunt 2013]; this works well for unidirectional path tracing. However, bidirectional path tracing connects light paths to the camera directly, which makes it possible to sample paths occluded from these primary hits (Figure 6). These samples must be included for an unbiased estimate, but they degrade temporal reuse as shifting them to the next frame's primary hit immediately fails. To fix this, we add a simple rejection heuristic to reject non-caustic $t \leq 1$ paths that cannot be shifted to the primary hit. This greatly improves temporal stability, albeit with a negligible darkening bias in penumbra regions due to missing samples. A better solution might apply Zhang et al.'s [2024] concurrent Area ReSTIR to reuse samples within a pixel.

Lightweight BDPT. A simple way to improve performance is to omit techniques involving subpath connections. We refer to this version of our algorithm as "lightweight" (LW) BDPT. Specifically, LW BDPT refers to the combination of $s = 0$ (PT), $s = 1$ (NEE) and $t = 1$ (light tracing) techniques. This is an appealing middle ground as it provides high quality caustic sampling with a lower cost than full BDPT. We show results for this algorithm in Figure 11.

8 RESULTS

In Figure 11, we show several scenes [Bitterli 2016] that are challenging with unidirectional path tracing. All results use an RTX 4090 GPU, a Ryzen 7 3700x CPU, at a resolution of 1920×1080. We

Algorithm 1: SampleLightPaths()

```
1  LVC ← ∅ // Light Vertex Cache
2  LRM ← ∅ // Light Reservoir Map
3  parallel for i ← 1 to N_L
4      y_0 ← InitializeLightPath()
5      for j ← 1 to N_bounces − 1 do
6          y_j ← TraceNewVertex(y_{j−1})
7          if y_j is rough then
8              LVC.Append(y_j) // Append vertex to the LVC
9              (r, k) ← ConnectToCamera(y_j)
10             LRM.Insert(k, r) // Append reservoir r to pixel k
11 LRM.Sort() // Sort reservoirs into per-pixel lists
12 return (LVC, LRM)
```

Algorithm 2: SampleInitialPaths(LVC, LRM)

```
1  parallel for i ← 1 to N_pixels
2      r_i ← InitializeReservoir()
3      x_0 ← InitializeCameraPath()
4      for j ← 1 to N_bounces do
           // Sample direction and trace ray
5          x_j ←TraceNewVertex(x_{j−1})
6          if x_j is rough then
               // Connection strategies (s ≥ 1 techniques)
7              r_NEE ← ConnectToRandomLight(x_j)
8              r_LVC ← ConnectToRandomLVC(LVC, x_j)
9              r_i.Merge( r_NEE )
10             r_i.Merge( r_LVC )
11         if x_j is emissive then
               // Found full path (s = 0 technique)
12             r ← EvalEmission(x_j)
13             r_i.Merge( r )
14
       // Merge reservoirs from light tracing
15     c_i ← InitializeReservoir() // Caustic reservoir
16     foreach r_LT ∈ LRM[i] do
17         if r_LT.sample is caustic then
18             c_i.Merge(r_LT)
19         else
20             r_i.Merge(r_LT)
21     OutputReservoirs[i] ← r_i
22     OutputCausticReservoirs[i] ← c_i
```

Algorithm 3: ShiftPath(basePath, newPrimaryHit)

```
   // This shows a temporal shift where camera, geometry, and
      light can change. For a spatial shift, some ray traces
      (inside functions marked with *) can be skipped due to
      unchanged subpaths which have partial path throughputs
      available in the reservoir.
1  ȳ ← ShiftLightSubpath*(basePath)
2  if Caustic t ≤ 1 case then
3      return ConnectToCamera(ȳ)
4  x̄_0...x_{r−1} ← ShiftCameraPrefix(basePath, newPrimaryHit)
5  if basePath.cameraBounces == r − 1 then
6      z̄ ← x̄_0...x_{r−1}
7  else
8      x_r ← basePath.CameraRcVertex
       // Connect to the 1st suffix vertex
9      x̄_0...x_r ← Connect(x̄_0...x_{r−1}, x_r)
10     if basePath.cameraBounces > r then
11         x̄_0...x_{r+1} ← TraceRay*(x̄_0...x_r,
             basePath.CameraRcVertexScatterDirection)
           // Replay the rest of the suffix
12         z̄ ← ExtendPathByReplay*(basePath, x̄_0...x_{r+1})
13     else
14         z̄ ← x̄_0...x_r
15 if basePath.numLightSubpathVertices == 0 then
16     return z̄
17 else
18     return ConnectToLightSubpath(z̄, ȳ)
```

compare numerical errors using mean absolute percentage error (MAPE)[3].

We sample $N_L = 1920 \cdot 1080 \approx 2M$ light subpaths, unless otherwise noted. All results use a maximum path length of 20 bounces, or 8 diffuse bounces (whichever is reached first), where a "diffuse bounce" refers to sampling a BSDF lobe with roughness above the connectability threshold. We classify vertices with roughness greater than 0.08 as connectable, which is the minimum roughness our material system supports. See Figure 7 for comparison with other thresholds. We apply Russian roulette by randomly terminating paths inside TraceNewVertex (Algorithm 2, line 5) according to the probability

$$p_{\text{terminate}}(x_i) = 0.2 \cdot \text{rgh}(x_i),$$

where $\text{rgh}(x_i)$ returns roughness of the material (not just the sampled BSDF lobe) at vertex x_i[4]. We limit confidence weights c_i to a maximum of 20 in all experiments.

Our strongest results are on scenes lit primarily by caustics, e.g., the Bathroom and Sponza that only have emissive geometry inside glass light fixtures. The glass fixtures in Sponza and Bathroom cause next event estimation ($s=1$) to fail, which makes BSDF sampling ($s = 0$) the only viable technique for ReSTIR PT. The glass also causes failed reconnections, forcing ReSTIR PT to use (unidirectional) random replay for most scene lighting; this introduces obvious blotchy correlation artifacts. In contrast, our work efficiently samples light paths through the glass, greatly improving sampling of LSD*E paths and better resolving scene caustics.

In the Veach Bidir scene, the caustics on the wall are challenging without bidirectional techniques. Even at high sample counts, unidirectional path tracing fails to resolve them. Additionally, when

[3]We use MAPE = mean $(|I - I_{\text{ref}}| / (I_{\text{ref}} + 0.01 \cdot \text{mean}(I_{\text{ref}}))$

[4]$p_{\text{terminate}}$ cannot rely on camera subpath information (path length, throughput, etc.) since this information is not known during light tracing [Georgiev 2012; van Antwerpen 2011].

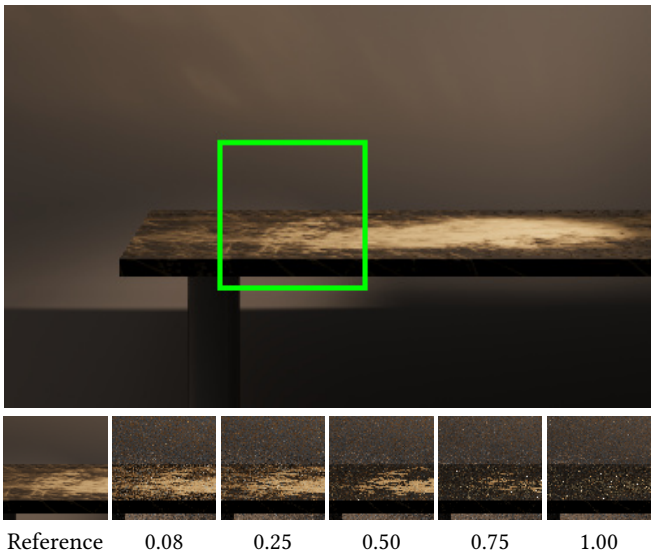

Fig. 7. The Veach Bidir scene with a metallic table with a roughness texture, rendered with different roughness thresholds. Using too large of a threshold disables bidirectional connections (including NEE) on rough surfaces, and leads us to select less efficient shift mappings, which produces more noise.

the light is animated (Figure 9), our temporal caustic shift enables efficient resampling between frames, greatly improving the result.

For diffuse lighting, as in the White Room, our improvement over ReSTIR PT is reduced. Reconnection works well on these scenes, enabling spatial reuse to share many light paths between pixels. However, BDPT still improves the candidate distribution, so our method still shows benefits, especially at low sample counts where candidate path distribution especially matters.

In Figure 8, we show a cut-down version of our algorithm without camera connections ($t \leq 1$ techniques), similar to prior methods [Liu and Gan 2023; Nabata et al. 2020]. Importantly, $t \leq 1$ techniques excel at sampling caustic paths, making it the dominant technique on these scenes. Even with only the $t = 1$ technique, most illumination in the scene is well-sampled, except for regions where the primary hits are not rough, which makes connecting to the camera impossible.

Our algorithm costs roughly twice as much as ReSTIR PT at the same sample count. This is mostly because BDPT traces roughly twice as many rays as regular path tracing, due to light tracing and subpath connections. Our algorithm also employs additional data structures (the LVC and caustic reservoirs) and a larger path reservoir to store light subpath and recursive reconnection MIS information, increasing register and memory pressures.

Our per-pixel reservoirs have size 244B if using our recursive reconnection MIS, and 160B otherwise. The size of LVC vertices is 112B. We give the full contents of our reservoir and light vertex data structures in the supplementary document. For 1920×1080 renders, the total storage cost of our full unbiased algorithm is 8.2GB. We did not attempt to optimize storage.

Figure 12 shows error (MAPE) over time for an offline variant of our algorithm, compared to ReSTIR PT. As Lin et al. [2022] note, temporal resampling is less effective for offline rendering as temporal correlations hurt offline convergence (when averaging multiple frames). Instead, our offline variant uses spatial resampling only. As BDPT has a higher-quality initial candidate distribution, our error is lower than Lin et al. [2022].

Spatial reuse from pixels with very different target functions can sometimes reduce efficiency [Pan et al. 2024; Tokuyoshi 2023]. While we observe better offline results than standard BDPT in all of our scenes when measured *per-sample*, in some scenes the reduced efficiency no longer compensates for the added computational cost of resampling, leading to standard BDPT converging faster. This suggests a careful study on how to best benefit from ReSTIR algorithms in the offline context.

9 CONCLUSION

We presented a method to combine bidirectional path tracing with ReSTIR. By combining spatiotemporal reuse and path-space sampling, we enable interactive rendering of scenes with caustics and complex occlusion; this was previously limited to offline renderers.

9.1 Limitations and future work

Not all scenes benefit from light tracing. In these cases, our method's efficiency may decrease below ReSTIR PT, due to being more expensive (Figure 10). However, this could potentially be improved with better light subpath sampling strategies leveraging camera information [Grittmann et al. 2018; Vorba et al. 2014].

We did not optimize our light subpath shifts. Our implementation shifts camera and light subpaths in the same GPU dispatch, dealing with caustic and non-caustic cases separately. A more optimal implementation could reduce divergence by shifting camera and light subpaths in separate dispatches.

As discussed in Section 5.1, we use a proxy confidence weight c_v during temporal reuse, where v is found via motion vectors. This is suboptimal for animation, as animated caustics do not follow the diffuse motion vectors. We did not explore alternative schemes to update confidence weights. Future work could explore using information from all caustic reservoirs to better update the confidence weights.

Specular-diffuse-specular (SDS) paths such as the bathroom mirror reflections are still challenging with our method. This is a known limitation of bidirectional sampling [Veach 1997]. Future work could apply vertex merging to capture these effects. Manifold shifts could also help reuse in hard situations.

ACKNOWLEDGMENTS

This work was partially funded Google and Adobe's gifts, and the NSF IIS grant 2238839. We thank Aaron Lefohn for the discussions and support, and the anonymous reviewers for their constructive feedback.

REFERENCES

Benedikt Bitterli. 2016. Rendering resources. https://benedikt-bitterli.me/resources/.
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* 39, 4 (2020), 148:1–148:17. https://doi.org/10.1145/3386569.3392481

Fig. 8. Without light tracing ($t \leq 1$) techniques (as in [Liu and Gan 2023; Nabata et al. 2020]), scenes lit by caustics are difficult to sample. Light tracing techniques very efficiently sample caustics, while standard path tracing struggles to find such paths.

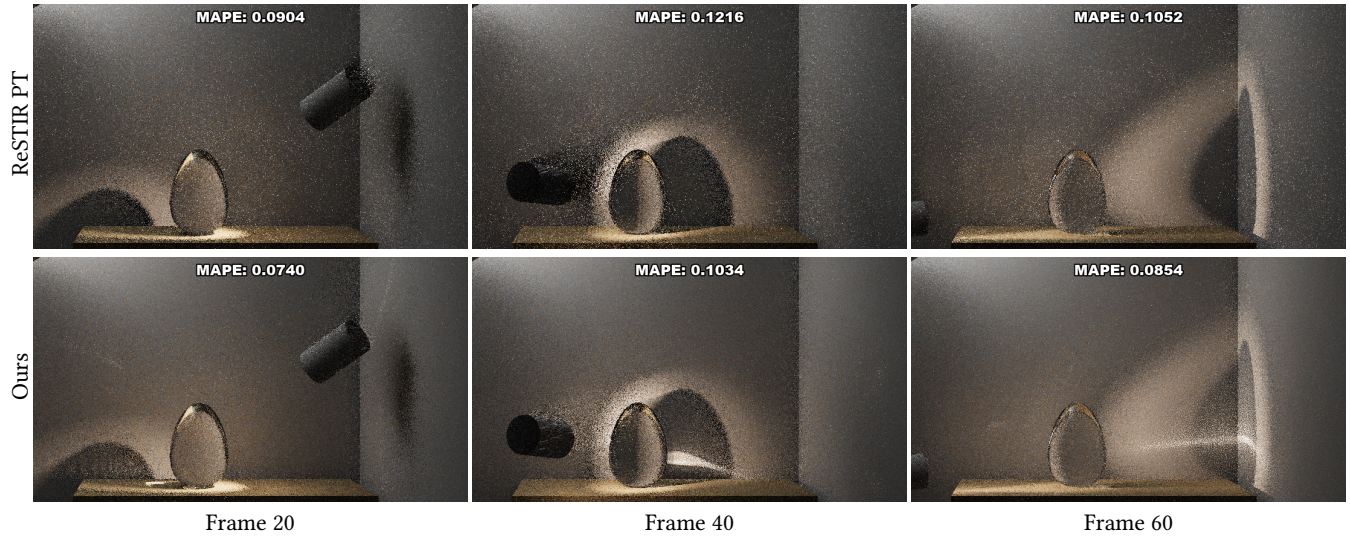

Fig. 9. The Veach Bidir scene with animated caustics rendered with 1 sample per pixel. The light moves from right to left during animation, focusing light through the glass egg. With light subpath random replay and separate caustic reservoirs, our work temporal resamples the caustics, even capturing smaller details on the wall. Note: the light around the egg's shadow captured by ReSTIR PT is direct illumination, not a caustic.

Fig. 10. Results with 1 sample per pixel in the Bistro Interior. In this scene, light tracing struggles to sample high-contribution paths, especially the caustics on the wall, as most scene lights do not contribute to the image. Our method performs similarly to unidirectional ReSTIR in this case.

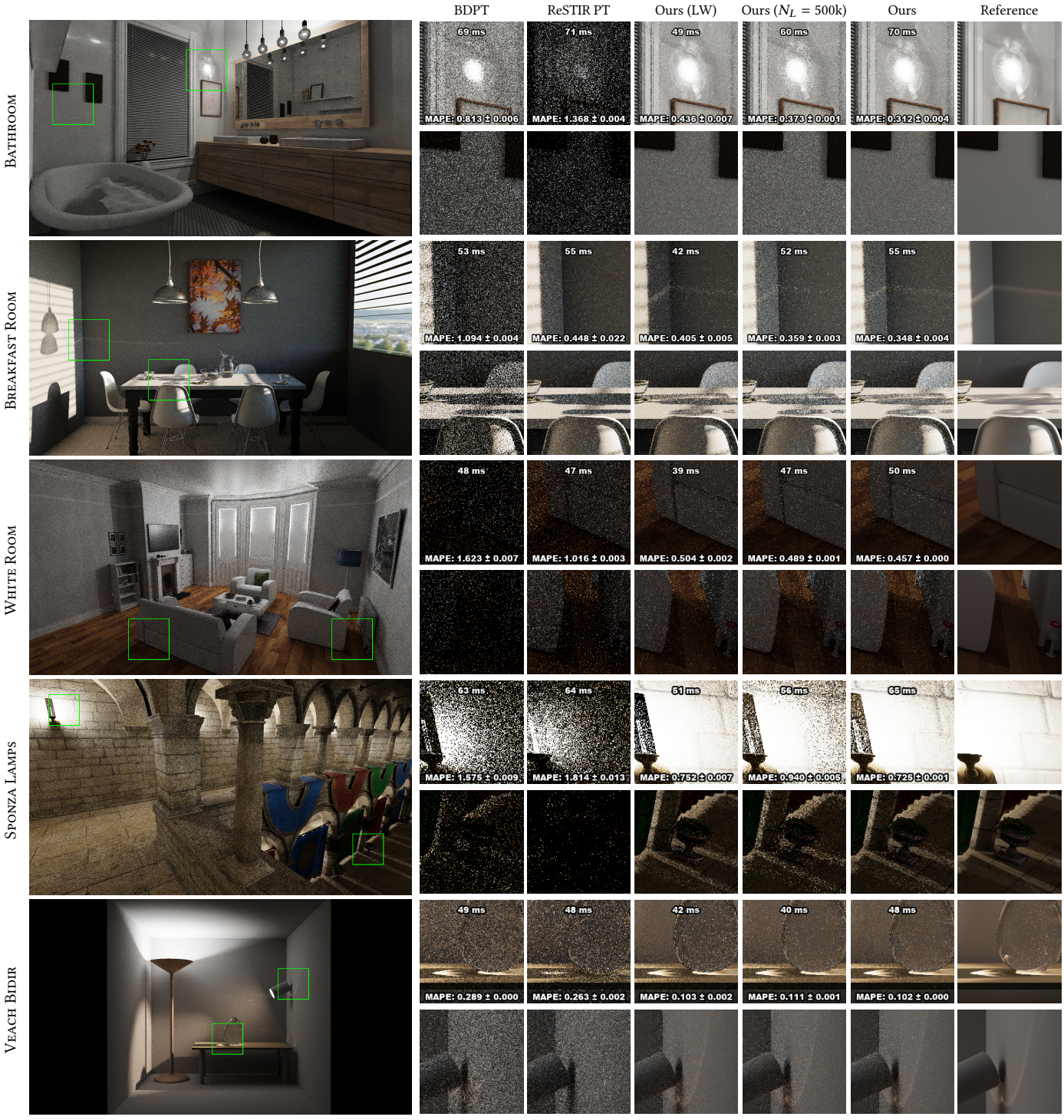

Fig. 11. Results on various scenes. All scenes are modified to be more challenging for unidirectional path tracing (except Veach). Sponza and Bathroom are lit only by filaments inside glass light fixtures. Breakfast Room has glass dishes, adding caustics on the table and wall. The White Room is lit from the outside, but the blinds are closed, blocking much of the window. We show results at 1spp, after warming the reservoirs with 200 frames of reuse. We perform equal-time comparisons by increasing initial candidate count for other methods. Unless otherwise specified, we set the number of light subpaths N_L to the number of pixels (1920×1080). The lamp in Sponza comes from Sketchfab user hectopod, under CC-Attribution.

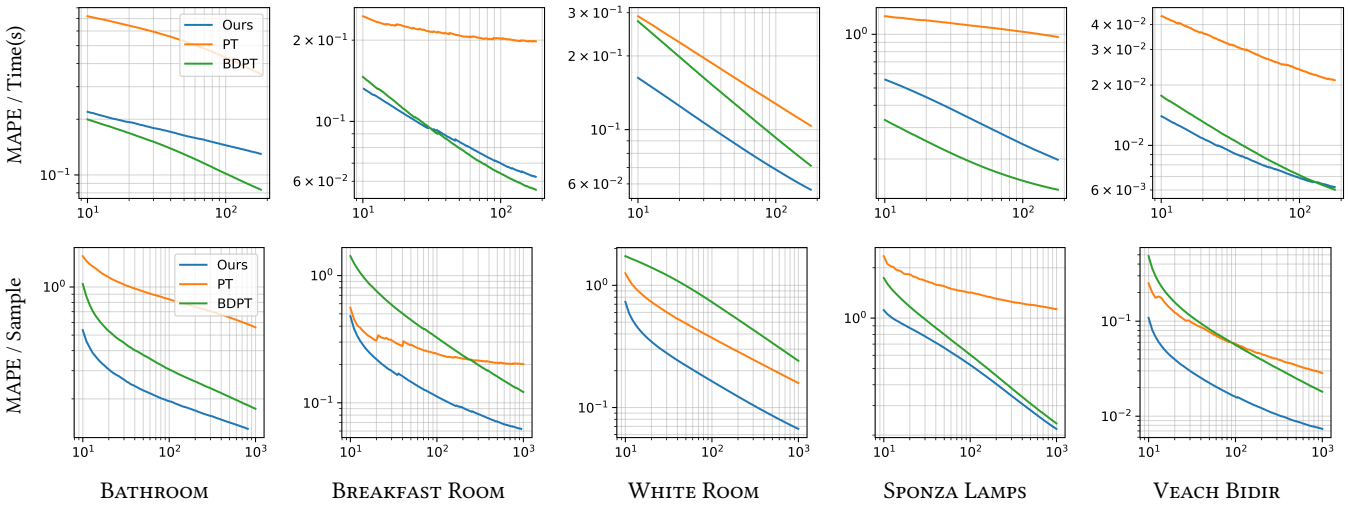

Fig. 12. Convergence graphs comparing standard BDPT to offline ReSTIR variants (3 spatial reuse passes with 6 candidates each, no temporal reuse) on our scenes. Our method has a lower error than ReSTIR PT due to the improved candidate distribution from bidirectional sampling. Standard BDPT is able to accumulate more samples in less time, however the per-sample error is lower with resampling.

Guillaume Boissé. 2021. World-Space Spatiotemporal Reservoir Reuse for Ray-Traced Global Illumination. *SIGGRAPH Asia 2021 Technical Communications* (2021). https://doi.org/10.1145/3478512.3488613

Christopher A. Burns and Warren A. Hunt. 2013. The Visibility Buffer: A Cache-Friendly Approach to Deferred Shading. *Journal of Computer Graphics Techniques (JCGT)* 2, 2 (12 August 2013), 55–69. http://jcgt.org/published/0002/02/04/

Tomáš Davidovič, Jaroslav Křivánek, Miloš Hašan, and Philipp Slusallek. 2014. Progressive Light Transport Simulation on the GPU: Survey and Improvements. *ACM Transactions on Graphics (TOG)* 33, 3 (2014), 1–19. https://doi.org/10.1145/2602144

Iliyan Georgiev. 2012. Implementing Vertex Connection and Merging. *Technical Report. Saarland University* (2012).

Iliyan Georgiev, Jaroslav Křivánek, Tomáš Davidovič, and Philipp Slusallek. 2012. Light Transport Simulation with Vertex Connection and Merging. *ACM Transactions on Graphics (TOG)* 31, 6, Article 192 (Nov. 2012), 10 pages. https://doi.org/10.1145/2366145.2366211

Pascal Grittmann, Iliyan Georgiev, Philipp Slusallek, and Jaroslav Křivánek. 2019. Variance-aware multiple importance sampling. *ACM Trans. Graph.* 38, 6, Article 152 (Nov. 2019), 9 pages. https://doi.org/10.1145/3355089.3356515

Pascal Grittmann, Arsène Pérard-Gayot, Philipp Slusallek, and Jaroslav Křivánek. 2018. Efficient Caustic Rendering with Lightweight Photon Mapping. *Computer Graphics Forum* 37, 4 (2018). EGSR '18.

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. https://doi.org/10.1111/cgf.13652

James T. Kajiya. 1986. The Rendering Equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '86)*. Association for Computing Machinery, New York, NY, USA, 143–150. https://doi.org/10.1145/15922.15902

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

René Kern, Felix Brüll, and Thorsten Grosch. 2024. ReSTIR FG: Real-Time Reservoir Resampled Photon Final Gathering. In *Eurographics Symposium on Rendering*, Eric Haines and Elena Garces (Eds.). The Eurographics Association. https://doi.org/10.2312/sr.20241155

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. https://doi.org/10.1145/2766997

Eric P. Lafortune and Yves D. Willems. 1993. Bi-Directional Path Tracing. In *Proceedings of Compugraphics*. 145–153. https://graphics.cs.kuleuven.be/publications/BDPT/

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. https://doi.org/10.1145/2461912.2461943

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* 41, 4 (2022), 75:1–75:23. https://doi.org/10.1145/3528223.3530158

Fuyan Liu and Junwen Gan. 2023. Light Subpath Reservoir for Interactive Ray-Traced Global Illumination. *Comput. Graph.* 111, C (apr 2023), 37–46. https://doi.org/10.1016/j.cag.2023.01.004

Marco Manzi, Markus Kettunen, Miika Aittala, Jaakko Lehtinen, Frédo Durand, and Matthias Zwicker. 2015. Gradient-Domain Bidirectional Path Tracing. In *Eurographics Symposium on Rendering - Experimental Ideas & Implementations*, Jaakko Lehtinen and Derek Nowrouzezahrai (Eds.). The Eurographics Association. https://doi.org/10.2312/sre.20151168

Kosuke Nabata, Kei Iwasaki, and Yoshinori Dobashi. 2020. Two-stage Resampling for Bidirectional Path Tracing with Multiple Light Sub-paths. *Computer Graphics Forum* 39 (2020). https://doi.org/10.1111/cgf.14139

Anthony Pajot, Loïc Barthe, Mathias Paulin, and Pierre Poulin. 2011. Combinatorial bidirectional path-tracing for efficient hybrid CPU/GPU rendering. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 315–324.

Xingyue Pan, Jiaxuan Zhang, Jiancong Huang, and Ligang Liu. 2024. Enhancing Spatiotemporal Resampling with a Novel MIS Weight. *Computer Graphics Forum* 43, 2 (2024), e15049. https://doi.org/10.1111/cgf.15049 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.15049

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. *Physically Based Rendering: From Theory to Implementation (3rd ed.)* (3rd ed.). Morgan Kaufmann Publishers Inc. 1266 pages.

Stefan Popov, Ravi Ramamoorthi, Fredo Durand, and George Drettakis. 2015. Probabilistic Connections for Bidirectional Path Tracing. *Computer Graphics Forum* 34, 4 (jul 2015), 75–86.

Fujia Su, Sheng Li, and Guoping Wang. 2022. SPCBPT: Subspace-based Probabilistic Connections for Bidirectional Path Tracing. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–14. https://doi.org/10.1145/3528223.3530183

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

Yusuke Tokuyoshi. 2023. Efficient Spatial Resampling Using the PDF Similarity. *Proc. ACM Comput. Graph. Interact. Tech.* 6, 1, Article 4 (May 2023), 19 pages. https://doi.org/10.1145/3585501

D. van Antwerpen. 2011. Recursive MIS Computation for Streaming BDPT on the GPU. (2011).

Eric Veach. 1997. *Robust Monte Carlo Methods for Light Transport Simulation*. Ph. D. Dissertation.

Eric Veach and Leonidas J. Guibas. 1995a. Bidirectional Estimators for Light Transport. In *Photorealistic Rendering Techniques*. Springer, 145–167. https://doi.org/10.1007/978-3-642-87825-1_11

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. https://doi.org/10.1145/218380.218498

Jiří Vorba, Ondřej Karlík, Martin Šik, Tobias Ritschel, and Jaroslav Křivánek. 2014. On-line Learning of Parametric Mixture Models for Light Transport Simulation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)* 33, 4 (aug 2014).

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 Courses*. 1–38.

Song Zhang, Daqi Lin, Markus Kettunen, Cem Yuksel, and Chris Wyman. 2024. Area ReSTIR: Resampling for Real-Time Defocus and Antialiasing. 43, 4 (July 2024). https://doi.org/10.1145/3658210

A TEMPORAL CAUSTIC REUSE

The pixel a caustic path ($t \leq 1$) hits is determined by its primary hit vertex x_1. We shift caustic paths temporally by random replay, i.e., re-tracing the path in the new frame with the same random numbers. This means its pixel may change. This initially seems incompatible with GRIS, which requires pre-selecting the reuse domains, and the selection cannot depend on their samples. To resolve this, we reformulate the reuse process as follows.

First, the *domain* of current and previous pixels covers the whole image, but their *support* covers only the current pixel i. The target function $\hat{q}$ is zero outside of the pixel, e.g., by a pixel filter h,

$$\hat{q}_i(\hat{x}) = h_i(\bar{x})\hat{p}(\hat{x}), \tag{48}$$

where $\hat{p}$ is the target function, $\bar{x}$ is the path of the path-technique pair $\hat{x}$, and $h_i(\bar{x})$ evaluates the pixel filter at the subpixel location of $\bar{x}$'s primary hit.

Second, in candidate sampling, all N_L $t \leq 1$ paths are conceptually treated as candidates for each pixel i, leading to resampling weights

$$w_i = \frac{1}{N_L} h_i(\hat{X}_i)\hat{p}_i(\hat{X}_i)W_{\hat{X}_i}, \tag{49}$$

where pixel filter h_i automatically discards samples not in pixel i, and $\hat{p}$ discards non-caustic paths. The initial sampling's output $\hat{X}$, now a caustic path in pixel i or a zero-contribution null sample $\emptyset$ [Wyman et al. 2023], is then chosen proportionally to the w_i.

Third, in temporal reuse, each current pixel i conceptually reuses from all prior frame pixels j (and pixel i's initial sampling). Most pixels do not contribute, as the temporal random replay shift to i fails. Substituting $\hat{q}$ (Equation 48) to the generalized balance heuristic function (Equation 17) with confidence weights M_i yields

$$m_i(\hat{x}) = \frac{M_i h_i(T^{-1}(\bar{x}))\hat{p}_{\leftarrow i}(x)}{\sum_j M_j h_j(T^{-1}(\bar{x}))\hat{p}_{\leftarrow j}(\hat{x}) + M_{\text{is}} h_i(\hat{x})\hat{p}(\hat{x})}, \tag{50}$$

where T^{-1} is the random replay shift to the previous frame, shared with all pixels, i runs over all previous-frame pixels and the current pixel's initial sampling result, and j runs over all previous-frame pixels. M_{is} is the confidence weight given to the initial sample. We immediately see that the pixel a caustic path $\hat{x}$ shifts to with random replay is unique, and hence the denominator contains only two active terms, assuming the box filter, and simple sum over multiple pixels for wider filters (the shift mapping, its Jacobian, and $\hat{p}$ do not depend on the pixel).

In the following, we assume a 1-pixel box filter to avoid notation clutter without loss of generality. Given a sample $\hat{X}_i$ from previous frame's pixel i, and its successful random replay shift $\hat{Y}_i = T(\hat{X}_i)$ in the *current pixel*, a simple substitution yields

$$m_i(\hat{Y}_i) = \frac{M_i \hat{p}_{\text{prev}}(\hat{X}_i)\left|\frac{\partial T}{\partial X_i}\right|^{-1}}{M_i \hat{p}_{\text{prev}}(\hat{X}_i)\left|\frac{\partial T}{\partial X_i}\right|^{-1} + M_{\text{is}}\hat{p}(\hat{Y}_i)}, \tag{51}$$

where $\hat{p}$ is the current frame's target function, $\hat{p}_{\text{prev}}$ is the previous frame's target function, M_i is the confidence weight of the source pixel's reservoir in the previous frame, and M_{is} is the confidence weight of the initial sample. If the shift does not succeed or $\hat{X}_i$ does not hit the current pixel when shifting, the resampling MIS weight $m_i(\hat{Y}_i)$ is zero.

Similarly, the initial sample $\hat{Y}_{\text{is}}$, which is already in current pixel's domain, gets a resampling MIS weight

$$m_{\text{is}}(\hat{Y}_{\text{is}}) = \frac{M_{\text{is}}\hat{p}(\hat{Y}_{\text{is}})}{M_i \hat{p}_{\text{prev}}(\hat{X}_{\text{is}})\left|\frac{\partial T^{-1}}{\partial Y_{\text{is}}}\right| + M_{\text{is}}\hat{p}(\hat{Y}_{\text{is}})}, \tag{52}$$

where $\hat{X}_{\text{is}} = T^{-1}(\hat{Y}_{\text{is}})$ is the initial sample shifted to the previous frame with random replay, and i is the pixel it hits. If the shift fails or hits no pixel, the term containing M_i is zero.

This mathematical construction allows temporal reuse of caustics from pixels as defined by the random replay shift. It also allows stratification and unbiased temporal accumulation of caustics in the pixels they land, extending the limits of GRIS theory by seemingly allowing the samples themselves to define the pixels included in the MIS weights. Yet this does not eliminate the requirement to keep confidence weights independent of realized samples. We still *cannot* increment confidence weights by counting the number of caustics *landing* on each pixel. We explain our method for updating the caustic reservoirs' confidence weights in Section 5.1.

B SOLID-ANGLE JACOBIANS

Here we give solid-angle Jacobians for our bidirectional hybrid shift. For further discussion on the random replay and reconnection Jacobians, we refer to Lin et al. [2022].

The solid-angle Jacobian for random replay is the ratio of solid-angle sampling PDFs between original and shifted paths. We denote the normalized direction from x_i to x_j on the original path as $\omega_{i\to j}$, and similarly the normalized direction from x'_i to x'_j on the shifted path as $\omega'_{i\to j}$. For random replay on the camera subpath, we have

$$\left|\frac{\partial \omega'_{i-1\to i}}{\partial \omega_{i-1\to i}}\right| = \frac{\overrightarrow{p}^{\sigma}_i}{\overrightarrow{p}'^{\sigma}_i} \tag{53}$$

and for random replay on the light subpath, we have

$$\left|\frac{\partial \omega'_{i+1\to i}}{\partial \omega_{i+1\to i}}\right| = \frac{\overleftarrow{p}^{\sigma}_i}{\overleftarrow{p}'^{\sigma}_i} \tag{54}$$

where $\overrightarrow{p}^{\sigma}_i$ and $\overleftarrow{p}^{\sigma}_i$ are the forward and reverse solid-angle PDFs of sampling vertex x_i, respectively ($\overrightarrow{p}^{\sigma}_i$ is simply $\overrightarrow{p}_i$ expressed in the solid-angle measure).

The Jacobian for forward reconnection is the ratio of forward geometry terms along the reconnection edge:

$$\left|\frac{\partial \omega'_{i-1\to i}}{\partial \omega_{i-1\to i}}\right| = \frac{\left|n'_i \cdot \omega'_{i-1\to i}\right|}{\left|n_i \cdot \omega_{i-1\to i}\right|} \frac{\|x_{i-1} - x_i\|^2}{\|x'_{i-1} - x'_i\|^2}, \tag{55}$$

where n_i is the geometry normal at x_i. Note that we must also apply the reconnection Jacobian to the edge between z_{t-1} and y_{s-1} for $s \geq 1, t \geq 2$ techniques (NEE and vertex connection).

For reconnection along the light subpath (only used for $t \leq 1$ techniques), reverse reconnection Jacobian is the ratio of reverse geometry terms along the reconnection edge:

$$\left|\frac{\partial \omega'_{i+1\to i}}{\partial \omega_{i+1\to i}}\right| = \frac{\left|n'_i \cdot \omega'_{i+1\to i}\right|}{\left|n_i \cdot \omega_{i+1\to i}\right|} \frac{\|x_{i-1} - x_i\|^2}{\|x'_{i-1} - x'_i\|^2} \tag{56}$$

where $i = 2$ for noncaustic paths, where x'_2 is reconnected to x'_1. For caustic paths where a light subpath is reconnected directly to the camera, the Jacobian is the same equation but with $i = 1$.