Lab Report: Conditional Resampled Importance Sampling and ReSTIR

Jan Spindler [3338877]

Institute of Computer Science II - Computer Graphics
University of Bonn

- Conditional Resampled Importance Sampling and ReSTIR by Kettunen et al. (2023)
- Further advancements to the recently very successful application of RIS in rendering
- Introduced subpath resampling
- Presents Suffix ReSTIR
- Reduces correlations

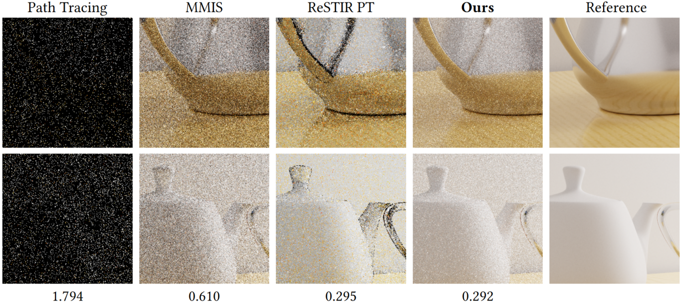


Figure: (Kettunen et al., 2023)

- A very successful application of RIS
- Thousands of emitters
- Introduction of ReSTIR

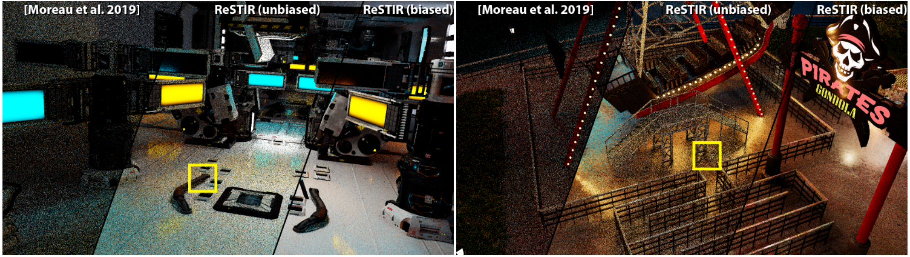


Figure: ReSTIR DI (Bitterli et al., 2020)

- Generalized RIS
- Full path reuse
- High quality at low sample counts
- Problem with high correlations -> Postponing reuse can help

Figure: ReSTIR PT (Lin et al., 2022)

Subpath Resampling

Idea:

- What if we do not want to resample full paths but only a subpath?
- E.g. generate a prefix and then reconnect to a resampled suffix
- Estimate a joint integral over prefixes x_1 and suffixes x_2

$$I = \int \int f(x_1, x_2) dx_2 dx_1 = \mathbb{E}\left[\frac{f(X_1, X_2)}{p(X_1, X_2)}\right] = \mathbb{E}\left[\mathbb{E}\left[\frac{f(X_1, X_2)}{p(X_2|X_1)}\middle| X_1\right]\frac{1}{p(X_1)}\right]$$

New queations:

- How to exploit ReSTIR for subpath reuse
- How to perform RIS with conditional dependence?

Resampled Importance Sampling

Importance Sampling:

- Monte Carlo integration generates samples X using a distribution p to evaluate the integrand f
- Differences in $f(X)$ and $p(X)$ introduce variance in the estimator (noisy images)
- Generating more samples to filter the noise is costly
- Better sampling distributions are desirable (ideally proportional)
- Good distributions might be hard or impossible to sample from using conventional methods

$$\langle I \rangle_{\mathsf{MC}} = \frac{f(X)}{p(X)}$$

Idea:

- If a target distribution $\hat{p}$ can be evaluated but not sampled from
- Sample candidate using a simpler distribution p
- Perform weighted resampling to approximate $\hat{p}$

- Source distribution p
- Target distribution $\hat{p}$
- Candidate samples $X_1, ..., X_M$
- Output sample Y
- Resampling weights w_i
- Resampling probability $P(Y = X_i)$

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

$$P(Y = X_i) = \frac{w_i}{\sum_j^M w_j}$$

- $\langle I \rangle_{\mathsf{RIS}}$ is an unbiased estimator
- The distribution of Y only approximates $\hat{p}$
- The sum over w_i compensates for the difference
- The difference decreases in M

$$\langle I \rangle_{\mathsf{RIS}} = \frac{f(Y)}{\hat{p}(Y)} \sum_j^M w_j$$

- What is the true distribution of Y?
- Don't we need the true distribution of Y?
- E.g. for multi stage RIS

Unbiased Contribution Weights

- We don't!
- Instead we can use an unbiased estimate of the reciprocal of the sampling distribution of X
- We call this the unbiased contribution weight (UCW) of the random variable X
- $\langle I \rangle_{\text{UCW}}$ is an unbiased estimator of I

$$W_X = \mathbb{E}[1/p(X)]$$

$$I = \mathbb{E}\left[\frac{f(X)}{p(X)}\right] = \mathbb{E}[f(X)\mathbb{E}[1/p(X)]] = \mathbb{E}[f(X)W_X]$$

$$\langle I \rangle_{\text{UCW}} = f(X)W_X$$

- Note that we have used UCWs in the previous slide:

$$\langle I \rangle_{\text{RIS}} = f(Y)W_Y \text{ with } W_Y = \frac{1}{\hat{p}(Y)}\sum_j^M w_j$$

Resampling From Multiple Distributions

- Importance sampling can be improved by using multiple source distributions (multiple importance sampling)
- Talbot (2005) uses MIS to improve RIS
- Given candidate samples $X_1, ..., X_M$ with source distributions $p_1, ..., p_M$

$$w_i = m_i(X_i)\frac{\hat{p}(X_i)}{p_i(X_i)} = m_i(X_i)\hat{p}(X_i)W_{X_i}$$

- The MIS weights m_i must be a partition of unity
- E.g. the balance heuristic:

$$m_i(x) = \frac{p_i(x)}{\sum_j^M p_j(x)}$$

Problem:

- But what if p_i is unknown?
- E.g. because it is the result of a RIS pass
- Then we can't evaluate the balance heuristic

Idea:

- p_i is supposed to approximate a known target distribution $\hat{p}_i$
- Use the target distributions $\hat{p}_i$ as proxies
- Modified balance heuristic:

$$m_i(x) = \frac{\hat{p}_i(x)}{\sum_j^M \hat{p}_j(x)}$$

Resampling From Different Domains

Motivation (ReSTIR PT):

- We want to resample full paths
- We want to reuse samples from neighbor pixels (exploit spatial and temporal correlations)

Problem:

- The neighbor path carries no contribution to the target pixel
- It does not hit that pixel in space

Idea:

- Transform the neighbor path
- Then resample from the transformed candidate samples

Resampling From Different Domains

More formally we have:

- Target domain Ω
- Input domains Ω_i
- Candidate samples $X_i \in \Omega_i$
- UCWs W_{X_i}
- Shift mappings $T_i : \Omega_i \to \Omega$ (bijective)

$$w_i = \begin{cases} m_i(T_i(X_i))\hat{p}(T_i(X_i))W_{X_i} \cdot \left|\frac{\partial T_i}{\partial X_i}\right|, & \text{if } X_i \in \mathcal{D}(T_i), \\ 0, & \text{otherwise} \end{cases}$$

Questions:

- Are neighbor samples sufficient to produce an unbiased estimate?
- How to evaluate $\hat{p}_j(T_i(X_i))$? How to calculate MIS weights with multiple domains?
- What are good shift mappings for path reuse?

Canonical Samples

Unbiased integration:

- For $\langle I \rangle$ to be unbiased, we require supp$(t) \subseteq$ supp(p)
- Or equivalently $\forall x : f(x) > 0 \Rightarrow p(x) > 0$
- Otherwise parts of the integration domain are "ignored"

$$\langle I \rangle = \frac{f(X)}{p(X)}$$

Are neighbor samples sufficient to produce an unbiased estimate?

- Images of the shift mappings must cover the entire target domain: $\Omega = \bigvee_i^M \mathcal{I}(T_i)$
- Problem: Difficult or impossible to guarantee
- Solution: Add samples X_i from the target domain with supp$(X_i) = \Omega$
- Those samples are called canonical samples
- E.g. use path tracing to generate a canonical path for the target pixel

Problem:

- Naively using the balance heuristic would look like this:
- But this can usually not be evaluated because the distributions are defined on their respective domain

$$m_i(Y_i) = \frac{\hat{p}_i(Y_i)}{\sum_j^M \hat{p}_j(Y_i)} \text{ with } Y_i = T_i(X_i)$$

Solution:

- Shift mappings are bijective
- Shift Y_i back into the domain Ω_j of the distribution $\hat{p}_j$ using the inverse shift mapping

$$\hat{p}_{\leftarrow i}(y) = \begin{cases} \hat{p}_i(T_i^{-1}(y)) \left| \partial T_i^{-1} / \partial Y_i \right|, & \text{if } y \in T_i(\mathsf{supp}(X_i)) \\ 0, & \text{otherwise} \end{cases}$$

$$m_i(Y_i) = \frac{\hat{p}_{\leftarrow i}(Y_i)}{\sum_j^M \hat{p}_{\leftarrow j}(Y_i)}$$

Problem with the balance heuristic:

- Evaluates M distributions for M samples
- Cost is $O(M^2)$

Solution (Bitterli et al., 2021):

- Assumption: The canonical target distribution is sufficiently good
- Instead of comparing all distributions $\hat{p}_j$ for each sample X_i
- Only compare the target distribution of the sample $\hat{p}_i$ with the canonical target distribution $\hat{p}$
- Cost is $O(M|R|)$. R is the set of canonical samples.

$$m_i(y) = \begin{cases} \frac{1}{M-|R|} \sum_{j \notin R} \frac{\hat{p}(y)}{|R|\hat{p}+(M-|R|)\hat{p}_{\leftarrow j}(y)}, & \text{if} i \in R \\ \frac{\hat{p}_{\leftarrow i}(y)}{|R|\hat{p}+(M-|R|)\hat{p}_{\leftarrow i}(y)}, & \text{if} i \notin R \end{cases}$$

Reconstruction Shift:

- Given a source path $X_i = (x_0, ..., x_i, x_{i+1}, ...) \in \Omega_i$
- We can construct a path $Y_i = (y_0, ..., y_i, x_{i+1}, ...) \in \Omega$ by reconnecting at the i-th vertex
- The segment (y_0, y_1) is known and crosses the target pixel
- Problem: Reconnecting on very specular materials causes large differences in path contribution

Hybrid Shift: Postpone reconnection using random replay until reconnection is feasible

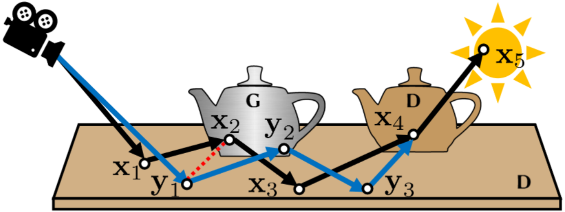


Figure: Hybrid Shift (Lin et al., 2022)

ReSTIR

Remaining problems with RIS

- We need to store all M candidate samples and resampling weights
- After RIS is complete, all gained information (about a good distribution) is omitted

Spatiotemporal reservoir resampling (ReSTIR)

- Use weighted reservoir sampling which only stores a single sample
- Resample from temporal and spatial neighbor reservoirs
- Information is kept and improved over time and space

Reservoir Resampling

- Stream each candidate sample one-by-one
- Store the current candidate with $P(Y = X_i) = w_i / w_{\text{sum}}$
- Store the currently selected sample Y and the sum of all resampling weights so far w_{sum}
- Each pixel only stores it's reservoir

Algorithm 1: Reservoir Resampling

```
class Reservoir:
    Y ← ∅
    w_sum ← 0
    function update(X_i, w_i):
        w_sum ← w_sum + w_i
        if rand() < (w_i / w_sum) :
            Y ← X_i
function Resample(M):
    Reservoir r
    for i ← 1 to M :
        generate X_i
        calculate w_i
        r.update(X_i, w_i)
    return r
```

Temporal and Spatial Reuse

Temporal Reuse:

1. Find the temporal neighbor of the target pixel
2. Shift into target domain
3. Calculate resampling weights
4. Stream into reservoir

Spatial Reuse:

1. Select multiple screen space neighbors
2. Shift into target domain
3. Calculate resampling weights
4. Stream into reservoir

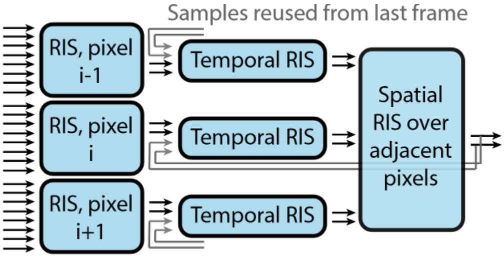


Figure: ReSTIR (Bitterli et al., 2020)

Confidence Weights

Note:

- What if 2 input reservoirs R_1 and R_2 have seen different amounts of input samples M_1 and M_2
- The higher the input sample count the more information is contained within the reservoir

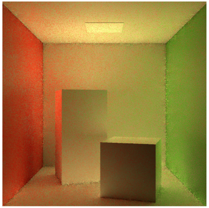
Idea:

- Intoduce confidence weights c_i for each reservoir
- Add confidence weights to MIS weight calculation
- For each input sample increment the confidence weight
- When resampling the output sample of another reservoir add the confidence weights

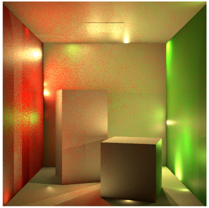
$$m_i(y) = \frac{c_i \hat{p}_{\leftarrow i}(y)}{\sum_j^M c_j \hat{p}_{\leftarrow j}(y)}$$

Problem:

- Confidence weights overestimate the effective input sample count
- Due to correlations
- This hinders convergence
- This causes artifacts
- Confidence weights need to be capped (M-capping)



(a) *M-cap:* $M_c = 20$



(b) *No M cap*

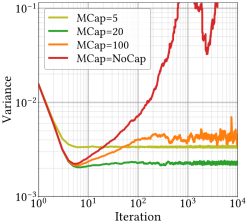


Figure: (Lin et al., 2022)

What we have seen

Revisited:

- Basic resampled importance sampling
- Unbiased contribution weights
- Resampling from different distributions
- Resampling from different domains
- ReSTIR

New:

- Conditional unbiased contribution weights
- Joint unbiase contribution weights
- Conditional resampled importance sampling
- Suffix ReSTIR (rendering algorithm)

Conditional Unbiased Contribution Weights

- We want to integrate a function $f(x)$
- Using samples X drawn from a conditional distribution $p_{X|Y}$
- E.g. when integrating the suffix path contribution given a prefix

$$\mathbb{E}\left[\frac{f(X)}{p_{X|Y}(X|Y)}\middle|Y\right] = \int_{\mathsf{supp}(X|Y)} f(x)dx$$

- If $p_{X|Y}(X|Y)$ can't be evaluated
- But an unbiased estimate of the reciprocal $W_{X|Y}$ is available
- Unbiased estimation is possible

$$\mathbb{E}\left[f(X)W_{X|Y}|Y\right] = \int_{\mathsf{supp}(X|Y)} f(x)dx$$

Joint Unbiased Contribution Weights

Estimating a joint integral over a function $f(x, y)$ can be done using:

- The marginal distribution p_Y
- The conditional distribution $p_{X|Y}$

$$\langle I \rangle = \frac{f(X,Y)}{p_{X,Y}(X,Y)} = \frac{f(X,Y)}{p_Y(Y)p_{X|Y}(X|Y)}$$

Kettunen et al. (2023) show that the same can be achieved using UCWs W_Y and $W_{X|Y}$

$$\langle I \rangle = f(X,Y)W_{X|Y}W_Y = f(X,Y)W_{X,Y}$$

when X and $W_{X|Y}$ are conditionally independent of W_Y, given Y.

When performing RIS in a probability space conditioned on Z we can simply reuse the terms seen before (Kettunen et al., 2023):

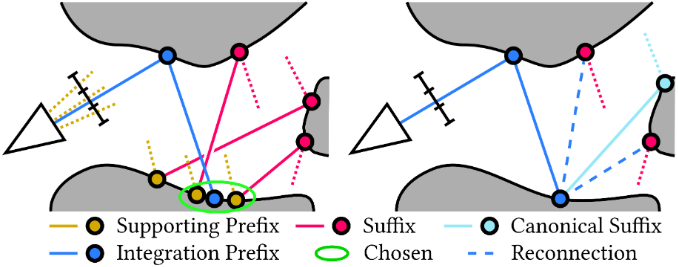
$$w_i = m_i(Y_i|Z)\hat{p}(Y_i|Z)W_{X_i|Z}\left|\frac{\partial T_i}{\partial X_i|Z}\right|$$

$$m_i(y|Z) = \frac{c_i \hat{p}_{\leftarrow i}(y|Z)}{\sum_j^M c_j \hat{p}_j(y|Z)}$$

Effectively, ReSTIR with suffixes is the same as with full paths. When using them to integrate a joint function (prefix + suffix) we must ensure the condition formulated above (Kettunen et al., 2023).

Suffix ReSTIR (Kettunen et al., 2023):

- Uses conditional resampled importance sampling
- Improve a distribution of suffixes within the scene
- Reconnect those suffixes to new prefixes
- Each pixel has a prefix and a suffix reservoir
- Final gather with multiple prefixes



- To generate suffixes we require prefixes
- Good prefix distribution helps develop good suffix distribution
- Changing prefixes changes the suffix's support and target function
- Compromise: Use canonical prefixes and temporal prefix reuse (ReSTIR PT)
- Target function: Prefix throughput $f_p(x_p)$

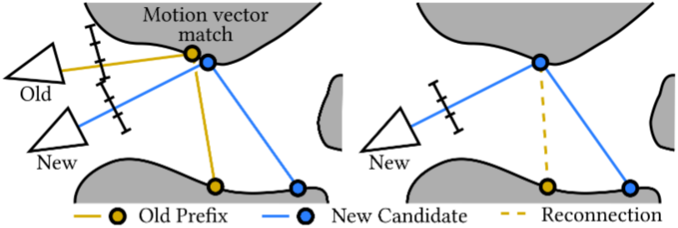


Figure: Temporal prefix reuse (Kettunen et al., 2023)

- Use conditional RIS to improve suffix distribution
- Use temporal and spatial reuse
- Target function: Throughput of reconnection segment $f_{ps}(x_p, x_s)$ and suffix $f_s(x_s)$

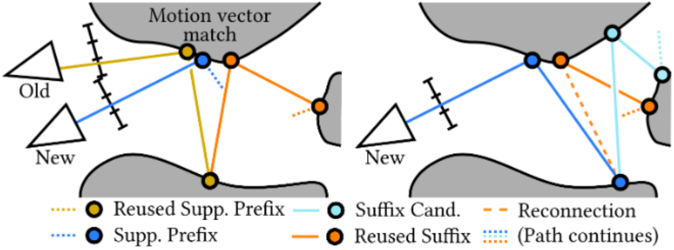


Figure: Temporal suffix reuse (Kettunen et al., 2023)

Suffix ReSTIR - Final Gather

Final Gather:

- Trace multiple canonical prefixes
- Reconnect each prefix with multiple suffixes from the distribution
- Add canonical suffixes to cover support of f

Problems:

- Which suffixes to select?
- Generating canonical suffixes is costly compared to reusing

Which suffixes to select?

- Search KNN prefixes in world space (last vertex position)
- Reconnect their suffixes to the current prefix
- Each stored prefix has a unique radius in which it responds to spatial queries
- Kettunen et al. (2023) select the radius proportional to the euclidian prefix length

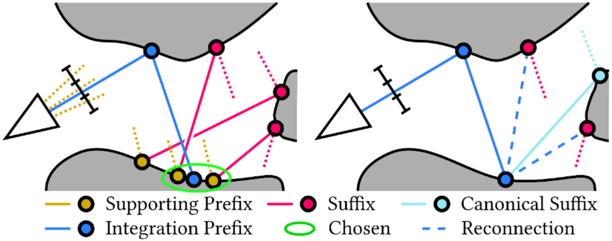


Figure: Borrow suffixes from KNN prefixes (Kettunen et al., 2023)

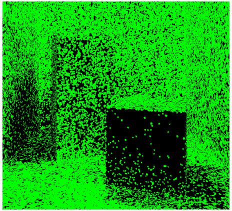


Figure: Prefix distribution in Cornell Box

Generating canonical suffixes is costly compared to reusing:

- Usually prefixes are smaller than suffixes
- Generating suffixes is more expensive than generating prefixes
- We need canonical suffixes to cover the integration domains contitioned on their respective prefix
- But we need only a single canonical prefix-suffix-pair to cover the support of the joint integrand

Kettunen et al. (2023) propose to:

- Generate a single canonical suffix for some canonical prefix
- Multiply it's contribution by the number of prefixes N
- MIS weights have to respect the non-existing canonical suffixes

$$
\begin{aligned}
\langle I \rangle_{\mathsf{FG}} =& m_1(X^s_{11} | X^p_1) f(X^p_1, X^s_{11}) W_{X^p_1, W^s_{11}} \\
&+ \frac{1}{N} \sum_{i=1}^{N} \sum_{j=2}^{M} m_i(Y^s_{ij} | X^p_i) f(X^p_i, Y^s_{ij}) W_{X^p_i, Y^s_{ij}}
\end{aligned}
$$

Renderer Stages:

1. Prefix generation and temporal reuse
2. Prefix spatial reuse (for ReSTIR PT)
3. Suffix generation and temporal reuse
4. Suffix spatial reuse
5. Build prefix search acceleration structure (Evangelou et al., 2021)
6. Final gather

- Each stage is implemented as an OptiX shader (Parker et al., 2010)
- ReSTIR PT only uses stage 1 and 2

Content	**1 Pixel**	**1920x1080 Pixel**
2 Prefix reservoir	384 B	796 MB
PrefixPath	184 B	382 MB
2 Suffix reservoir	304 B	630 MB
SuffixPath	144 B	299 MB
ReSTIR G Buffer	32 B	67 MB
Prefix Search AABB	24 B	50 MB
Total	1072 B	2223 MB

Table: Memory usage per pixel and for 1920 x 1080 pixels.

Evaluation

Testing Environment:

- Windows 11, Cuda 12.5, OptiX 8.0.0
- CPU: AMD Ryzen 7 7700x
- GPU: Nvidia GeForce RTX 3070
- Resolution: 800×800 pixel
- Scene: Cornell Box

Performance

Path tracing: 45 ms

ReSTIR PT:

- Base runtime without temporal or spatial reuse: 80 ms
- Temporal reuse: 45 ms
- Spatial reuse: Approximately 20 ms per spatial neighbor

Suffix ReSTIR:

- Base runtime without prefix or suffix reuse: 170 ms (stage 1 [45 ms], stage 3 [72 ms], stage 5 [7.5 ms], stage 6 [50 ms])
- Temporal prefix reuse: 22 ms
- Temporal suffix reuse: 63 ms
- Spatial suffix reuse: Approximately 95 ms per spatial neighbor

- Sublinear growth in number of prefixes
- Search radius has a high effect (linear?) on per-prefix-cost
- Sublinear growth in number of suffixes
- Search radius has a constant effect on per-suffix-cost
- Cost of spatial query $>$ cost of reconnection

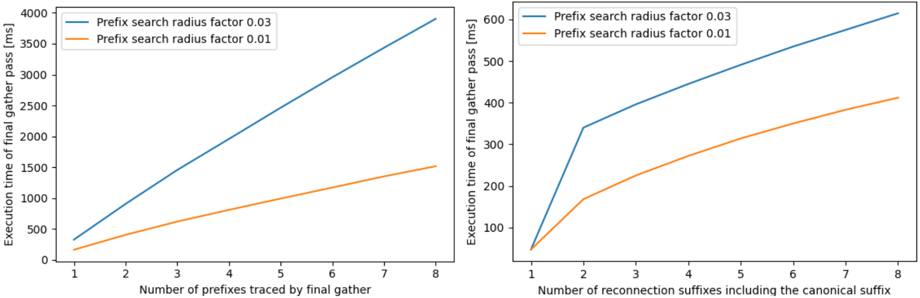


Figure: Final gather runtime depending on amounts of prefixes and suffixes

Performance

Cost of memory access:

- Both ReSTIR PT and Suffix ReSTIR require a lot of memory accesses
- Original implementations are faster and use significantly smaller reservoirs (184 Bytes vs. 88 Bytes)

Information compression:

- PackedInteraction (40 Bytes) vs. Interaction (88 Bytes): Successful
- u8 vs. f32: Unsuccessful (Lin et al., 2022)?
- f16 vs. f32: Unsuccessful

Cost of spatial queries:

- Prime driver of cost in our implementation
- Accounts for 30% of cost in the original implementation (Kettunen et al., 2023)

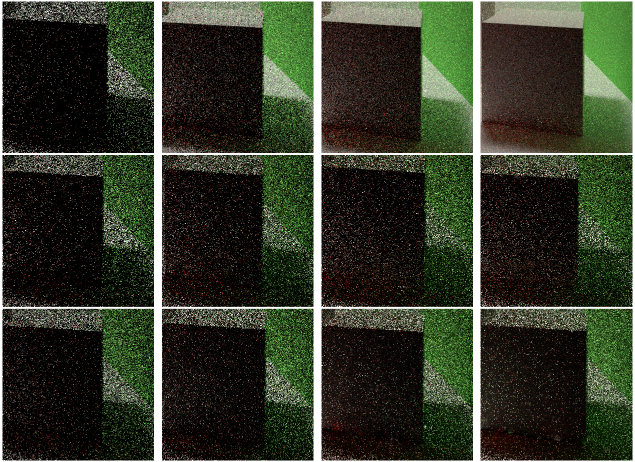
Quality

Row:

1. Path tracing with 1, 8, 32, and 128 spp
2. Final gather with 1 prefix and 2, 8, 32, and 128 suffixes
3. Final gather with 1, 8, 32, 128 prefixes and 1 reconnection suffix

We see that:

- Postponing resampling introduces noise
- Noise by canonical suffix remains visible for low suffix counts



Spatial reuse in ReSTIR PT:

- Enabling spatial reuse causes a significant darkening
- Increasing the spatial neighbor count increases the brightness
- Likely a flawed implementation

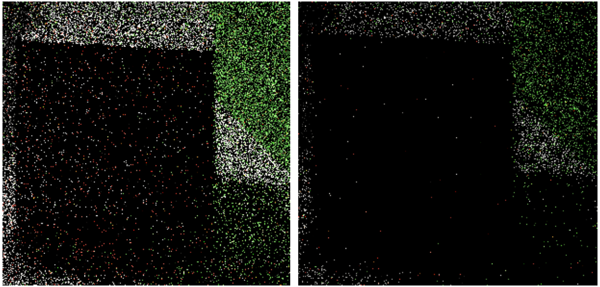


Figure: (Left) 1 spp path tracing. (Right) 1 spp ReSTIR PT with only spatial reuse using one neighbor and one spatial reuse round.

Temporal and spatial reuse in ReSTIR PT:

- Enabling temporal and spatial reuse with over 4 neighbors
- Bright spots develop and spread spatially over time
- 3 spatial neighbors maintain the artifact
- With less than 3 spatial neighbors the artifacts fade away

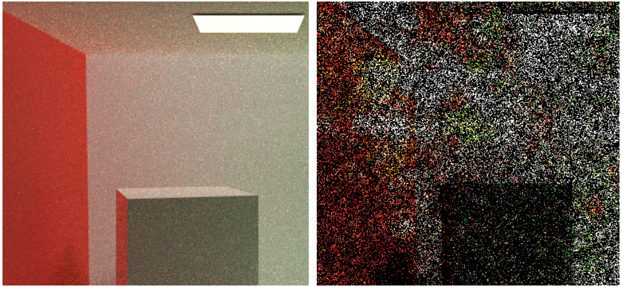


Figure: (Left) 512 spp path tracing. (Right) 1 spp ReSTIR PT with temporal and spatial reuse enabled. ReSTIR PT performs a single round of spatial reuse with 4 neighbors. Confidence weights are capped by 20.

Quality

Temporal and spatial reuse in Suffix ReSTIR:

- Using both introduces single very bright spots
- Accumulating shows that the estimate is biased

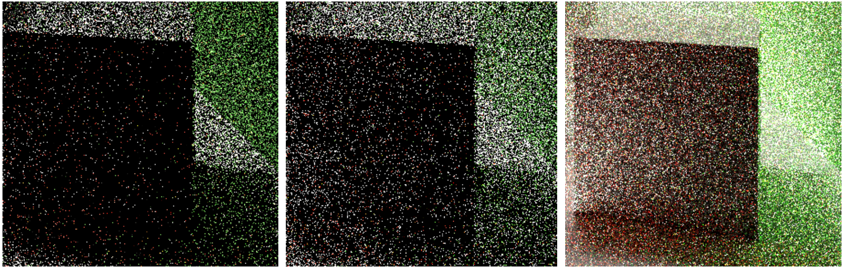


Figure: (Left) 1 spp path tracing. (Middle) 1 spp Suffix ReSTIR with prefix and suffix resampling. (Right) 32 spp Suffix ReSTIR with prefix and suffix resampling.

Conclusion

We have seen:

- How to generalize RIS to conditional probability spaces
- How to use CRIS to select a suffix for joint prefix-suffix integration
- Suffix ReSTIR (final-gather-based)

Out evaluation shows:

- That memory access is a prime driver of runtime in ReSTIR
- That spatial queries are very costly (Evangelou et al., 2021)
- Unfortunately, that our implementation of ReSTIR PT and Suffix ReSTIR is flawed

Future work:

- Reduce memory usage of ReSTIR
- Try other methods for spatial queries (e.g. KD trees)
- Better heuristics for prefix search radius
- Discover other applications of CRIS

References I

[1] B. Bitterli et al. "Correlations and Reuse for Fast and Accurate Physically Based Light Transport". AAI28966216. PhD thesis. USA, 2021.

[2] B. Bitterli et al. "Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting". In: *ACM Trans. Graph.* 39.4 (2020). URL: https://doi.org/10.1145/3386569.3392481.

[3] I. Evangelou et al. "Fast Radius Search Exploiting Ray Tracing Frameworks". In: *Journal of Computer Graphics Techniques (JCGT)* 10.1 (2021), pp. 25–48. URL: http://jcgt.org/published/0010/01/02/.

[4] M. Kettunen et al. "Conditional Resampled Importance Sampling and ReSTIR". In: *SIGGRAPH Asia (Conference Track)*. 2023.

[5] D. Lin et al. "Generalized Resampled Importance Sampling: Foundations of ReSTIR". In: *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2022)* 41.4 (July 2022). (*Joint First Authors), 75:1–75:23. URL: https://doi.org/10.1145/3528223.3530158.

[6] S. G. Parker et al. "OptiX: A General Purpose Ray Tracing Engine". In: *ACM Transactions on Graphics* (2010).

[7] J. Talbot. *Importance Resampling for Global Illumination*. Master's thesis. Available at https://scholarsarchive.byu.edu/etd/663/. 2005.