Gradient Estimation for Real-Time Adaptive Temporal Filtering

CHRISTOPH SCHIED, CHRISTOPH PETERS, and CARSTEN DACHSBACHER, Karlsruhe Institute of Technology, Germany

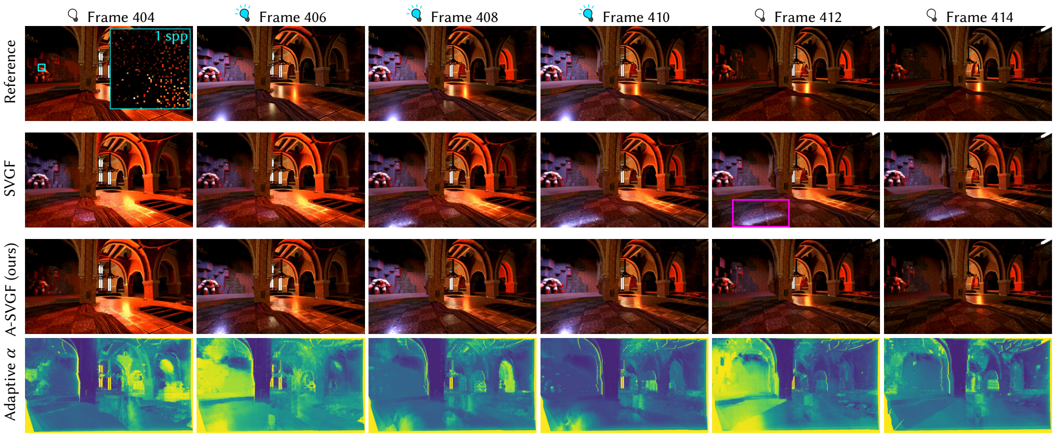


Fig. 1. Results of our novel spatio-temporal reconstruction filter (A-SVGF) for path tracing at one sample per pixel (cyan inset in frame 404) with a resolution of 1280×720. The animation includes a moving camera and a flickering, blue area light. Previous work (SVGF) [Schied et al. 2017] introduces temporal blur such that lighting is still present when the light source is off and glossy highlights leave a trail (magenta box in frame 412). Our temporal filter estimates and reconstructs sparse temporal gradients and uses them to adapt the temporal accumulation factor α per pixel. For example, the regions lit by the flickering blue light have a large α in frames 406 and 412 where the light has been turned on or off. Glossy highlights also receive a large α due to the camera movement. Overall, stale history information is rejected reliably.

With the push towards physically based rendering, stochastic sampling of shading, e.g. using path tracing, is becoming increasingly important in real-time rendering. To achieve high performance, only low sample counts are viable, which necessitates the use of sophisticated reconstruction filters. Recent research on such filters has shown dramatic improvements in both quality and performance. They exploit the coherence of consecutive frames by reusing temporal information to achieve stable, denoised results. However, existing temporal filters often create objectionable artifacts such as ghosting and lag. We propose a novel temporal filter which analyzes the signal over time to derive adaptive temporal accumulation factors per pixel. It repurposes a subset of the shading budget to sparsely sample and reconstruct the temporal gradient. This allows us to reliably detect sudden changes of the sampled signal and to drop stale history information. We create gradient samples through forward-projection of surface samples from the previous frame into the current frame and by reevaluating the shading samples using the same random sequence. We apply our filter to improve real-time path tracers. Compared to previous work, we show a significant reduction of lag and ghosting as well as

Authors' address: Christoph Schied, schied@kit.edu; Christoph Peters, christoph.peters@kit.edu; Carsten Dachsbacher, dachsbacher@kit.edu, Karlsruhe Institute of Technology, Am Fasanengarten 5, 76131, Karlsruhe, Germany.

 The definitive Version of Record was published in *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, https://doi.org/10.1145/3233301.

improved temporal stability. Our temporal filter runs in 2 ms at 1080p on modern graphics hardware and can be integrated into deferred renderers.

CCS Concepts: • **Computing methodologies → Ray tracing**; **Image processing**;

Additional Key Words and Phrases: temporal filtering, global illumination, reconstruction, denoising, real-time rendering

ACM Reference Format:
Christoph Schied, Christoph Peters, and Carsten Dachsbacher. 2018. Gradient Estimation for Real-Time Adaptive Temporal Filtering. *Proc. ACM Comput. Graph. Interact. Tech.* 1, 2, Article 24 (August 2018), 16 pages.
https://doi.org/10.1145/3233301

1 INTRODUCTION

Stochastic sampling has proven to be the most practical approach to the many integration problems that arise in rendering. However, it is only used to a limited extent in real-time rendering as the requirement of temporally stable results usually necessitates high sample counts that are prohibitively expensive. Recent research on real-time reconstruction filters [Chaitanya et al. 2017; Mara et al. 2017; Schied et al. 2017] has made a significant step towards making real-time path tracing viable. Their common assumption is a high frame rate of 30 to 60 Hz, but only a single stochastic shading sample per pixel. These works strongly suggest that under these conditions, high image quality and stable results can only be obtained by increasing the effective sample count through reuse of information from previous frames.

Most temporal filters used in real time rely on exponential moving averages. The previous frame, including temporal accumulation, is reprojected and blended together with the current frame using a temporal accumulation factor α. In state of the art reconstruction filters for stochastic sampling [Mara et al. 2017; Schied et al. 2017], this temporal accumulation factor is constant. Thus, shading samples from the previous frame are reused even when the shading of the scene has changed dramatically. This inevitably leads to ghosting and temporal lag (see Figure 1). While approaches exist to reject stale history [Iglesias-Guitian et al. 2016; Karis 2014; Patney et al. 2016; Xiao et al. 2018], they are designed for non-stochastic shading.

In this paper, we address temporal overblurring. We show how to reliably change the temporal accumulation factor α *per frame and per pixel* to reliably drop stale history information. The fixed accumulation factor in previous work specifies a fixed tradeoff between temporal lag and stability. By setting the parameter adaptively, we achieve fast response times in case of sudden changes while using more aggressive temporal filtering in regions where the signal is constant.

To control temporal accumulation adaptively, we estimate a temporal gradient measuring changes of the signal over time. This temporal gradient needs to incorporate global information such as shadows and indirect illumination effects. In general, it depends on the entire shading function of the previous frame and the current frame. On the other hand, it needs to be robust to noise because we must not drop history information solely due to changes in random numbers.

Our solution is to repurpose a part of the shading budget of the current frame for temporal gradients. Per 3×3 pixels, one surface sample from the previous frame is selected and projected to the current frame using forward projection. It replaces the closest surface sample that would be evaluated in the current frame and is shaded in the same way (Section 3.3). Since this reevaluated shading sample refers to the exact same surface location as the sample from the previous frame, it enables the computation of a temporal gradient that is entirely free of spatial offsets (Section 3.1). Additionally, we make it independent of random changes by reusing the same sequence of random numbers (Section 3.2). The sparse and noisy temporal gradients enter a joint-bilateral filter to obtain an estimate per pixel (Section 3.4). Finally, the temporal gradient, normalized by absolute brightness, is used to control the temporal accumulation factor (Section 3.5).

We apply our temporal filter to improve various existing real-time light transport solutions (Section 4). We reconstruct soft shadows and global illumination using adaptive spatiotemporal variance-guided filtering (A-SVGF) which augments SVGF [Schied et al. 2017] with our adaptive accumulation factor. Additionally, we improve the temporal stability of reconstruction with a recurrent autoencoder [Chaitanya et al. 2017]. Our design decisions are justified through an extensive evaluation (Section 5). Although our approach is heuristic in nature, it successfully diminishes temporal blur while maintaining high effective sample counts where possible.

2 RELATED WORK

When rendering animations, there is usually a large amount of coherence between frames, which can be exploited to amortize cost. In offline rendering, the whole animation is usually known in advance, allowing entire image sequences to be denoised simultaneously [Delbracio et al. 2014; Manzi et al. 2016; Meyer and Anderson 2006; Zimmer et al. 2015].

Manzi et al. [2016] compute temporal gradients through reuse of random sequences and precise reprojections in screen-space. Their goal is to reconstruct images from unbiased gradients by means of a Poisson reconstruction. This requires access to all frames simultaneously and sample counts currently out of reach for real-time rendering.

Zimmer et al. [2015] decompose their image in path-space which allows them to compute precise motion vectors for specific phenomena and enables reuse across frames without introducing objectionable artifacts. However, they require access to the full animation.

Günter and Grosch [2015] use gradient information directly to detect changes and to update older images. This necessitates sample counts that are impractical for real-time rendering and simultaneous access to old and new scene data. In contrast, we only require the current scene state and obtain gradients without evaluating additional samples. Rousselle et al. [2016] generalize the approach using the theory of control variates.

In real-time rendering, the coherence between frames is even higher, and the need to reduce shading cost more pressing. We focus on reprojection techniques and refer to Scherzer et al. [2012] for an exhaustive overview of various techniques for shading reuse.

Back-projection is currently the predominant technique for shading reuse in real-time rendering. A pixel position x of the current frame i is temporally projected into the previous frame as $\overleftarrow{x}$. Temporal filters [Iglesias-Guitian et al. 2016; Karis 2014; Nehab et al. 2007; Patney et al. 2016; Walter et al. 1999; Xiao et al. 2018; Yang et al. 2009] build an exponential moving average

$$\hat{c}_i(x) = \alpha \cdot c_i(x) + (1 - \alpha) \cdot \hat{c}_{i-1}(\overleftarrow{x}). \tag{1}$$

between the current frame c_i and the previously temporally filtered frame $\hat{c}_{i-1}$. As the back-projection generally introduces sub-pixel offsets, a resampling filter is required. Yang et al. [2009] analyzed and reduced the amount of spatial blur which stems from this resampling. In the context of stochastic sampling of shading, this error is usually negligible compared to the bias introduced by the spatial reconstruction filter. In recent work [Mara et al. 2017; Schied et al. 2017], this effect is used on purpose to implement recursive filtering by back-projecting into spatially filtered images.

There are two main classes of temporal filters where the distinctive feature is the handling of stale history data. *Temporal accumulation* [Nehab et al. 2007; Yang et al. 2009] verifies reprojections using geometric tests to detect changes in visibility. It does not adapt the filter weights based on the signal, making it inherently resilient to noisy shading signals. The drawback of this approach is the lag and ghosting introduced if the shading signal is changing drastically between frames. Such filters cannot be used for visibility anti-aliasing as they anchor the temporal history to surfaces.

Our proposed method falls into the same category but does not suffer from lag since we adaptively control the temporal filter weight.

Temporal anti-aliasing [Karis 2014] (TAA) does not employ geometric tests but uses the evaluated shading samples of the current frame to determine validity of the history. The previous estimate $\hat{c}_{i-1}$ is clipped against an approximate convex hull of all color samples in a spatial neighborhood in the current frame. In presence of noise, either due to high-frequency geometry or noisy shading, the estimated bounds become unreliable which manifests in ghosting and flickering artifacts. Patney et al. [2016] increase reliability by replacing the estimated bounds using a simple statistical model, which however is not sufficiently reliable for stochastic sampling. Moon et al. [2015] fit linear models to adaptively sized neighborhoods. Their approach incorporates previous frames with exponentially decaying weights for the objective function. The resulting predicted colors serve as the denoised estimate. Similarly, Iglesias et al. [2016] use a recursively updated linear model to capture the history of the pixel color, reducing ghosting and flickering artifacts. The accumulation factor for the underlying features uses a luminance-based perceptual heuristic. This addresses flickering artifacts caused by high-frequency geometry but cannot handle stochastic sampling of the shading function.

Besides backward-projection, different parametrizations have been used to support reconstruction with temporal accumulation. Munkberg et al. [2016] perform temporal accumulation and reconstruction in texture-space. Dal Corso et al. [2017] use forward-projection and adaptive sampling of visibility samples. The aforementioned techniques inherently provide stable shading sample positions. Thus, they can be augmented with our gradient sampling and adaptive temporal filter to increase temporal reuse and reduce lag.

Recent *reconstruction filters* [Chaitanya et al. 2017; Mara et al. 2017; Schied et al. 2017] showed dramatic improvements in quality and performance. All of the filters employ temporal reuse as an integral component. Please refer to [Chaitanya et al. 2017; Schied et al. 2017; Zwicker et al. 2015] for an exhaustive review of related work on reconstruction filters. Mara et al. [2017] filter indirect illumination and use a BRDF decomposition which allows them to use different motion vectors for diffuse and glossy terms, thus reducing lag for glossy reflections. However, the reprojection only takes relative camera motions into account and cannot handle global effects of shading. SVGF [Schied et al. 2017] uses temporal accumulation to increase the effective sample count and to build statistics over the samples to control a subsequent spatial filter. It can be used for arbitrary stochastically sampled shading but exhibits temporal lag under fast motion. Chaitanya et al. [2017] employ a neural network with an autoencoder topology. They compare a non-recurrent autoencoder (AE) against a recurrent autoencoder (RAE) and show significant improvements in image quality and temporal stability by using a recurrent network. However, achieving temporally stable results from autoencoders remains challenging due to the lack of a general metric for temporal stability required for training. AE and RAE do not exhibit temporal lag but suffer from temporal instabilities. In Section 5, we demonstrate dramatically reduced temporal lag for SVGF and improved temporal stability for RAE.

3 GRADIENT ESTIMATION

We strive to replace the fixed temporal weight α (Equation 1) by an adaptive per-pixel weight computed from temporal gradients. In the following, we begin with a practical definition of temporal gradients and combine it with stochastic sampling. Then we discuss their efficient computation at sparse locations, followed by a reconstruction step. The resulting dense gradient field is used to compute the per-pixel weight. Section 4 discusses several applications of these general principles.

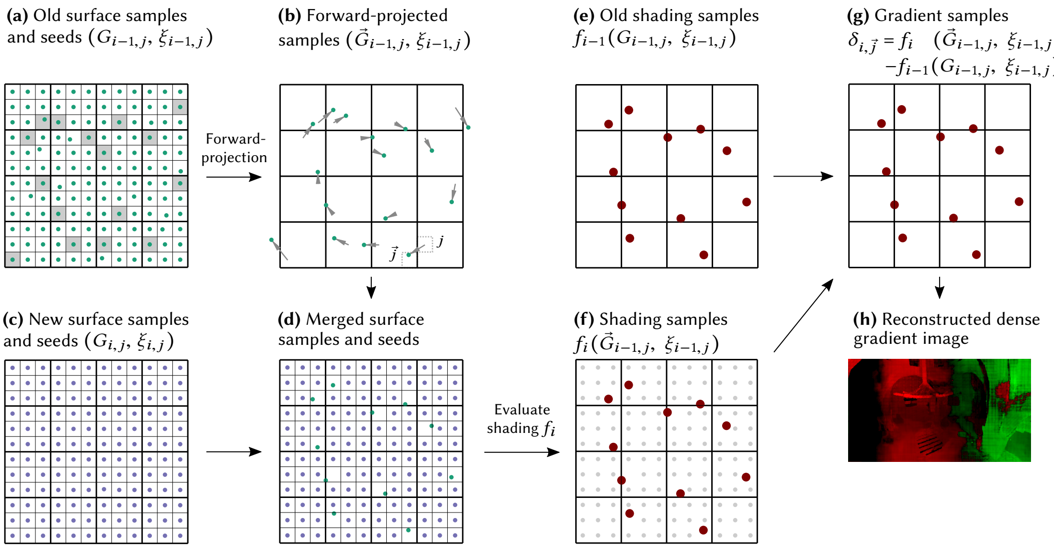


Fig. 2. An overview of our gradient sampling. A sparse subset of surface and shading samples is reprojected (a, b, e). The reprojected surface samples are merged into the new visibility buffer (c, d). Combining the reprojected shading samples with the newly shaded samples yields gradient samples (f, g). There is at most one gradient sample per 3×3 stratum. A reconstruction step turns these scattered and noisy samples into a dense, denoised gradient image (h).

3.1 Definition of the Temporal Gradient

Temporal reuse of information requires temporal reprojection of samples from the previous frame. Our technique performs this reprojection in screen space and thus we need to maintain information about visible surface samples. In a first render pass, a g-buffer [Saito and Takahashi 1990] or a visibility buffer [Burns and Hunt 2013] is generated. For each pixel j in frame i, this yields a surface sample $G_{i,j}$ providing access to surface attributes such as world-space position, normal and diffuse albedo. The g-buffer stores each attribute explicitly whereas the visibility buffer stores information about the triangle intersection. Then a deferred full-screen pass [Saito and Takahashi 1990], typically implemented in a fragment or compute shader, applies the shading function $f_i(G_{i,j})$ to compute a color for pixel j.

Forward projection carries a surface sample $G_{i-1,j}$ from the previous frame $i-1$ to the current frame i. The forward projected surface sample $\vec{G}_{i-1,j}$ provides access to all surface attributes in the current frame for the same point on the surface. This reprojection is challenging with a g-buffer but trivial with a visibility buffer, which is why our implementation uses the latter. Having access to the new world space location through $\vec{G}_{i-1,j}$, the coordinate transforms for frame i yield the corresponding screen space location in the current frame. In particular, we can compute the index of the pixel $\vec{j}$ that covers the surface sample in the current frame (see Figure 2b).

Similarly, we may consider a backprojected surface sample $\overleftarrow{G}_{i,j}$. Both projection operations lead to potential definitions for the temporal gradient of f_i:

$$f_i(G_{i,j}) - f_{i-1}(\overleftarrow{G}_{i,j}) \quad \text{or} \quad f_i(\vec{G}_{i-1,j}) - f_{i-1}(G_{i-1,j}).$$

While the former approach may seem more intuitive, it poses a major problem for the implementation. The backprojected sample $\overleftarrow{G}_{i,j}$ is not known until frame i. However, it has to be used as input to the shading function of frame $i-1$. Thus, all state required to implement the shading function of the previous frame needs to be maintained. With the other formulation, the shading sample $f_{i-1}(G_{i-1,j})$ has already been computed in frame $i-1$ and the forward projected sample enters the shading function for the current frame i. Hence, only the shading samples from the previous frame need to be kept.

Consequently, we define the temporal gradient using forward projection. Since pixel j is being projected forward, it is natural to associate the temporal gradient with the pixel $\vec{j}$ that corresponds to the projected location. Then the temporal gradient is

$$\delta_{i,\vec{j}} := f_i(\vec{G}_{i-1,j}) - f_{i-1}(G_{i-1,j}). \tag{2}$$

In practice, we only work with the luminance of the temporal gradient, thus saving resources.

It may seem tempting to replace the shading sample $f_i(\vec{G}_{i-1,j})$ by a readily available, similar sample from the current frame. However, this would often introduce a sub-pixel offset to the sample location. We have explored this approach but found that this additional spatial offset frequently leads to stark overestimation of the temporal gradient. At least for the applications discussed in Section 4, the perfectly stable sample locations achieved by forward projection are essential.

3.2 Stable Stochastic Sampling

We are concerned with the use of temporal filtering for denoising of techniques with stochastic sampling. Specifically, we are interested in cases where the shading function f_i does not only depend on the surface sample $G_{i,j}$ but also on a vector of random numbers $\xi_{i,j}$ (examples can be found in Section 4).

In this setting, we define the temporal gradient as random variable

$$\delta_{i,\vec{j}} := f_i(\vec{G}_{i-1,j}, \xi_{i,\vec{j}}) - f_{i-1}(G_{i-1,j}, \xi_{i-1,j}), \tag{3}$$

where $\xi_{i,\vec{j}}$ and $\xi_{i-1,j}$ are independent random variables. The variance of this temporal gradient is given by

$$\operatorname{Var}(\delta_{i,\vec{j}}) = \operatorname{Var}(f_i(\vec{G}_{i-1,j}, \xi_{i,\vec{j}})) + \operatorname{Var}(f_{i-1}(G_{i-1,j}, \xi_{i-1,j})) \tag{4}$$

$$- 2 \cdot \operatorname{Cov}(f_i(\vec{G}_{i-1,j}, \xi_{i,\vec{j}}), f_{i-1}(G_{i-1,j}, \xi_{i-1,j})). \tag{5}$$

Since the covariance resulting from the independent random variables $\xi_{i,\vec{j}}$ and $\xi_{i-1,j}$ vanishes, the variance of the temporal gradient is always worse than that of the contributing shading samples.

A large covariance is desirable to diminish this variance. Thus, we apply forward projection to carry along the random number $\xi_{i-1,j}$ used in the previous frame as in temporal gradient-domain path tracing [Manzi et al. 2016]. It overrides the random numbers used for the reprojected sample in the new frame:

$$\xi_{i,\vec{j}} := \xi_{i-1,j} \tag{6}$$

We expect a high degree of temporal coherence leading to a large correlation between f_i and f_{i-1}. In particular, if the relevant parts of the scene do not change, we reliably obtain a temporal gradient of zero and maximize temporal reuse. With greater changes in the scene, noise in the temporal gradient increases but unless the shading functions across two frames are anticorrelated, our technique still benefits from reuse of random numbers.

Instead of storing the full sequence of random numbers, we only store one seed for a pseudo-random number generator per pixel. One might consider computing the seed entirely from the involved indices but such attempts fail because seeds may be older than one frame.

3.3 Constructing Gradient Samples

Having defined the temporal gradient, we now discuss its computation. In each frame, the first step is to render a new visibility buffer and to generate new seeds (Figure 2c). Rather than affording one sample of the temporal gradient per pixel, we only repurpose part of our shading budget to evaluate gradient samples sparsely. We define strata of 3×3 pixels. In each of them, we randomly select one pixel j from the previous frame that is to be reprojected (Figure 2a). Through this stratified sampling, we trade aliasing for temporally incoherent noise.

Next, we apply forward projection to these samples to determine their screen space locations in the current frame (Figure 2b). We use the depth buffer of the current frame to discard reprojected surface samples which are occluded in the current frame. The other surface samples and seeds are merged into the new visibility buffer at the appropriate pixel $\vec{j}$ (Figure 2d). Per stratum, we allow no more than one gradient sample. However, the reprojection may map multiple samples to the same stratum. We resolve such conflicts efficiently by means of GPU atomics. Whichever sample finishes the reprojection computations first will be merged into the visibility buffer.

We reproject the shading samples $f_{i-1}(G_{i-1,j}, \xi_{i-1,j})$ from the previous frame in the same manner as the visibility information, without interpolation (Figure 2e). The shading function of the current frame f_i is applied to all surface samples in the visibility buffer. In particular, this yields shading samples for the reprojected surface samples $f_i(\vec{G}_{i-1,j}, \xi_{i-1,j})$ (Figure 2f). Now, a simple subtraction produces the gradient samples (Equation 3 and Figure 2g).

Note that all shading samples for the reprojected surface samples are valid shading samples for the new frame. They sample a visible surface within the pixel footprint, only the sample location is not at the pixel center. Thus, our strategy does not introduce gaps into the frame buffer that otherwise would need to be filled. Nonetheless, shading samples resulting from new surface samples and random numbers are preferable. If random numbers are reused, temporal filters gain less new information. Besides, low-discrepancy properties of random numbers are diminished if some of them are remnants of a previous frame. At a stratum size of 2×2, these problems can be quite pronounced but at 3×3 the fraction of reprojected samples is small enough (Figure 5).

Although the shading samples are dense, the gradient samples are sparse. By construction, we have at most one sample per stratum but due to the reprojection and occlusions, there may be gaps.

3.4 Reconstruction of the Temporal Gradient

The gradient samples are not only sparse and irregular but also noisy. We need a reconstruction to obtain a dense, denoised estimate of the temporal gradient. This reconstruction needs to be efficient, edge-preserving and it needs to support large filter regions to obtain enough samples per pixel. Since SVGF [Schied et al. 2017] addresses very similar challenges, we take a similar approach.

Like SVGF, our technique applies an edge-aware À-trous wavelet transform [Dammertz et al. 2010]. This filter performs a cross-bilateral transformation over multiple iterations. To achieve a large filter region efficiently, taps are spread apart further in each iteration. Our gradient reconstruction is joint-bilateral; we filter the luminance and simultaneously use it to derive filter weights used for reconstruction of the gradient and luminance samples.

We store the shading and gradient samples into a regular grid at stratum resolution. These buffers are indexed by stratum coordinates p or q. For empty strata the gradient is set to zero. The per-stratum luminance estimate $\hat{l}^{(0)}$ is initialized as the average luminance of all samples inside of one stratum. The initial variance estimate $\mathrm{Var}(\hat{l}^{(0)})$ is the variance of the luminance inside of one stratum. Iteration $k \in \{0, \ldots, 4\}$ of the filter to reconstruct luminance is defined as a weighted sum with explicit renormalization:

$$\hat{l}^{(k+1)}(p) := \frac{\sum_{q\in\Omega} h^{(k)}(p,q) \cdot w^{(k)}(p,q) \cdot \hat{l}^{(k)}(q)}{\sum_{q\in\Omega} h^{(k)}(p,q) \cdot w^{(k)}(p,q)} \tag{7}$$

where Ω is the set of all stratum coordinates and $h^{(k)}(p,q)$ is the sparse filter kernel defined by the À-trous wavelet transform. For $h^{(0)}(p,q)$, we use a simple 3×3 box filter and as k grows its non-zero entries are spread apart.

The weight function

$$w^{(k)}(p,q) := w_z(p,q) \cdot w_n(p,q) \cdot w_l^{(k)}(p,q) \tag{8}$$

combines multiple edge-stopping functions controlled by user parameters $\sigma_n, \sigma_z, \sigma_l > 0$, which are chosen as for SVGF [Schied et al. 2017]. The first uses world-space normals n:

$$w_n(p,q) := \max(0, \langle n(p),\, n(q)\rangle)^{\sigma_n} \tag{9}$$

The second incorporates differences in depth z, normalized by the screen-space derivative of the depth to account for oblique surfaces:

$$w_z(p,q) := \exp\left(-\frac{|z(p) - z(q)|}{\sigma_z \cdot |\langle \nabla z(p),\, p - q\rangle|}\right) \tag{10}$$

The third one uses the filtered luminance of the previous iteration. It is normalized using the standard deviation of the luminance, filtered by a 3×3 Gaussian blur:

$$w_l^{(k)}(p,q) := \exp\left(-\frac{\left|\hat{l}^{(k)}(p) - \hat{l}^{(k)}(q)\right|}{\sigma_l \cdot \sqrt{g_{3\text{x}3}(\mathrm{Var}(l^{(k)}(p)))}}\right) \tag{11}$$

The temporal gradient is filtered in the same manner as the luminance. In particular, it uses the weights that result from the luminance reconstruction:

$$\hat{\delta}^{(k+1)}(p) := \frac{\sum_{q\in\Omega} h^{(k)}(p,q) \cdot w^{(k)}(p,q) \cdot \hat{\delta}^{(k)}(q)}{\sum_{q\in\Omega} h^{(k)}(p,q) \cdot w^{(k)}(p,q)} \tag{12}$$

3.5 Controlling the Temporal Accumulation Factor

The reconstructed temporal gradient provides a denoised estimate of the *absolute* change of the shading function. To control the temporal filter, we are more interested in the relative rate of change. Therefore, we sample an additional normalization factor:

$$\Delta_{i,\vec{j}} := \max\left(f_i(\vec{G}_{i-1,j},\, \xi_{i-1,j}), f_{i-1}(G_{i-1,j},\, \xi_{i-1,j})\right) \tag{13}$$

Again, empty strata are treated as zero. It is reconstructed using the same joint-bilateral filter (Equation 12) yielding $\hat{\Delta}_i(p)$. We use it to define the dense, normalized history weight:

$$\lambda(p) := \min\left(1, \frac{|\hat{\delta}_i(p)|}{\hat{\Delta}_i(p)}\right) \tag{14}$$

Since empty strata have no contribution in $\hat{\delta}_i(p)$ or $\hat{\Delta}_i(p)$, the density of gradient samples cancels out in the quotient and holes are filled automatically. For disoccluded regions this hole filling produces meaningless history weights but the history is dropped in any case. The adaptive temporal accumulation factor is defined as

$$\alpha_i(p) := (1 - \lambda(p)) \cdot \alpha + \lambda(p). \tag{15}$$

It blends continuously between the global α parameter and a complete rejection of the temporal history. To account for reconstruction error, we use the maximum α_i over a 3×3-neighborhood of p.

Once available for each pixel, the temporal accumulation factor is used by temporal filters. In particular, we use it to replace the constant α in the exponential moving average with backprojection defined in Equation 1.

4 APPLICATIONS

Thus far, we have not assumed any particular stochastic shading function. Any deferred renderer may be augmented with our gradient estimation. To validate the approach, we now apply it to various light transport problems in two different ways.

Reconstruction of Soft Shadows. While ray tracing provides a simple physically based solution to render shadows, the ray budget in real-time applications is very low. For sufficient quality, extensive spatial and temporal filtering is needed. We render ray traced soft shadows with one ray per pixel using OptiX [Parker et al. 2010]. Three random numbers are used to select an area light and a point sample on this light source. These pseudorandom numbers use blue noise dithered sampling [Georgiev and Fajardo 2016; Ulichney 1993] in conjunction with the Halton sequence [Halton 1964]. For reconstruction, we use our novel A-SVGF which extends SVGF [Schied et al. 2017]. Our implementation is different from the original technique in that the edge-aware À-trous wavelet uses a simple 3×3 box filter. Over the five levels of the transform, this leads to an effective filter region of 49×49 pixels. We found that this barely reduces the quality but yields a notable speedup. The fixed temporal accumulation factor α used in SVGF is replaced by our adaptive factor $\alpha_i(p)$. This way, we diminish ghosting artifacts in dynamic parts and increase the effective sample count in static parts. When the history is dropped, A-SVGF performs the variance estimate spatially. This tends to introduce stronger spatial blur but the overall noise remains acceptable.

Real-Time Path Tracing. The distributed ray tracing approach used for the soft shadows is very general. For example, we could apply it to render glossy reflections, ambient occlusion and diffuse global illumination. Rather than handling all of these effects explicitly, we skip forward to a genuine path tracer, much like the one used with SVGF [Schied et al. 2017]. Primary visibility is still handled through rasterization into the visibility buffer. From there, we trace one ray to gather one bounce of indirect illumination. We apply path-space regularization [Kaplanyan and Dachsbacher 2013] after the first scattering event. Additionally, we trace a shadow ray from each of the two surface intersections. Again we use our novel A-SVGF for the reconstruction. However, we do not separate direct and indirect illumination for separate filtering. This makes the filtering more efficient but also more challenging.

Improving Temporal Stability of a Recurrent Autoencoder (RAE). A recurrent autoencoder offers a different approach for reconstruction of real-time path tracing results [Chaitanya et al. 2017]. We use it with the ray and path tracers described above. While individual frames of the output tend to look compelling, a common artifact is temporally unstable, low-frequency noise. Temporal anti-aliasing fails to remove this artifact because it is blind to noise at a large scale. Thus, our adaptive recurrent autoencoder (A-RAE) applies additional temporal filtering to improve the stability of the results. The noisy path tracer output is used for the temporal gradient estimation leading to an adaptive accumulation factor. Then this accumulation factor is used to perform temporal accumulation on the output of the recurrent autoencoder. We use the pre-trained autoencoder provided to us by the authors [Chaitanya et al. 2017].

5 EVALUATION AND DISCUSSION

We now evaluate our filters A-SVGF and A-RAE (see Section 4) in comparison to their predecessors [Chaitanya et al. 2017; Schied et al. 2017] and converged ground truth. We implemented our filter in the Falcor framework [Benty et al. 2017] using OpenGL fragment shaders. For SVGF and A-SVGF, we use identical filter parameters except for the temporal accumulation factor that we set to $\alpha = 0.2$ and $\alpha = 0.1$, respectively. Unless otherwise stated, we divided the screen into 3×3-strata. The filters are evaluated with respect to image quality (Section 5.1), temporal stability (Section 5.2), and performance (Section 5.3). Section 5.4 discusses the limitations of our approach.

5.1 Image Quality

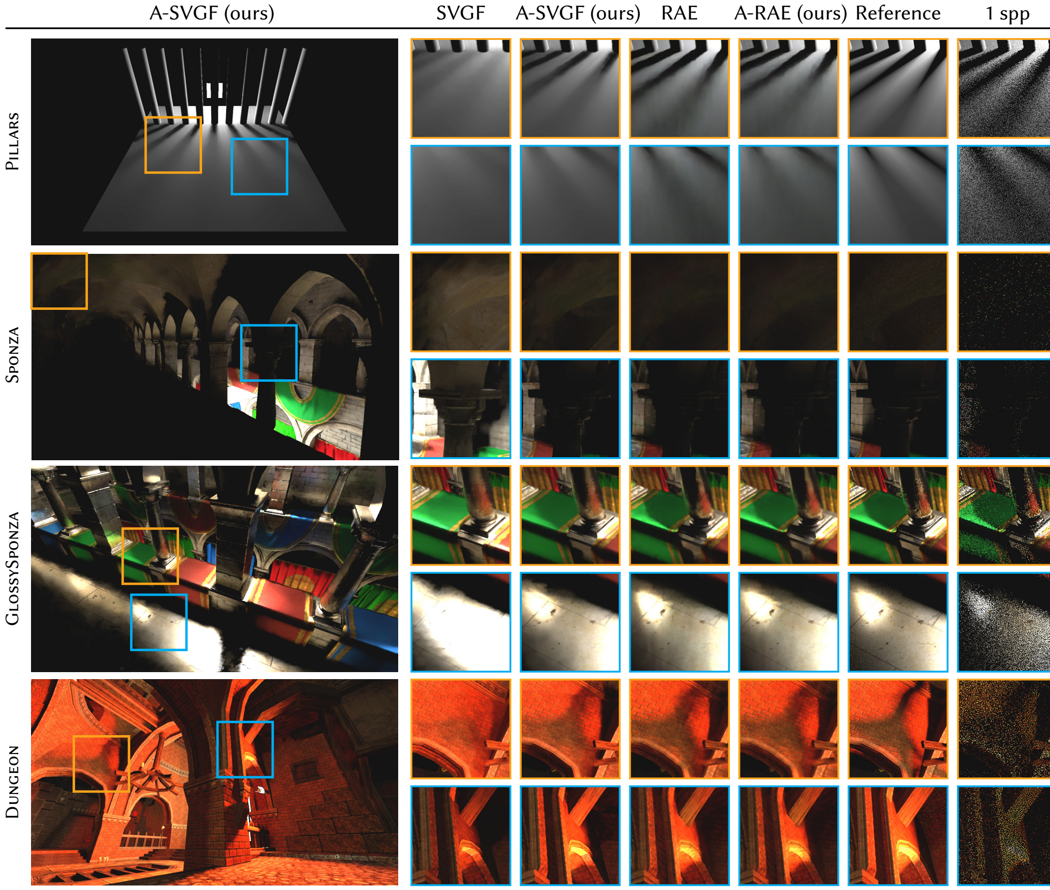
Figure 3 compares the various reconstruction filters in animated scenes. The main improvement of our A-SVGF over SVGF is an overall reduction of lag and ghosting. Correspondingly, the RMSE and SSIM [Wang et al. 2004] error metrics indicate that A-SVGF is closer to the ground truth across all scenes. A-RAE improves temporal stability without introducing noticeable temporal lag or ghosting. As this is hard to convey with still-images, please refer to Section 5.2 and the supplemental video. The fact that A-RAE does not improve individual frames over RAE is reflected by the similar RMSE and SSIM. Indeed, the metrics get slightly worse due to mildly increased temporal and spatial blur but in practice the improved temporal stability is far more noticeable.

To assess the quality of our soft shadows, we use the Pillars-scene which contains eleven pillars moving left to right in front of an area light. The fixed temporal accumulation of SVGF causes the shadows to lag behind the moving geometry (orange inset) as well as a loss of structure of the shadow in the penumbra (blue inset). A-SVGF adapts to the moving shadow, causing the temporal filter to drop the stale history information. In regions where it lacks history, it has to rely on its spatial variance estimate instead, which results in a slight overblur of the contact shadows (orange inset). Similarly, A-RAE drops the history close to the occluders keeping hard contact shadows intact while increasing the temporal stability in the smoother penumbra regions.

In Sponza, we evaluate our techniques with rapidly-changing diffuse direct and indirect illumination. The camera flies through the corridor while an area light drops to the ground quickly. The rapid shift of the illumination, in combination with the temporal lag and camera motion causes objectionable artifacts for SVGF. It drops history in disoccluded regions only, which leads to differently shaded regions with sharp boundaries (blue inset). Our A-SVGF correctly recognizes where history needs to be dropped leading to a sufficiently fast response everywhere.

We use a fast camera animation in GlossySponza as an isolated test-case for direct and indirect glossy reflections. SVGF vastly over-blurs the glossy reflections as it does not employ separate reprojections for glossy and diffuse BRDF terms [Mara et al. 2017]. Note that a reprojection could only help in reducing the trails of the moving highlights but, in contrast to our gradient estimation, does not incorporate global effects.

The Dungeon-scene combines all these challenging situations. It is illuminated using multiple static area light sources. Additionally, we use one rapidly moving red area light and a flickering, blue area light. Figure 1 depicts multiple frames of the animation (refer to the supplemental material for the full animation). SVGF over-blurs moving shadows (Figure 3, orange inset) and highlights (blue inset) and shows objectionable ghosting artifacts (Figure 1). Despite the large amount of noise, our gradient estimation reliably detects fast changes of the shading function and controls the temporal filter accordingly. Figure 4 evaluates the reconstruction error for an excerpt of the full animation using the SSIM error metric [Wang et al. 2004]. A-SVGF consistently has the lowest error over the whole animation. For all filters except SVGF, which cannot handle the fast animations, the error spikes at similar animation times, caused by the discontinuous animations that have a global



Filter	Pillars		Sponza		GlossySponza		Dungeon	
	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM	RMSE	SSIM
SVGF	0.0178	0.9539	0.0753	0.4315	0.1313	0.7413	0.1003	0.7627
A-SVGF	0.0152	0.9682	0.0227	0.9227	0.0507	0.7817	0.0540	0.8603
RAE	0.0133	0.9817	0.0189	0.9164	0.0451	0.8079	0.0776	0.8230
A-RAE	0.0139	0.9732	0.0222	0.9193	0.0531	0.8014	0.0854	0.8032

Fig. 3. Frames captured from rapidly animated sequences, rendered at 1280 × 720 at 30 Hz. All filters are provided with identical 1 spp input. Pillars uses one shadow ray per pixel, the other scenes use the path tracer described in Section 4. In contrast to our A-SVGF, SVGF shows significant lag as it cannot drop stale history information. Despite the compelling results of RAE in still-images, it exhibits temporal instabilities (please refer to the supplemental video) visible as low-frequency noise under animation. Our temporally-filtered A-RAE increases stability without introducing temporal filtering artifacts.

effect on the shading. A-RAE sees a minor improvement over RAE, confirming that our filter does not introduce artifacts, but overall performs worse than A-SVGF. This is mostly visible as a slight color shift that we attribute to insufficient training data.

Figure 5 compares results for different choices of the stratum size. If the strata are too large, resolution of the temporal gradients is insufficient and small regions with large gradients cannot be resolved properly. On the other hand, repurposing too many shading samples for the temporal gradients destroys blue noise properties of our random numbers and yields less new information (Section 3.3). At 2 × 2 up to 25% of all samples are gradient samples and thus problems are likely to

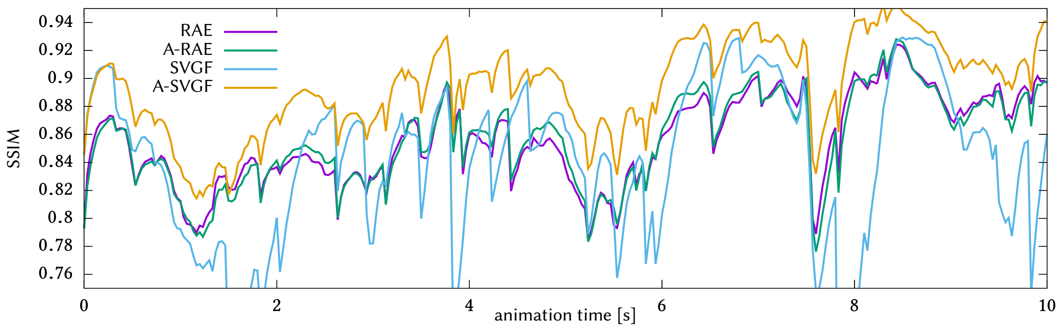


Fig. 4. Image quality comparison at 1280 × 720 at 30 Hz compared to a 1024 spp reference using SSIM [Wang et al. 2004] in the animated sequence shown in Figure 1. The adaptive temporal filter in A-SVGF quickly drops stale history information and consistently shows improved image similarity compared to SVGF. Temporally filtering the output of RAE introduces only negligible amounts of lag, shows minor improvements in image similarity, and significantly improves temporal stability (see Figure 6).

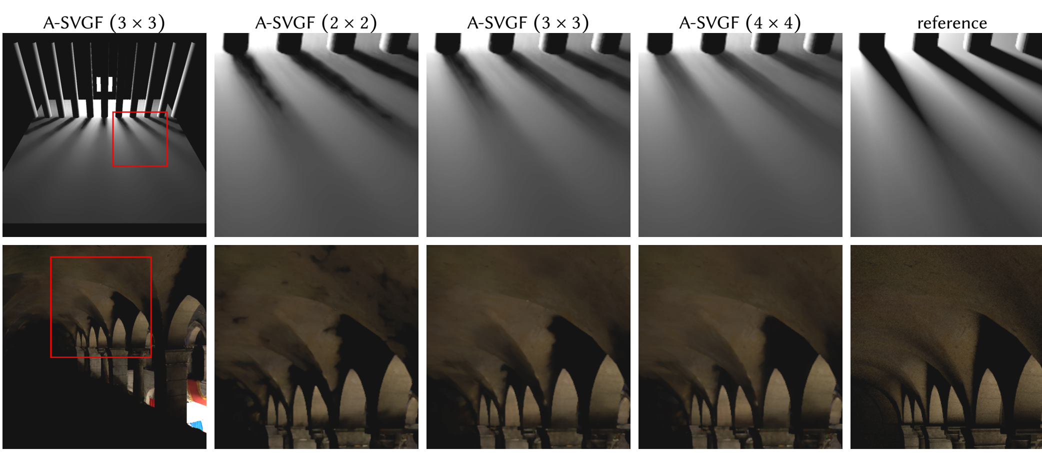


Fig. 5. Results in the animated Pillars and Sponza scenes for several stratum sizes. Large strata (4 × 4) imply a lower spatial resolution for the temporal gradients. For the Pillars scene, this effectively shrinks the umbra since low gradients leak in. On the other hand, too many gradient samples (2 × 2) are detrimental to temporal accumulation in the stochastic sampling (Section 3.3). In Sponza this leads to dark splotches on the ceiling where the light source has not been found. Based on these observations, we choose 3 × 3 for the other results.

occur. At 3 × 3 at most 11% of the samples are gradient samples but the resolution of the gradients is still satisfactory.

5.2 Temporal Stability

Training (recurrent) autoencoders to be temporally stable is challenging due to the lack of an appropriate error metric. We therefore demonstrate how to improve temporal stability using our filter. For similar reasons, we assess the improvement in temporal stability using a static scene configuration (see Figure 6). We measure the perceptual difference between consecutive frames using SSIM. SVGF and A-SVGF perform a switch from the spatial to the temporal estimate, which introduces less blur. Switching the variance estimates causes a momentary drop in temporal

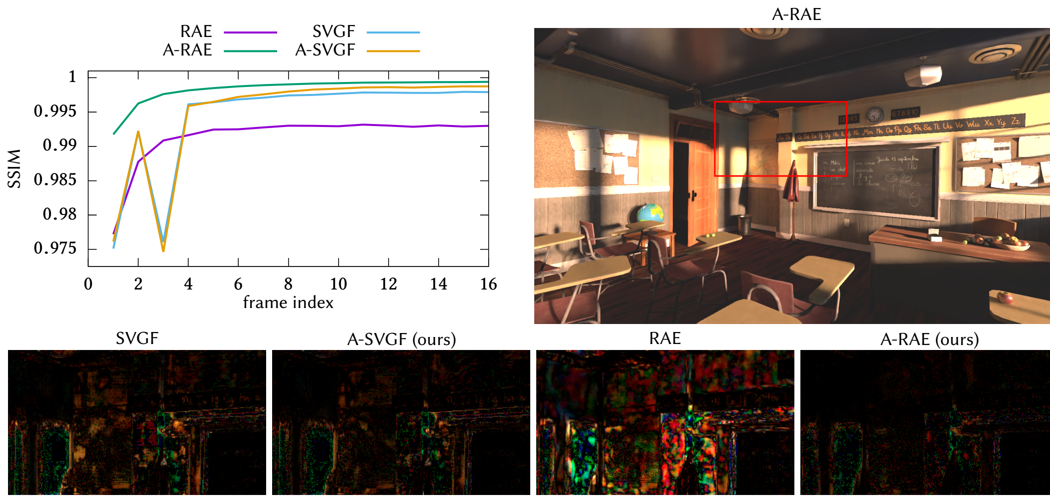


Fig. 6. We evaluate the temporal stability by computing a perceptual difference [Wang et al. 2004] between consecutive frames for the static scene depicted on the right. Crops show a magnified difference between frame 12 and 13. Our adaptive temporal filter (A-RAE) significantly improves temporal stability over RAE without introducing visible temporal lag. See the supplemental video for the full sequence and an animated test case.

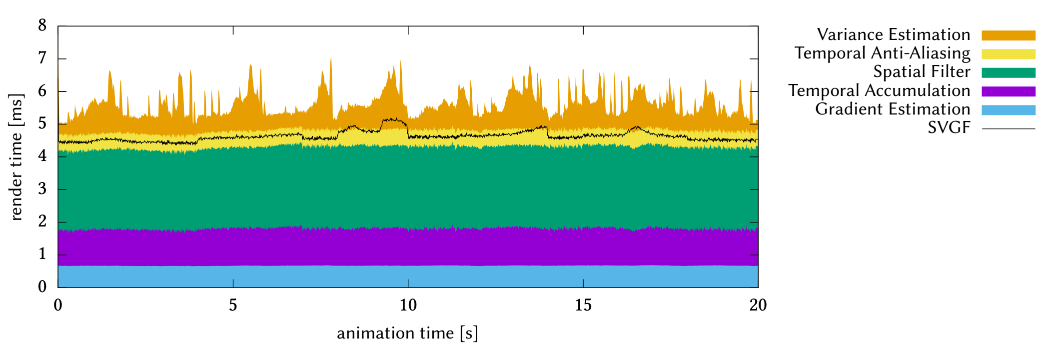


Fig. 7. Runtime breakdown of A-SVGF with our gradient estimation compared to SVGF using an an animated flythrough in the Dungeon-scene. The animation was rendered at 1920 × 1080 at 60 Hz on a NVIDIA Titan Xp. The cost for the gradient estimation and temporal filtering is mostly constant. However, since the temporal filter is dropping history, the spatial fallback for the variance estimate is used more frequently which is more costly than the temporal estimate.

stability after sufficient history acquisition. However, this is only perceptible if there are large regions switching the estimate simultaneously.

The temporal stability of A-RAE is significantly improved and already surpasses RAE after only a single frame. After the initial convergence phase, SVGF and A-SVGF are close to A-RAE. The insets in Figure 6 show the magnified difference between two consecutive frames and exhibit the splotchy characteristic of RAE.

5.3 Performance

To assess the performance characteristics of our adaptive temporal filter, we measure a breakdown of the frame times of A-SVGF in comparison to SVGF (Figure 7). Our measurement was performed on an NVIDIA Titan Xp at a screen-resolution of 1920×1080 and uses our animation in the Dungeon-scene (refer to the supplemental video), rendered at 60 Hz. Our temporal filter consistently runs in less than 2 ms with only small variations whereas the whole gradient sampling and reconstruction pipeline accounts for less than 1 ms. The main overhead of A-SVGF compared to SVGF is the gradient estimation. However, the filter more frequently drops the history and causes A-SVGF to rely more heavily on its spatial variance estimate. The frame time of RAE is constant at 191 ms and independent of the input. Thus, the overhead of applying our temporal filter in A-RAE is negligible.

5.4 Limitations

In Section 3.2, we discuss our goal to maximize the correlation between shading functions of the current and the previous frame. This requires the primary sample space to stay temporally coherent across frames. Changes (e.g. due to added light sources or different importance sampling) will be reflected in the temporal gradient, and potentially cause the history to be dropped. Usually such changes reflect a changed importance of particular phenomena, therefore depending on the situation, this behaviour may or may not be desirable. With the current technique, the user should avoid unnecessary changes to the primary sample space. This could be addressed by carrying along more information about samples from the previous frame. For example, light source indices may be stored besides the seeds for the random numbers. Then the shading sample for the temporal gradient would be forced to reuse this light rather than selecting it based on stable random numbers.

Our approach has some inherent limitations due to the screen-space temporal accumulation. We need stable sampling positions and thus rely on a forward-projection. Compared to backward-motion vectors, a forward projection is more involved. The gradient sampling and reconstruction as described in this paper cannot be used with stochastically sampled primary visibility, e.g. motion blur or depth of field. One possible solution is to rely on texture-space sampling and reconstruction instead [Munkberg et al. 2016].

6 CONCLUSION

Our adaptive temporal filter is able to detect changes reliably in spite of strong noise in the stochastic shading. While previous reconstruction filters for real-time path tracing have significantly benefited from their temporal filtering, the issues of temporal blur and stability remained largely unaddressed. We augment these techniques with a widely applicable and highly efficient solution that removes most of these artifacts. Thus, our techniques are applicable to scenarios with rapid action. Furthermore, the game industry is eager to embrace more dynamic and physically accurate solutions for global illumination [Barré-Brisebois 2017] and we are hopeful that our work will be a key component in enabling them to do so. Though, our work is not limited to shading based on ray tracing.

While we have focused on an implementation with a visibility buffer and forward projection, it would also be interesting to use stable sample locations defined differently [Corso et al. 2017] or to work in texture space [Munkberg et al. 2016]. Another promising direction might be to control adaptive sampling according to the available history information.

ACKNOWLEDGEMENTS

We thank Petrik Clarberg and Marco Salvi at NVIDIA Corporation for the helpful discussions. Additionally, we thank Aaron Lefohn at NVIDIA Corporation for supporting our research. We thank Simon O'Callaghan for granting us permission to use the DUNGEON scene from the Quake mod *Arcane Dimensions* and thank Tobias Rapp for adapting it to our renderer. Thanks to Frank Meinl and Efgeni Bischoff for the SPONZA scene, Guillermo M. Leal Llaguno for the SANMIGUEL scene and Christophe Seux for the original CLASSROOM scene. Thanks also to Anjul Patney and Nicholas Hull who adapted the scenes for our renderer.

REFERENCES

Colin Barré-Brisebois. 2017. A Certain Slant of Light: Past, Present and Future Challenges of Global Illumination in Games, Open Problems in Real-time Rendering. In *ACM SIGGRAPH 2017 Courses (SIGGRAPH '17)*. ACM. https://doi.org/10.1145/3084873.3127940

Nir Benty, Kai-Hwa Yao, Tim Foley, Conor Lavelle, and Chris Wyman. 2017. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor

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 (2013), 55–69. http://jcgt.org/published/0002/02/04/

Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. 2017. Interactive Reconstruction of Monte Carlo Image Sequences Using a Recurrent Denoising Autoencoder. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 36, 4 (2017), 98:1–98:12. https://doi.org/10.1145/3072959.3073601

Alessandro Dal Corso, Marco Salvi, Craig Kolb, Jeppe Revall Frisvad, Aaron Lefohn, and David Luebke. 2017. Interactive Stable Ray Tracing. In *Proc. of High Performance Graphics (HPG '17)*. 1:1–1:10. https://doi.org/10.1145/3105762.3105769

Holger Dammertz, Daniel Sewtz, Johannes Hanika, and Hendrik Lensch. 2010. Edge-Avoiding À-Trous Wavelet Transform for fast Global Illumination Filtering. In *Proc. of High Performance Graphics (HPG '10)*. 67–75. https://doi.org/10.2312/EGGH/HPG10/067-075

Mauricio Delbracio, Pablo Musé, Antoni Buades, Julien Chauvier, Nicholas Phelps, and Jean-Michel Morel. 2014. Boosting Monte Carlo Rendering by Ray Histogram Fusion. *ACM Transactions on Graphics* 33, 1 (2014), 8:1–8:15. https://doi.org/10.1145/2532708

Iliyan Georgiev and Marcos Fajardo. 2016. Blue-noise Dithered Sampling. In *ACM SIGGRAPH 2016 Talks*. 35:1–35:1. https://doi.org/10.1145/2897839.2927430

Tobias Günther and Thorsten Grosch. 2015. Consistent Scene Editing by Progressive Difference Images. *Computer Graphics Forum* 34, 4 (2015), 41–51. https://doi.org/10.1111/cgf.12677

John H. Halton. 1964. Algorithm 247: Radical-Inverse Quasi-Random Point Sequence. *Commun. ACM* 7, 12 (1964), 701–702. https://doi.org/10.1145/355588.365104

Jose A. Iglesias-Guitian, Bochang Moon, Charalampos Koniaris, Eric Smolikowski, and Kenny Mitchell. 2016. Pixel History Linear Models for Real-Time Temporal Filtering. *Computer Graphics Forum (Proc. Pacific Graphics)* 35, 7 (2016), 363–372. https://doi.org/10.1111/cgf.13033

Anton S. Kaplanyan and Carsten Dachsbacher. 2013. Path Space Regularization for Holistic and Robust Light Transport. *Computer Graphics Forum (Proc. of Eurographics)* 32, 2 (2013), 63–72. https://doi.org/10.1111/cgf.12026

Brian Karis. 2014. High-Quality Temporal Supersampling. In *ACM SIGGRAPH 2014 Courses: Advances in Real-time Rendering in Games, Part I (SIGGRAPH '14)*. https://doi.org/10.1145/2614028.2615455

Marco Manzi, Markus Kettunen, Frédo Durand, Matthias Zwicker, and Jaakko Lehtinen. 2016. Temporal Gradient-domain Path Tracing. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 35, 6 (2016), 246:1–246:9. https://doi.org/10.1145/2980179.2980256

Michael Mara, Morgan McGuire, Benedikt Bitterli, and Wojciech Jarosz. 2017. An Efficient Denoising Algorithm for Global Illumination. In *Proc. of High Performance Graphics (HPG '17)*. 3:1–3:7. https://doi.org/10.1145/3105762.3105774

Mark Meyer and John Anderson. 2006. Statistical Acceleration for Animated Global Illumination. *ACM Transactions on Graphics* 25, 3 (2006), 1075–1080. https://doi.org/10.1145/1141911.1141996

Bochang Moon, Jose A Iglesias-Guitian, Sung-Eui Yoon, and Kenny Mitchell. 2015. Adaptive Rendering with Linear Predictions. *ACM Transactions on Graphics* 34, 4 (2015), 121:1–121:11. https://doi.org/10.1145/2766992

Jacob Munkberg, Jon Hasselgren, Petrik Clarberg, Magnus Andersson, and Tomas Akenine-Möller. 2016. Texture Space Caching and Reconstruction for Ray Tracing. *ACM Transactions on Graphics* 35, 6 (2016), 249:1–249:13. https://doi.org/10.1145/2980179.2982407

Diego Nehab, Pedro V. Sander, Jason Lawrence, Natalya Tatarchuk, and John R. Isidoro. 2007. Accelerating Real-time Shading with Reverse Reprojection Caching. In *SIGGRAPH/Eurographics Workshop on Graphics Hardware*. 25–35. https:

//doi.org/10.2312/EGGH/EGGH07/025-036

Steven G. Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, and Martin Stich. 2010. OptiX: A General Purpose Ray Tracing Engine. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 29, 4 (2010), 66:1–66:13. https://doi.org/10.1145/1778765.1778803

Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. 2016. Towards Foveated Rendering for Gaze-Tracked Virtual Reality. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 35, 6 (2016), 179:1–179:12. https://doi.org/10.1145/2980179.2980246

Fabrice Rousselle, Wojciech Jarosz, and Jan Novák. 2016. Image-space Control Variates for Rendering. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 35, 6 (2016), 169:1–169:12. https://doi.org/10.1145/2980179.2982443

Takafumi Saito and Tokiichiro Takahashi. 1990. Comprehensible Rendering of 3-D Shapes. *Computer Graphics (Proc. SIGGRAPH)* (1990), 197–206. https://doi.org/10.1145/97880.97901

Daniel Scherzer, Lei Yang, Oliver Mattausch, Diego Nehab, Pedro V. Sander, Michael Wimmer, and Elmar Eisemann. 2012. Temporal Coherence Methods in Real-Time Rendering. *Computer Graphics Forum* 31, 8 (2012), 2378–2408. https://doi.org/10.1111/j.1467-8659.2012.03075.x

Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R. Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvi. 2017. Spatiotemporal Variance-guided Filtering: Real-time Reconstruction for Path-traced Global Illumination. In *Proc. of High Performance Graphics (HPG '17)*. 2:1–2:12. https://doi.org/10.1145/3105762.3105770

Robert A. Ulichney. 1993. Void-and-cluster method for dither array generation. *Proc. SPIE* 1913 (1993), 1913 – 1913 – 12. https://doi.org/10.1117/12.152707

Bruce Walter, George Drettakis, and Steven Parker. 1999. Interactive Rendering Using the Render Cache. In *Eurographics Workshop on Rendering*, Vol. 10. 19–30. https://doi.org/10.2312/EGWR/EGWR99/019-030

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612. https://doi.org/10.1109/TIP.2003.819861

Kai Xiao, Gabor Liktor, and Karthik Vaidyanathan. 2018. Coarse Pixel Shading with Temporal Supersampling. In *Proc. ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D '18)*. ACM, Article 1, 7 pages. https://doi.org/10.1145/3190834.3190850

Lei Yang, Diego Nehab, Pedro V Sander, Pitchaya Sitthi-amorn, Jason Lawrence, and Hugues Hoppe. 2009. Amortized Supersampling. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 28, 5 (2009), 135:1–135:12. https://doi.org/10.1145/1618452.1618481

Henning Zimmer, Fabrice Rousselle, Wenzel Jakob, Oliver Wang, David Adler, Wojciech Jarosz, Olga Sorkine-Hornung, and Alexander Sorkine-Hornung. 2015. Path-space Motion Estimation and Decomposition for Robust Animation Filtering. *Proc. Eurographics Symposium on Rendering* 34, 4 (2015), 131–142. https://doi.org/10.1111/cgf.12685

Matthias Zwicker, Wojciech Jarosz, Jaakko Lehtinen, Bochang Moon, Ravi Ramamoorthi, Fabrice Rousselle, Pradeep Sen, Cyril Soler, and S-E Yoon. 2015. Recent Advances in Adaptive Sampling and Reconstruction for Monte Carlo Rendering. *Computer Graphics Forum (STAR)* 34, 2 (2015), 667–681. https://doi.org/10.1111/cgf.12592