Combining Analytic Direct Illumination and Stochastic Shadows

Eric Heitz
Unity Technologies

Stephen Hill
Lucasfilm

Morgan McGuire
NVIDIA

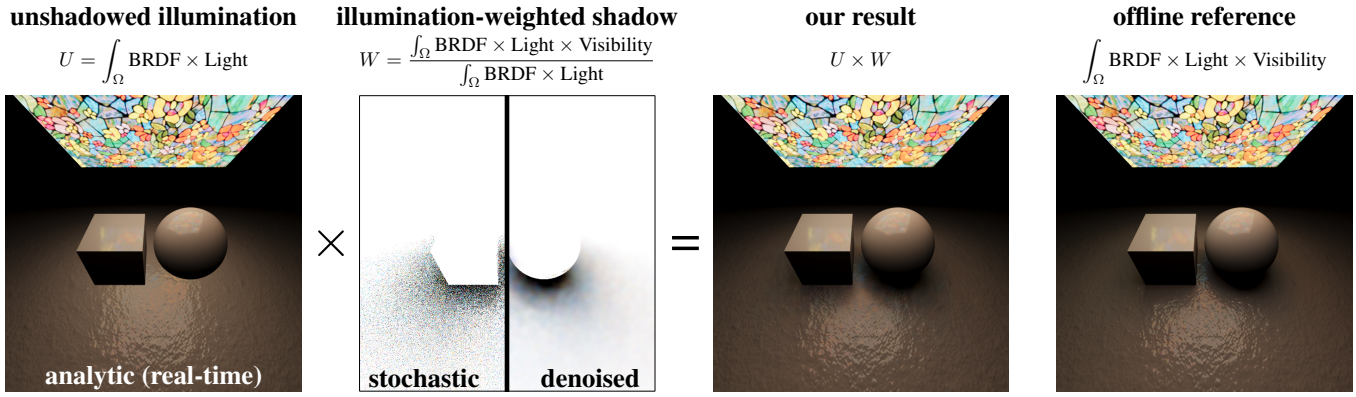

Figure 1: Illustration of our formulation. *We formulate direct illumination as the product of the* unshadowed illumination *and the* illumination-weighted shadow*, and we combine real-time shadowless illumination techniques with raytraced and denoised shadows. This yields a* ratio estimator *of direct illumination that has low variance and can be denoised without blurring shading details.*

Abstract

In this paper, we propose a ratio estimator of the direct-illumination equation that allows us to combine analytic illumination techniques with stochastic raytraced shadows while maintaining correctness. Our main contribution is to show that the shadowed illumination can be split into the product of the *unshadowed illumination* and the *illumination-weighted shadow*. These terms can be computed separately — possibly using different techniques — without affecting the exactness of the final result given by their product.

This formulation broadens the utility of analytic illumination techniques to raytracing applications, where they were hitherto avoided because they did not incorporate shadows. We use such methods to obtain sharp and noise-free shading in the unshadowed-illumination image and we compute the weighted-shadow image with stochastic raytracing. The advantage of restricting stochastic evaluation to the weighted-shadow image is that the final result exhibits noise only in the shadows. Furthermore, we denoise shadows separately from illumination so that even aggressive denoising only overblurs shadows, while high-frequency shading details (textures, normal maps, etc.) are preserved.

1 Introduction

This research is motivated by the lack of shadowing in certain state-of-the-art real-time shading techniques, such as the area-lighting framework developed by Heitz et al. [2016]. While it accurately models the effects of complex materials and lighting, the lack of shadowing ultimately produces unrealistic images (Fig. 2(a)) and this problem cannot be solved robustly by using a shadow map or a fake soft shadow (Fig. 2(b, c)). Given the recent progress in real-time path-tracing [Schied et al. 2017; Mara et al. 2017], it is tempting to believe that current real-time techniques might soon be replaced by fully stochastic approaches combined with efficient denoising algorithms. However, these stochastic techniques either suffer from noise or overblurred shading details due to the aggressive denoising that needs to be used for reconstruction (Fig. 2(d)). In summary, on the one hand we have real-time shading techniques that produce clean and sharp shading details but miss shadowing effects, and on the other we have stochastic techniques that incorporate all of the effects but yield noisy or overblurred results.

In this paper, we show how to get the best of both worlds by completing real-time shading techniques with ray-traced shadows in a way that is mathematically correct. Our main insight is the formulation of shadowed illumination as the product of the *unshadowed illumination* and the *illumination-weighted shadow*. Based on this idea, we develop the following contributions:

- A *ratio estimator* of shadowed illumination, in which shadowing is the only source of variance. The estimator produces noise-free results in unshadowed regions of the image.
- A denoising formulation that allows for denoising shadows separately from shading. Thanks to this, we can denoise our results without compromising the sharpness of shading details as only the shadows are overblurred (Fig. 2(e)). Note that any existing denoiser can benefit from this formulation.
- A real-time implementation of our estimator and a custom real-time denoiser renders high-quality results using only 1-4 shadow rays per light per pixel.

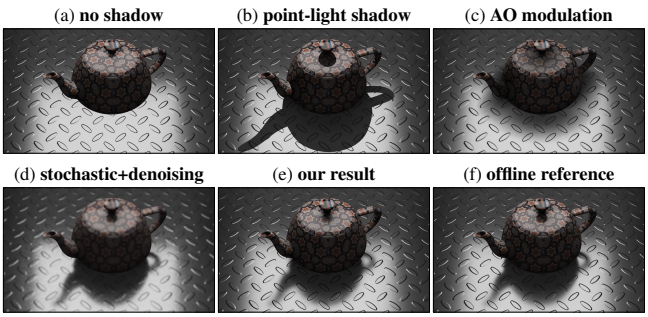

Figure 2: Comparison of real-time options for incorporating shadows in the direct illumination of an area light. *Options (d) and (e) are rendered at 2spp and denoised with the same bilateral filter.*

1.1 Mathematical Overview of our Method

Shadowed direct illumination The *shadowed direct illumination* at a point is the integral of the cosine-weighted Bidirectional Reflectance Distribution Function (BRDF), the incident lighting, and the visibility towards the light:

$$S = \int_\Omega \text{BRDF} \times \text{Light} \times \text{Visibility}. \tag{1}$$

Real-time unshadowed direct illumination Many real-time techniques provide efficient ways for computing the direct illumination with various surface materials and light types but without accounting for visibility, i.e. they compute the *unshadowed direct illumination*:

$$U = \int_\Omega \text{BRDF} \times \text{Light}. \tag{2}$$

For instance, Heitz et al. [2016] provide a closed-form evaluation for integrating the BRDF against polygonal area lights. Another typical example is the precomputation of the integral of the BRDF and an environment map using a prefiltered representation, which is a common practice in games [Kautz et al. 2000; Karis 2013]. These techniques produce shadowless results, as shown in Figure 2(a).

Problem Adding mathematically correct shadows on top of an analytic computation of U is complicated because in general the visibility is not separable:

$$S \neq \int_\Omega \text{BRDF} \times \text{Light} \times \int_\Omega \text{Visibility}. \tag{3}$$

Assuming separability of the visibility is mathematically wrong and might produce unrealistic results, as in Figure 2(c). Mathematically correct visibility is thus a stumbling block for real-time rendering and there is currently no alternative for obtaining exact and robust shadows other than turning to fully stochastic methods.

Our solution Our insight is to split the shadowed illumination into two parts:

$$S = \underbrace{\int_\Omega \text{BRDF} \times \text{Light}}_{\substack{\textit{unshadowed illumination } U \\ \text{analytic or precomputed}}} \times \underbrace{\frac{\int_\Omega \text{BRDF} \times \text{Light} \times \text{Visibility}}{\int_\Omega \text{BRDF} \times \text{Light}}}_{\substack{\textit{illumination-weighted shadow } W \\ \text{stochastic + denoising}}}. \tag{4}$$

We refer to the first part as the *unshadowed illumination* U, which we compute via existing analytic or precomputed real-time techniques as in Equation (2). The second part is the *illumination-weighted shadow* W, which we compute via stochastic raytracing followed by denoising. This formulation is inspired by the *ratio estimator*, a variance-reduction technique from the field of statistics [Hartley 1989; Mickey 1959].

Advantages This formulation provides two main advantages. First, only the illumination-weighted shadow W is computed stochastically. Consequently, only shadowed regions of the final image appear noisy and *lit regions are rendered noise-free even at 1 sample per pixel*. Second, the unshadowed illumination U containing all of the sharp shading details (textures, normals, etc.) is analytic and does not need to be denoised. Hence, we denoise only the stochastic illumination-weighted shadow W such that *agressive denoising overblurs only shadows while sharp shading details are preserved* in the final image.

2 Previous Work

2.1 Unshadowed Direct Illumination

Our technique is built on top of existing real-time techniques that compute direct illumination without shadows in an efficient manner, i.e. they provide an efficient and noise-free solution to Equation (2) and can be used in our formulation of the illumination shown in Equation (4).

Analytic illumination techniques originated from the study of form factors that yield the irradiance arriving at a point — i.e. direct illumination with a diffuse BRDF — from various geometric shapes [Baum et al. 1989]. The advantage of diffuse BRDFs is that they do not depend on the view direction, making it possible to derive the irradiance from complex emitters such as linearly varying area lights [Chen and Arvo 2000]. The concept of using analytic form factors has been extended to non-diffuse BRDFs with area lights [Arvo 1995; Snyder 1996]. For a long time these computations were too costly, so real-time rendering applications used coarser approximations [Drobot 2014], but they were recently made affordable thanks to accurate approximations [Lecocq et al. 2016]. The derivation of new directional distributions that offer analytic integration properties for area lighting is an active research topic [Heitz et al. 2016; Heitz and Hill 2017; Dupuy et al. 2017].

Precomputed illumination techniques are used for image-based emitters such as scene lighting represented as an environment map. Tyically, diffuse BRDFs use precomputed irradiance maps [Ramamoorthi and Hanrahan 2001] and glossy BRDF models use preconvolved environment maps [Kautz et al. 2000; Ramamoorthi and Hanrahan 2002; Karis 2013]. The same approach can also be used for refraction [de Rousiers et al. 2011].

2.2 Variance-Reduction Techniques

Ratio estimators Ratio estimators are variance-reduction techniques often used in statistics [Hartley 1989; Mickey 1959]. The goal is to estimate the expectation of a random variable Y by taking advantage of the information provided by a positively correlated random variable X whose expectation is known. A classic estimator consists in averaging observations y_n from Y:

$$\mathbb{E}[Y] = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} y_n. \tag{5}$$

If X and Y are positively correlated and if the expectation $\mathbb{E}[X]$ of X is known, it is possible to reduce the variance using:

$$\mathbb{E}[Y] = \mathbb{E}[X] \lim_{N\to\infty} \frac{\sum_{n=1}^{N} y_n}{\sum_{n=1}^{N} x_n}. \tag{6}$$

We obtained Equation (4) by applying this ratio estimator using the unshadowed direct illumination for X and the shadowed direct illumination for Y.

A ratio estimator has been employed previously in rendering by Stachowiak [2015], who uses a precomputed integral of the BRDF alone in order to absorb the variance due to imperfect importance sampling of the BRDF. To our knowledge, the idea of using ratio estimators when the integral of one or several terms of the integrand is known has not been used further in rendering applications.

Control variates method The main concurrent approach that takes advantage of analytic integrals is the *control variates* method, which has been used in rendering [Hery and Villemin 2013]. We

compare against in Section 3. Control variates are a variance reduction technique that consists of using an easily integrable approximation of the function to integrate, together with an error-correction term that is computed stochastically:

$$S = \underbrace{c\int_\Omega \text{BRDF}\times\text{Light}}_{\textit{control variate}\ \text{(analytic)}} \underbrace{-\int_\Omega \text{BRDF}\times\text{Light}\times(c-\text{Visibility})}_{\textit{error}\ \text{(stochastic)}}, \quad (7)$$

where c is a correlation coefficient that needs to be chosen carefully to ensure effective variance reduction. Computing the optimal value for c requires either supplemental computations or dedicated optimizations and data structures [Fan et al. 2006; Clarberg and Akenine-Moeller 2008]. In this paper, we are interested in estimators that work as a replacement for a fully stochastic evaluation without precomputations or supplemental data structures. Hence, the optimal value for c is unknown, so we use $c = 1$ by default for our comparisons. In Section 3, we show that a control variate used with a default value does not perform well. The main advantage of our formulation in Equation (4) is that it does not require finding such a coefficient and hence it does not require supplemental computations nor data structures to be effective. Furthermore, we show that our formulation allows us to make robust separate approximations or denoising for the illumination and the shadows, which is not possible with control variates.

3 Ratio Estimator

In this section, we introduce our formulation of the direct illumination, build a ratio estimator based on this formulation, and discuss its properties and performance in terms of variance reduction. For brevity we will use a more compact notation: R denotes the cosine-weighted BRDF, L the incident lighting, and V the visibility, and the shadowed and unshadowed direct illumination are

$$S = \int_\Omega R(\omega)\, L(\omega)\, V(\omega)\, d\omega, \quad (8)$$

$$U = \int_\Omega R(\omega)\, L(\omega)\, d\omega. \quad (9)$$

3.1 Previous Estimators of the Shadowed Illumination

Full-stochastic estimator A classic estimator of S is

$$S_N^{\text{sto}} = \frac{1}{N}\sum_{n=1}^{N} \frac{R(\omega_n)\, L(\omega_n)\, V(\omega_n)}{p(\omega_n)}, \quad (10)$$

where the directions ω_n are importance sampled from a Probability Density Function (PDF) p. We call it the *full-stochastic* estimator, since all of the terms are evaluated randomly.

Control-variate estimator Following Equation (7), with a correlation coefficient $c = 1$ as explained in the previous work section, we obtain the *control-variate* estimator

$$S_N^{\text{CV}} = U - \frac{1}{N}\sum_{n=1}^{N} \frac{R(\omega_n)\, L(\omega_n)\, (1 - V(\omega_n))}{p(\omega_n)}, \quad (11)$$

which uses an analytic expression for the unshadowed illumination U and a stochastic estimator for the additive error term.

3.2 Split Formulation of the Shadowed Illumination

Following Equation (4), we formulate direct illumination as

$$S = U \times W, \quad (12)$$

$$W = \frac{S}{U}, \quad (13)$$

where W is the *illumination-weighted shadow*. Intuitively, it is the average visibility of the rays weighted by their contribution to the illumination integral. This is shown more clearly if we write

$$W = \frac{\int_\Omega R(\omega)\, L(\omega)\, V(\omega)\, d\omega}{\int_\Omega R(\omega)\, L(\omega)\, d\omega} = \int_\Omega \frac{R(\omega)\, L(\omega)}{\int_\Omega R(\omega)\, L(\omega)\, d\omega}\, V(\omega)\, d\omega, \quad (14)$$

which emphasizes that $RL/\int RL$ acts as a normalized filter over the binary visibility values V. As a result, the values of the weighted-shadow image are always distributed in $[0, 1]$ and the more a ray contributes to the illumination — i.e. it has both high BRDF and light values — the more it is accounted for in the weighted-shadow image.

Interpretation of the illumination-weighted shadow The value W can be seen as *the probability that a ray sampled perfectly from the normalized product of the BRDF and the light is unshadowed.*

3.3 Ratio Estimator of the Shadowed Illumination

The formulation of Equation (12) yields our *ratio estimator*:

$$S_N^{\text{ratio}} = U \times \frac{S_N^{\text{sto}}}{U_N^{\text{sto}}}, \quad (15)$$

$$S_N^{\text{sto}} = \frac{1}{N}\sum_{n=1}^{N} \frac{R(\omega_n)\, L(\omega_n)\, V(\omega_n)}{p(\omega_n)},$$

$$U_N^{\text{sto}} = \frac{1}{N}\sum_{n=1}^{N} \frac{R(\omega_n)\, L(\omega_n)}{p(\omega_n)},$$

where S_N^{sto} and U_N^{sto} are stochastic estimators of the shadowed and unshadowed illumination that are computed with the same samples ω_n generated from PDF p, i.e. they use the same random numbers. It is important that the $R(\omega_n)\, L(\omega_n)/p(\omega_n)$ values are the same in S_N^{sto} and U_N^{sto} to ensure that the result of their division is a *normalized average* of visibility values $V(\omega_n)$ that remains in $[0, 1]$.

Note that any other sampling strategy can be used to evaluate S_N^{sto} and U_N^{sto} as long as they use the same samples. In our implementation, we use Multiple Importance Sampling (MIS) to compute all of the estimators with lower variance, i.e. we generate samples from both the BRDF and the light and we weight them with the balance heuristic [Veach and Guibas 1995].

The ratio estimator is biased The expectation of the ratio estimator is not the shadowed illumination. Indeed, for a *fixed number of samples* N, the expectation and the fraction do not commute:

$$\mathbb{E}\left[\frac{S_N^{\text{sto}}}{U_N^{\text{sto}}}\right] \neq \frac{S}{U} \quad \Rightarrow \quad \mathbb{E}\left[S_N^{\text{ratio}}\right] \neq S, \quad (16)$$

and the analytic U does not cancel out with the expectation of U_N^{sto} such that only S would remain.

The ratio estimator is consistent The ratio estimator converges toward the right result *as the number of samples increases* because the limit and the fraction commute:

$$\lim_{N\to\infty} \frac{S_N^{\text{sto}}}{U_N^{\text{sto}}} = \frac{S}{U} \quad \Rightarrow \quad \lim_{N\to\infty} S_N^{\text{ratio}} = U \times \frac{S}{U} = S, \qquad (17)$$

and the analytic U effectively cancels out with the limit of U_N^{sto} such that only S remains.

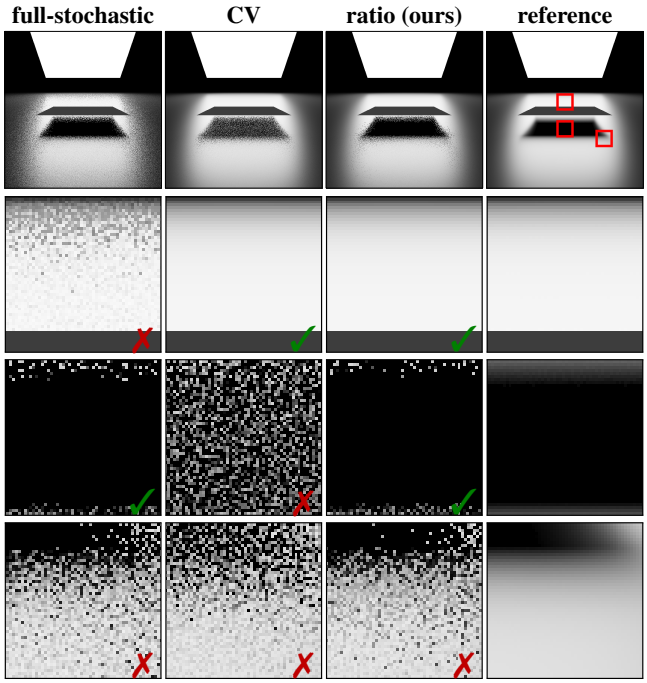

Figure 3: Variance of the estimators at 4spp. *Note that the control-variate (CV) estimator produces negative values that are clamped to 0 (black pixels). Because of this, the shadow appears gray but its average is effectively 0. Furthermore, in the last row, the control-variate result appears less noisy, but the black pixels actually span more variance in the negative domain.*

The ratio estimator has low variance In our experience, the variance of our ratio estimator is generally lower or similar to the variance of both the full-stochastic and control-variate estimators. However, we could not find a mathematical proof of this. To support this observation, we investigate the problem through numerical experiments. A comparison of the noisy results produced by the different estimators is available in Figures 3 and 6, and our supplemental material provides a numerical convergence analysis with an equivalent 1D problem. Our results show that the full-stochastic estimator is generally good in shadowed regions but noisy in lit regions, while the control-variate estimator has the opposite property. Our ratio estimator is as good as the both estimators in their respective good configurations: i.e. it yields noise-free results in lit regions like the control-variate estimator and does not increase the variance in shadowed regions. In penumbra regions, the three estimators are roughly equivalent.

3.4 Approximate Ratio Estimators

Grayscale approximation of illumination-weighted shadows We must compute the estimator for each RGB channel to be exact. This means storing three RGB images: analytic, stochastic shadowed, and stochastic unshadowed. To save memory and bandwidth, especially in denoising, it is possible to compute and process stochastic shadowed luminance and stochastic unshadowed luminance as before, but only a grayscale weighted-shadow image (Figure 4). The approximation diverges from the exact result if the scene exhibits strong correlation between directions and light-source colors, as in Figure 4. However, the approximate result remains plausible and noise-free, and for many applications superior to using no shadow at all.

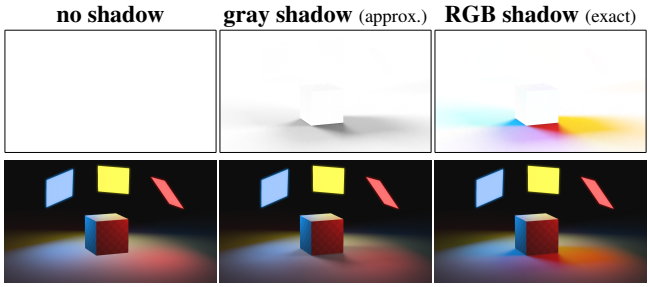

Figure 4: Grayscale approximation of the illumination-weighted shadows. *In order to save memory and bandwidth it is possible to compute a grayscale (rather than RGB) weighted-shadow image.*

Approximate real-time unshadowed illumination So far, we assumed that the *exact* unshadowed illumination was available analytically or through a precomputed representation. However, in many cases, only an *approximate* result is available in real-time. Our method allows us to extend even approximate real-time shading techniques with robust shadows, as shown in Figure 5. For instance, the integral of the BRDF and environmental lighting is hard to precompute for complicated BRDFs. The common real-time approximation for this is to use an environment map prefiltered by an approximate isotropic filter [Karis 2013]. The approximation introduced by this technique is thus the replacement of the exact BRDF R by an approximate BRDF $\tilde{R}$:

$$\int_\Omega \tilde{R}\,L \neq \int_\Omega R\,L, \qquad (18)$$

which is what we desire to obtain a sharp and clean real-time result. However, for the stochastic computation of the weighted shadow, it is simpler to use the exact material R. Indeed, we usually know how to importance sample the exact R but we might not know how to sample the approximate $\tilde{R}$ that has been used to prefilter the environment map, especially if the prefiltering has been further approximated and optimized [Manson and Sloan 2016]. In this case, the denominator of the weighted shadow no longer cancels with the unshadowed illumination in our formulation:

$$\int_\Omega \tilde{R}\,L \times \frac{\int_\Omega R\,L\,V}{\int_\Omega R\,L} \neq \int_\Omega R\,L\,V. \qquad (19)$$

Fortunately, this approximation is visually plausible and artifact-free. Indeed, the weighted shadow $\int_\Omega R\,L\,V / \int_\Omega R\,L$ is a value in $[0,1]$ that modulates the approximate unshadowed illumination $\int_\Omega \tilde{R}\,L$. What we obtain in Figure 5 is thus the approximate unshadowed illumination modulated such that the exact shadow appears. This result is plausible if the real-time unshadowed illumination approximation is plausible.

3.5 Denoising the Ratio Estimator

We now explain how to apply our ratio estimator with a generic **denoise**(). In Section 4, we propose a novel denoiser for shadows.

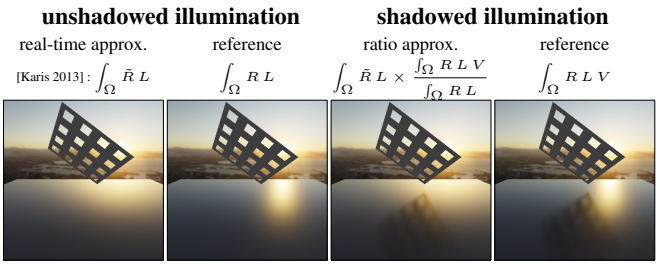

Figure 5: Approximate real-time unshadowed illumination. *We use our formulation to incorporate the exact shadow with a real-time approximation of the unshadowed illumination.*

Computations We compute three images: the *analytic unshadowed illumination* image U, the stochastic *shadowed illumination* S_N, and the stochastic *unshadowed illumination* U_N:

$$U = \int_\Omega R(\omega)\, L(\omega)\, d\omega, \tag{20}$$

$$S_N = \sum_{n=1}^{N} \frac{R(\omega_n)\, L(\omega_n)\, V(\omega_n)}{p(\omega_n)}, \tag{21}$$

$$U_N = \sum_{n=1}^{N} \frac{R(\omega_n)\, L(\omega_n)}{p(\omega_n)}, \tag{22}$$

The analytic image U can be computed using an exact solution if it is available or with an approximate real-time technique otherwise, as explained in Section 3.4. We obtain the stochastic *illumination-weighted shadow* W_N with the division

$$W_N = \frac{S_N}{U_N}, \tag{23}$$

and our stochastic (noisy) result is the product

$$\textbf{result} = U \times W_N. \tag{24}$$

Denoising before dividing An important point is that one should denoise the stochastic shadowed and unshadowed illuminations *before* computing their division:

$$\textbf{denoised result} = U \times \frac{\textbf{denoise}[S_N]}{\textbf{denoise}[U_N]}, \tag{25}$$

instead of denoising W_N directly.

Mathematical justification An alternative could be to denoise the ratio instead, i.e. performing denoising *after* the division. To get an intuition, let us consider denoising a set of pixels that have exactly the same BRDF, light, and visibility, and should therefore evaluate to the same result. In this case, the denoising operation between these pixels should not introduce bias and we expect convergence towards the exact result as the number of denoised pixels increases. Similar to Equation (16), denoising after the division does not make the denoising operation converge toward the right result because the estimator is *biased*:

$$\textbf{denoise}\left[\frac{S_N}{U_N}\right] \not\longrightarrow \frac{S}{U}. \tag{26}$$

Similar to Equation (17), denoising before and dividing after makes the denoising operation converge toward the right result because the estimator is *consistent*:

$$\frac{\textbf{denoise}[S_N]}{\textbf{denoise}[U_N]} \longrightarrow \frac{S}{U}. \tag{27}$$

Practical justification Besides being the right mathematical approach, denoising before dividing (as in Equation (27)) also presents the advantage of being robust to pixels with low denominators or undefined values. Let us consider the case where the illumination weight evaluates to 0, i.e.

$$S_N = U_N = 0 \implies W_N = \frac{S_N}{U_N} = \frac{0}{0} = \text{NaN}. \tag{28}$$

If we denoised the weighted-shadow values $W_N = S_N/U_N$, this NaN value would propagate and cause visual artifacts. In contrast, by denoising S_N and U_N before dividing as we suggest, they get mixed with their neighboring pixel values and become positive (as long as they are positive for at least one pixel in the neighborhood) and the division is well-defined. In general, with this approach, pixels with small denominators U_N do not contribute to themselves or their neighborhoods, which is the expected behavior. Indeed, the weighted-shadow is by definition a weighted sum of shadow values. The smaller the weights, the less relevant the shadow values.

Robustness for denoising Figures 6 and 7 apply a naive bilateral filter over the different estimators. This simple denoising model demonstrates the robustness of the estimators against aggressive filtering; we later show a more sophisticated denoiser. Naive denoising works well for all the estimators where neighboring pixels expect similar results. However, if the image contains high-frequency shading details, the methods behave differently. Denoising the full-stochastic estimator blurs these details. Denoising the control-variate estimator introduces visual artifacts because some pixels might have negative values of high magnitude. These negative values can be spread safely to similar neighboring pixels but they should not be spread to different pixels that do not expect so much variance in the negative domain. The consequence in this example is that denoising the control-variate estimator makes the shadow disappear. In contrast, denoising our weighted-shadow image, as explained above, only makes the shadow appear blurrier and shading details are preserved.

Using better denoisers Recent denoisers, such as that of Schied et al. [2017], produce significantly less overblurring of shading details and achieve better results compared to our simple bilateral filter in Figures 6 and 7. We did not choose such a state-of-the-art denoiser for our comparisons because we wanted to validate the robustness of our formulation against rough denoising techniques, i.e. in the *worst case*.

Furthermore, the objective of recent research focusing on denoising is to produce the best possible **denoise** $[S_N]$ as a final result. Our denoising formulation $U \times \frac{\textbf{denoise}[S_N]}{\textbf{denoise}[U_N]}$ works on top of that. It uses the information provided by the analytic U to prevent shading noise and shading overblur. Thanks to this, our formulation ensures that, for any given **denoise**() function, our result is always better. Hence, even the most efficient *direct-illumination denoisers can only be improved by our formulation*. Furthermore, our formulation with an existing denoiser comes at negligible cost because the analytic computation of U is usually cheap and the denoising of U_N is exactly the same as S_N, so most computations can be factorized.

In summary, we directly benefit from existing (and future) efficient direct-illumination denoisers and, reciprocally, our formulation provides an improved way to use them that is simple to implement and comes at negligible overhead.

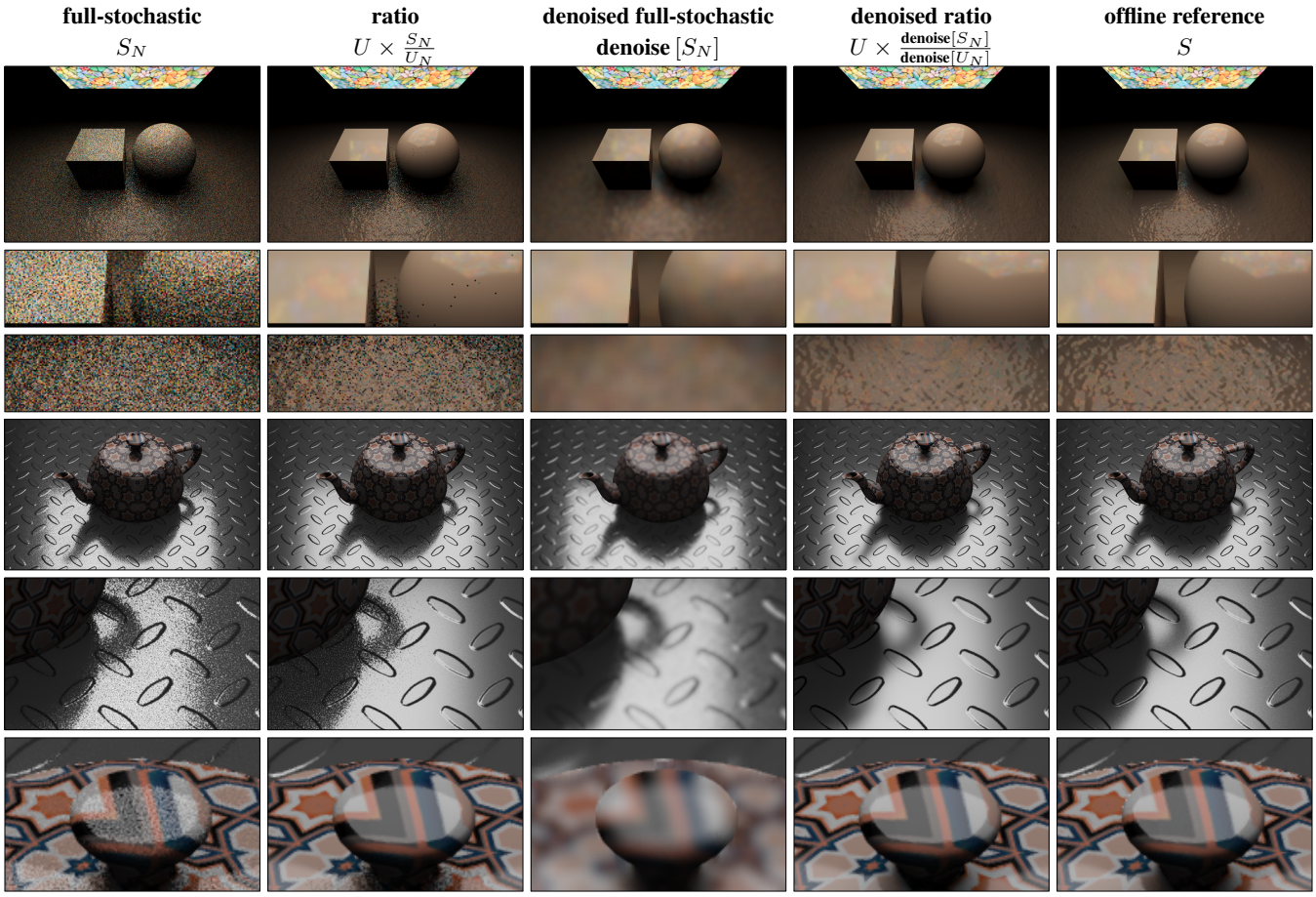

Figure 6: Our ratio estimator. *All results are rendered using 4 shadow rays per pixel and the denoiser is a bilateral filter. In contrast to a classic full-stochastic estimator, our ratio estimator renders noise-free results in unshadowed regions of the image. Furthermore, the denoising formulation of the ratio estimator only overblurs shadows and shading details remain sharp.*

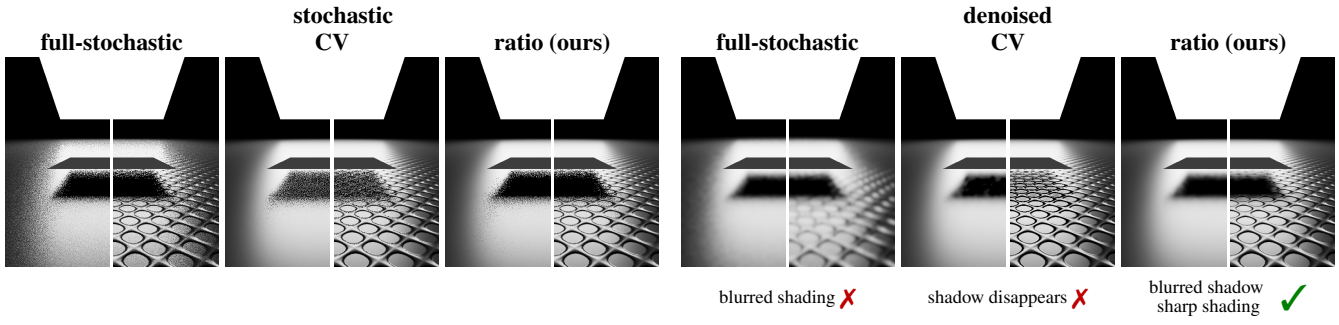

Figure 7: Robustness for denoising. *Estimators denoised by a $\sigma = 4$-pixel Gaussian. Denoising the estimators on a smooth surface produces acceptable results. With this normal map, denoising the full-stochastic estimator blurs the shading details, denoising the control-variate estimator removes the shadow because it spreads negative values from the error correction terms, and denoising the ratio estimator produces the best result.*

4 Our Real-Time Algorithm

In this section, we describe our real-time implementation dedicated to our primary motivating case of efficiently shadowing area lights.

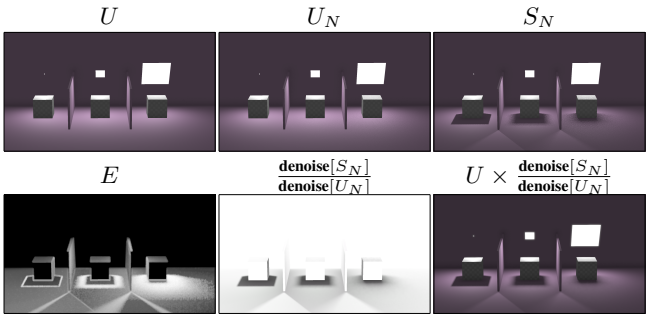

Figure 8: Our real-time algorithm: computations, intermediate terms and result. *The figures are at 1920×1080 each; zoom in to see stochastic sampling noise.*

4.1 Main algorithm

Our main algorithm operates in five full-screen passes per frame, as illustrated in Figure 8:

1. Compute the noise estimate E from $W_N = \frac{S_N}{U_N}$
2. Denoise E itself via a 3×3 box filter
3. Horizontally joint-bilateral filter S_N and U_N simultaneously
4. Vertically joint-bilateral filter S_N and U_N simultaneously and produce a single output: final image $U \times \frac{\textbf{denoise}[S_N]}{\textbf{denoise}[U_N]}$
5. Temporally integrate the final image via reverse reprojection

Filter passes 3 and 4 use an underlying Gaussian kernel with the bilateral weights driven by plane differences between the G-buffer position and normal at the center of the kernel and at the tap location. This is a standard denoising practice.

We set the standard deviation of the Gaussian kernel proportional to the noise estimate E. This allows the filter to grow in noisy regions produced by large penumbrae without overblurring where shadows harden, near surfaces and under small light sources. It also means that our filter incurs almost no cost outside of the penumbrae. These are innovations over previous real-time denoising approaches.

We take advantage of the need to denoise two signals by computing them in the same pass with multiple render targets, which amortizes the cost of the G-buffer memory access for bilateral filter coefficients. We also directly write the final image in the vertical pass without explicitly computing the illumination-weighted shadow image $W_N = S_N/U_N$, to reduce output bandwidth requirements. The visualizations shown in our result figures are for exposition and do not exist in memory in the actual implementation.

The final temporal integration is the game industry's standard practice for antialiasing [Salvi 2015]. It reduces all forms of image aliasing, including any residual low frequency noise from shadow undersampling.

4.2 Noise Estimate

Total-variation noise estimation The choice of estimator E is key. At first, one might consider statistical *variance* of each neighborhood, but this is actually a poor choice because it ignores ordering. For example, the variance of a comb function (impulse train) and a step function can be the same, but in our application the comb is probably a noisy shadow that should be blurred and the step function a well-sampled hard-shadow edge that should be preserved. So, we instead designed an estimator that considers pixel ordering.

We compute E at each pixel x from the *total variation* of the color *gradient* of $W_N = \frac{S_N}{U_N}$ estimated by n path integrals across a set of uniformly-angled lines $\{\ell\}$ through x:

$$E[x] = \frac{1}{n} \sum_{i}^{n} \int_{\ell_i} \left| \frac{||d^2 W_N[\ell_i(t)]||}{dt^2} \right| dt. \tag{29}$$

When both S_N and U_N are close to zero, arbitrarily let $W_N = 1$.

$E[x]$ is a robust measure for the amount of "wiggle" in the 2D neighborhood. We use an asterisk shape of $n = 4$ lines with a random per-pixel offset to avoid bias. The idea of total variation as a noise estimate was first introduced by Rudin et al. [1992], and has previously been used for denoising by solving a nonlinear optimization problem to minimize it. We instead use this measure to drive a joint-bilateral filter and work with derivatives. As desired, our method preserves both gradients and is much more efficient because it admits a closed-form implementation within a pixel shader, whereas an iterative solver would create execution divergence.

Because we estimate E from randomly chosen paths, it is slightly noisy itself. Rather than increasing n towards convergence, we simply box-filter the result. See the supplement for a GLSL implementation of the noise estimator.

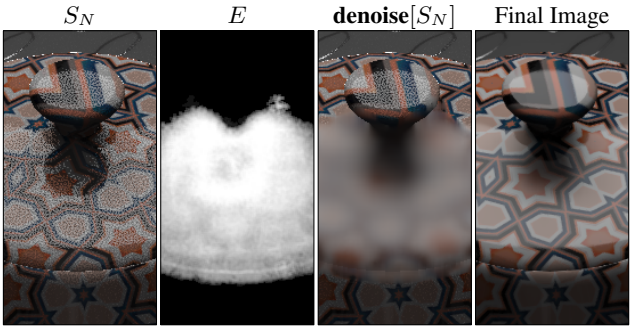

Figure 9: Detail of denoising terms from the teapot scene.

Validation of our noise estimate The definition of E in Equation (29) ensures that only penumbrae and not stochastic sampling of shading or materials are considered noise; after all, we have analytic shading results and this term is only computing shadowing. This point is subtle, but the implications for the difference from full-screen denoising are profound in both potential quality and performance, because the filter cost is dominated by bandwidth required, which is proportional to the radius.

Figure 9 shows detail near the top knob from the teapot scene of the numerator term undergoing denoising. Observe in the center that the E estimate accurately identifies the penumbra. The $\textbf{denoise}[S_N]$ subimage is the denoised numerator. It shows that no blurring (and thus, minimal cost) occurs outside of the penumbra. The umbra on the back of the teapot near the bottom of the figure and the unshadowed highlight remain untouched. Yet, the highlight is not noisy in the denoised weighted shadow or the *final* image Figure 2(e) because that stochastic noise is identical in the denominator $\textbf{denoise}[U_N]$ and thus cancels. See Figure 10 for more examples of accurate noise estimation.

4.3 Real-Time Area Lights

Heitz et al. [2016] give an efficient analytic solution suitable for use as U, for unshadowed direct illumination from polygonal area sources. We render a G-buffer and then compute U, U_N, and S_N in a single deferred shading pass. The terms used to compute the stochastic estimates are three textures that store the shadow ray origins (with start distance offset in the alpha channel), the shadow ray directions (with end distance in the alpha channel), and the radiance from the source along that ray. For a budget of N rays per pixel, these textures are N times larger than full-screen images.

Following the deferred shading pass, we cast all of the shadow rays in parallel. (Our implementation uses OptiX for peak performance; FireRays and Embree are viable alternatives). Then, we add the corresponding radiance contribution at each pixel to S_N if the shadow ray is unoccluded.

The stochastic images U_N and S_N depend on the choice of rays, i.e., points on the light to match with the fixed surface point at each pixel. To compute these, we apply weighted multiple importance sampling to the light surface and the BRDF at each pixel, with a fixed ray budget per light per pixel. For each ray, we choose BRDF (vs. light) sampling with probability proportional to the analytic reflected illumination U. For BRDF samples, we choose glossy (vs. matte) lobe sampling with probability proportional to the glossy reflected illumination divided by the total reflected illumination (again from U). For light samples, we use the solid angle sampling approach of Ureña et al. [2013]. To minimize clumping, we make all of these choices in a low-discrepancy fashion in screen space, so we use Halton sequences with 2, 3, 5, and 7 jittered by a 2D screen-space blue noise distribution as in [Georgiev and Fajardo 2016], for the four pseudo-random values required. See our supplement for the GPU implementation and normalization terms.

5 Results

Figure 8 shows the key terms, including the implicit conditional shadow image, for lights of increasing size casting shadows on cubes, using our area source application. Note the low-discrepancy pattern of the stochastic samples, and the preservation of texture and edge detail since only shadows and not the final image are denoised. The high-noise estimate is correctly tightly restricted to the noisy shadow edge for the left-most cube with the small light source and simultaneously correctly grows to fill the penumbra for the large source.

Figure 11 compares final images from two different state-of-the-art approaches: our area-light shadowing method, and a nearly converged reference image produced from 256 shadow rays per light per pixel. The state of the art for real-time rendering is analytic shading either with no shadows at all (left column) or with point shadows from the center of the light (right column). Point shadows can be produced by shadow maps or ray tracing; we use ray tracing here to avoid shadow map filtering issues. Our result (second column) produced with two shadow rays per light per pixel is closer to the reference image (third column) than these alternatives. The primary remaining error in our result is slight overblurring of some penumbrae due to aggressive denoising. Observe that material details, glossy highlights, and geometric edges are never blurred by our technique.

The first row shows a living room lit by two area sources that simulate indirect light. There is a slightly blue area light on the ceiling simulating reflected light that passed through the windows from the sky, and a slightly orange area light light off to the camera's left simulating indirect light from artificial sources inside the room. The extremely soft shadows throughout the room resemble ambient occlusion. Compare this to the implausible result from point lights.

The second row shows a breakfast room lit by sunlight through the window on the right. There is also a dim environment probe in this room so that the backs of objects are not black. The shadows in our result correctly soften with distance from the window and the shading wraps around the curve of the chair and the pitcher.

The third row shows the Sponza atrium with the Lucy statue in it. Lucy is backlit by a large area source the reflects in the glossy floor. The "reflection" of the statue is actually only a shadow that masks the area light analytic reflection, but due to the contrast may read visually as indirect light. Again, the point light and unshadowed analytic results are poor approximations compared to ours. The varying smoothness of the marble and sharp edges of the light in reflection are preserved because we only denoise the shadow, not the light or material.

Table 1 gives timings for the GPU render passes of this scene. The essential point is that it produces reasonable area light shadows in milliseconds, versus seconds for convergence by stochastic methods. The GPU ray casting dominates our denoising and weighted shadow computations. GPU ray casting is both very fast now and on a strong acceleration trend (especially for shadow rays). As ray cast cost decreases in the future, the net cost the passes in Table 1 for high-quality *area lights* could soon be less than the cost of shadow map generation and filtering for medium-quality *point lights*.

The fourth row shows San Miguel at sunset. The sky is modeled as very large purple area light overhead and the sun is a smaller orange light off the left of the frame. In the video, the sun sets and the sky dims as the camera tracks. The sky shadows are extremely diffuse due to the size of that source, yet are stable and noise free in our result.

Table 1: *Pass timings with a GeForce 1080 at 1920×1080 on Sponza, using 16-bit float precision for all buffers and N shadow rays per pixel.*

Shade Pass	Time	Denoise Pass	Time
Generate rays	$N \times 0.36$ ms	Estimate noise	2.12 ms
OptiX ray cast	$N \times 3.68$ ms	Horizontal bilateral	0.70 ms
Accumulate	$N \times 0.10$ ms	Vertical bilateral	0.70 ms

6 Conclusion and Future Work

We have shown that our formulation, based on the ratio estimator, allows us to combine real-time illumination techniques with ray-traced shadows. We believe that a similar approach could be fruitful for other problems:

Adding shadows from dynamic objects to precomputed irradiance maps Irradiance (a.k.a. light) maps contain precomputed illumination that is typically precomputed via radiosity or path tracing, or computed at runtime with very infrequent updates. As a result, dynamic objects such as animated characters do not correctly shadow irradiance-mapped surfaces.

To correct this, consider a light map rendered with no final-bounce shadowing to be a converged, shadow-less illumination image U, and compute stochastic final-bounce illumination with (S_N) and without (U_N) shadows. Then, compute and apply the denoised weighted shadow image to incorporate dynamic shadowing.

BRDF-aware denoising Separately denoising glossy and matte illumination-weighted shadowing terms with separate noise esti-

mates and thus kernel sizes would avoid overblurring glossy shadows at the cost of doubling the number of terms to be denoised.

Low-variance estimators for multiple-bounce path tracing, photon mapping, or virtual point lights The weighted "shadow" terminology follows from our explanation of the algorithm in terms of direct illumination. But nothing in our derivation precludes its application to *indirect* illumination, or requires the error in the U image to be brighter than the true result. We can apply the technique with U as any noise-free approximation of the illumination, U_N as a stochastic estimator of U, and S_N as a stochastic estimator of the true result, so long as the error has only low-frequency terms that are robust to denoising.

For a path tracer, we can compute U and U_N inexpensively using only direct illumination and let S_N capture a noisy estimate of the net global illumination. In this case, W will primarily contain magnitudes greater than 1 as its role is increasing illumination.

References

ARVO, J. 1995. Applications of irradiance tensors to the simulation of non-lambertian phenomena. In *Proc. ACM SIGGRAPH*, 335–342.

BAUM, D. R., RUSHMEIER, H. E., AND WINGET, J. M. 1989. Improving radiosity solutions through the use of analytically determined form-factors. *Computer Graphics (Proc. SIGGRAPH) 23*, 3, 325–334.

CHEN, M., AND ARVO, J. 2000. A closed-form solution for the irradiance due to linearly-varying luminaires. In *Proc. of Eurographics*, 137–148.

CLARBERG, P., AND AKENINE-MOELLER, T. 2008. Exploiting visibility correlation in direct illumination. In *Proc. of Eurographics*, EGSR '08, 1125–1136.

DE ROUSIERS, C., BOUSSEAU, A., SUBR, K., HOLZSCHUCH, N., AND RAMAMOORTHI, R. 2011. Real-time rough refraction. In *Symposium on Interactive 3D Graphics and Games*, ACM, I3D '11, 111–118 PAGE@7.

DROBOT, M. 2014. Physically based area lights. In *GPU Pro 5*. 67–100.

DUPUY, J., HEITZ, E., AND BELCOUR, L. 2017. A spherical cap preserving parameterization for spherical distributions. *ACM Trans. Graph. 36*, 4 (July), 139:1–139:12.

FAN, S., CHENNEY, S., HU, B., TSUI, K.-W., AND LAI, Y.-C. 2006. Optimizing control variate estimators for rendering. *Computer Graphics Forum 25*, 3, 351–357.

GEORGIEV, I., AND FAJARDO, M. 2016. Blue-noise dithered sampling. In *ACM SIGGRAPH 2016 Talks*, ACM, New York, NY, USA, SIGGRAPH '16, 35:1–35:1.

HARTLEY, H. O. 1989. Unbiased ratio estimators. *Nature 174*, 270–271.

HEITZ, E., AND HILL, S. 2017. Linear-light shading with linearly transformed cosines. In *GPU Zen*.

HEITZ, E., DUPUY, J., HILL, S., AND NEUBELT, D. 2016. Real-time polygonal-light shading with linearly transformed cosines. *ACM Trans. Graph. 35*, 4 (July), 41:1–41:8.

HERY, C., AND VILLEMIN, R. 2013. Physically based lighting at pixar. In *ACM SIGGRAPH Courses: Physically Based Shading in Theory and Practice*.

KARIS, B. 2013. Real shading in unreal engine 4. In *ACM SIGGRAPH Courses: Physically Based Shading in Theory and Practice*.

KAUTZ, J., VÁZQUEZ, P.-P., HEIDRICH, W., AND SEIDEL, H.-P. 2000. Unified approach to prefiltered environment maps. In *Proc. of Eurographics*, 185–196.

LECOCQ, P., DUFAY, A., SOURIMANT, G., AND MARVIE, J. 2016. Accurate analytic approximations for real-time specular area lighting. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 113–120.

MANSON, J., AND SLOAN, P.-P. 2016. Fast filtering of reflection probes. *Comput. Graph. Forum 35*, 4 (July), 119–127.

MARA, M., MCGUIRE, M., BITTERLI, B., AND JAROSZ, W. 2017. An efficient denoising algorithm for global illumination. In *Proc. of High Performance Graphics*, 3:1–3:7.

MICKEY, M. R. 1959. Some finite population unbiased ratio and regression estimators. *Journal of the American Statistical Association 54*, 287, 594–612.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. An efficient representation for irradiance environment maps. ACM, SIGGRAPH '01, 497–500.

RAMAMOORTHI, R., AND HANRAHAN, P. 2002. Frequency space environment map rendering. *ACM Trans. Graph. 21*, 3 (July), 517–526.

RUDIN, L. I., OSHER, S., AND FATEMI, E. 1992. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena 60*, 1, 259 – 268.

SALVI, M., 2015. Anti-aliasing: Are we there yet?, August. SIGGRAPH'15 Open Problems in Real-Time Rendering Course.

SCHIED, C., KAPLANYAN, A., WYMAN, C., PATNEY, A., CHAITANYA, C. R. A., BURGESS, J., LIU, S., DACHSBACHER, C., LEFOHN, A., AND SALVI, M. 2017. Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination. In *Proc. of High Performance Graphics*, 2:1–2:12.

SNYDER, J. M. 1996. Area light sources for real-time graphics. Tech. Rep. MSR-TR-96-11, Microsoft Research.

STACHOWIAK, T. 2015. Stochastic screen-space reflections. In *ACM SIGGRAPH Courses 2015: Advances in Real-Time Rendering in Games*.

UREÑA, C., FAJARDO, M., AND KING, A. 2013. An area-preserving parametrization for spherical rectangles. In *Proceedings of the Eurographics Symposium on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, EGSR '13, 59–66.

VEACH, E., AND GUIBAS, L. J. 1995. Optimally combining sampling techniques for monte carlo rendering. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, 419–428.

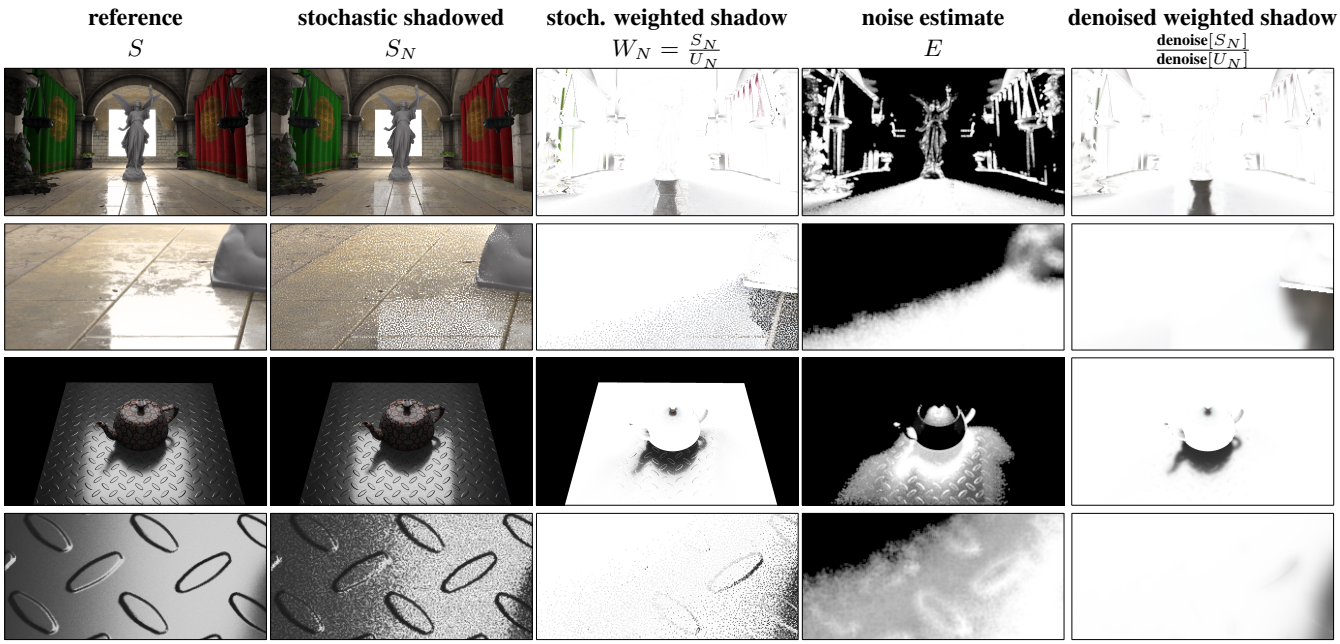

Figure 10: Validation of our noise estimate. *Our noise estimate detects regions where the stochastic weighted-shadow image W_N is noisy and is insensitive if only S_N is noisy. Thanks to this property, we use large denoising kernels only in shadowed regions of the image. The figures are at 1920×1080 each; zoom in to see stochastic sampling noise.*

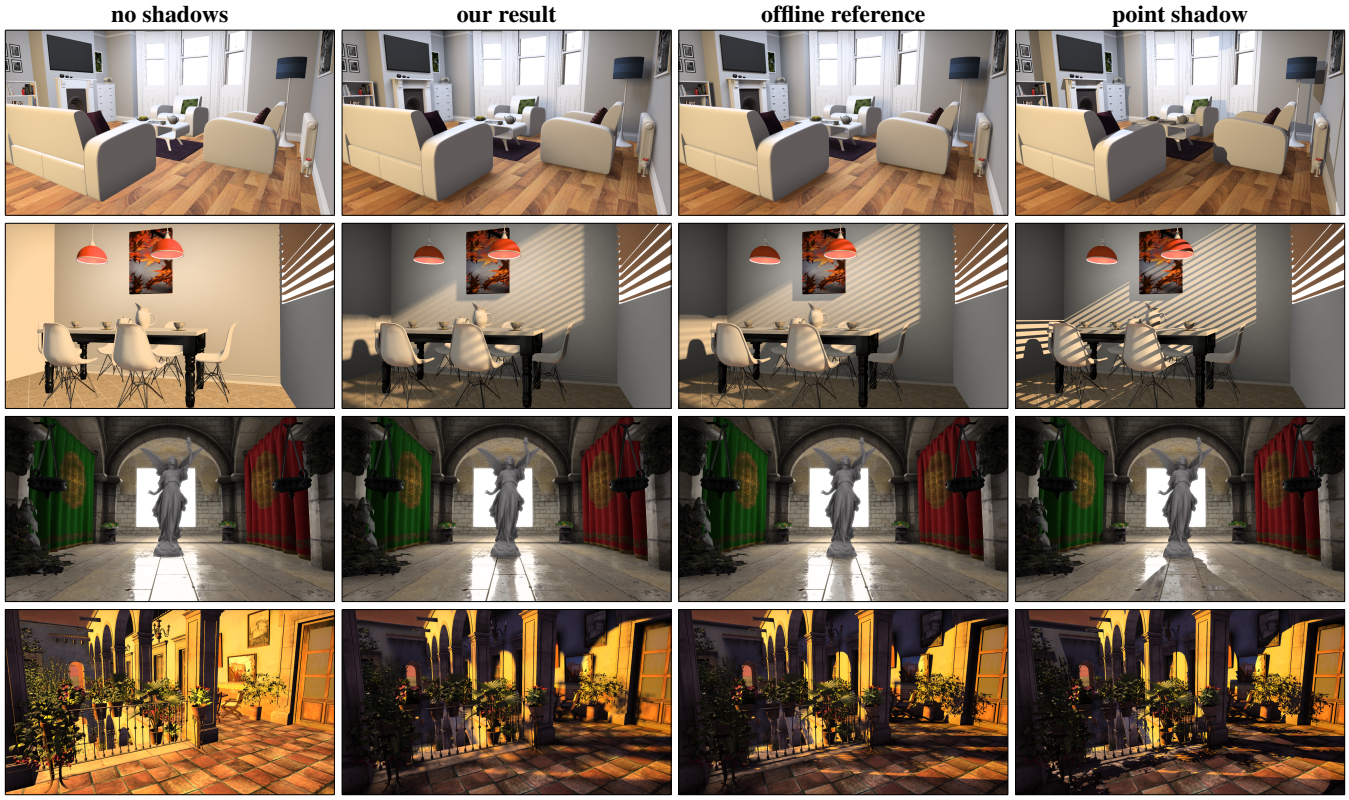

Figure 11: Our real-time algorithm: results with post-processed antialiasing, color grading, and tone mapping. *All methods use analytic shading of area lights. The three undesirable state-of-the-art results: no shadows, offline rendering with 256 shadow rays per light per pixel (high quality but too slow), and point shadows via ray tracing or shadow maps. Our result in the second column provides a good balance between performance and image quality. The figures are at 1920×1080 each.*