Fundamentals of Photorealistic Rendering (Optimizations)

Lecturer: Bochang Moon
GIST

Efficiency of MC estimators

- An efficiency measure for estimators F
  - $e(F) = \frac{1}{V(F)T(F)}$
  - To improve efficiency, we need to reduce the variance and time
  - $V(F)$ may be replaced with MSE(F) for biased estimators

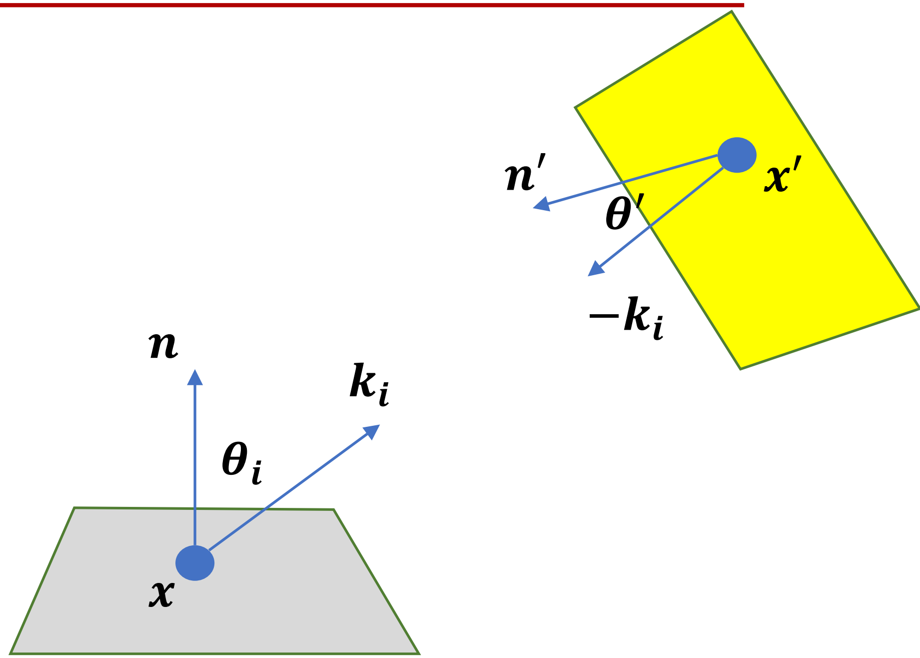
Outline

- Russian Roulette and Splitting
- Importance Sampling
- Multiple Importance Sampling
- Adaptive Rendering (Sampling and Reconstruction)
- Correlated Sampling (Antithetic Variates, Common Random Numbers, Control Variates)
- Gradient-Domain Rendering

Russian Roulette

- e.g. direct lighting
  - $L_s(x, k_o) = \int_{all\ x'} \frac{\rho(k_i,k_o)L_e(x',-k_i)v(x,x')cos\theta_i cos\theta'}{||x-x'||^2} dA'$

$$\approx \frac{1}{N}\sum_{i=1}^{N} \frac{\rho(k_i,k_o)L_e(x',-k_i)v(x,x')cos\theta_i cos\theta' A}{p(x')||x-x'||^2}$$



- Problems
  - Expensive to compute $v(x, x')$
  - Let suppose that there are some directions k_i where the integrand's value is almost 0 due to $\rho(k_i, k_o) \approx 0$
    - In this case, evaluating $v(x, x')$ is not a good idea, as it decreases the efficiency
  - Q. Can we somehow skip these directions while maintaining a correct answer?

Russian Roulette

- Given an estimator F, a new estimator F' with Russian Roulette can be given:
  - $F' = \begin{cases} \frac{F-qc}{1-q} & \xi > q \\ c & otherwise \end{cases}$
  - q: termination probability
  - c is usually chosen as 0

- Consistency check
  - $E(F') = (1-q)\left(\frac{E(F)-qc}{1-q}\right) + qc = E(F)$

- Properties
  - It does not reduce variance, but improves efficiency by skipping unimportant parts

Russian Roulette

- Examples
  - $L_s(k_o) \approx L_e(k_o) + \frac{\rho(k_i,k_o)L_f(k_i)cos\theta_i d\sigma_i}{p(k_i)}$
  - Problem: this is a recursive form, so ray depth can be infinite
  - Can apply the Russian Roulette to path tracing so that the ray depth can be reduced
  - $F' = \begin{cases} \frac{F-qc}{1-q} & \xi > q \\ c & otherwise \end{cases}$
  - Given a termination probability $q = 0.5\ and\ c = 0$, we can terminate the reflection with the probability. In this case, we need to scale the radiance value with $\frac{1}{1-q}$

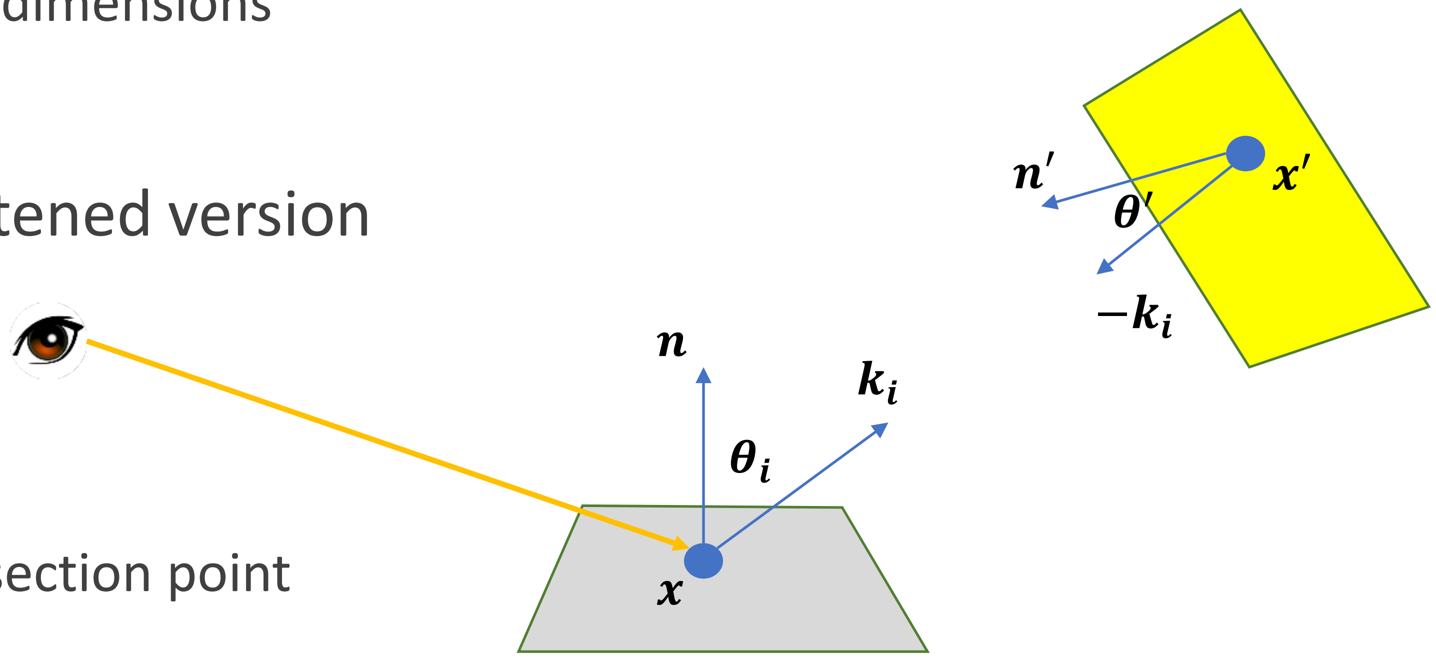
Russian Roulette

- Bad examples
  - Apply this to the camera rays with a termination probability $q = 0.99$
  - In this case,
    - Only trace 1% of the camera rays
    - Most of pixels are black and a few pixels are very bright, although its expectation is still correct
    - The variance of the estimator will be much higher than the original estimator

Splitting

- Splitting is a technique to increase the number of samples for improving the efficiency
  - Allocate more rays to important dimensions

- E.g., direct lighting with a shortened version
  - $\int_A \int_S L_d(x, y, w)dxdydw$
  - A: pixel area
  - S: light area
  - L_d: exitant radiance at the intersection point
  - (x, y): position on image



Splitting

- direct lighting with a shortened version
  - $\int_A \int_S L_d(x, y, w) dxdydw$
- A straightforward approach:
  - Generate N samples $(x_1, y_1, w_1), \ldots (x_N, y_N, w_N)$
  - Evaluate $L_d(x_1, y_1, w_1), \ldots, L_d(x_N, y_N, w_N)$
    - Need to generate N shadow rays
  - Average the radiance values
- Typically need a lot of samples given a large area light or many point lights
  - e.g., N = 100, 200 rays will be used (100 primary rays, 100 shadow rays)
  - Issue: 100 primary rays are often too much for a good antialiasing. Can we focus on

Splitting

- Typically need a lot of samples given a large area light or many point lights
  - e.g., N = 100, 200 rays will be used (100 primary rays, 100 shadow rays)
  - Issue: 100 primary rays are often too many for a good antialiasing.
  - Splitting
    - $\frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{L(x_i, y_i, w_{i,j})}{p(x_i, y_i)p(w_{i,j})}$
    - Are able to use 5 image samples and take 20 light samples per image sample. Total ray # = 5 + 5 x 20 = 105
    - Still use 100 shadow rays for high-quality soft shadows, but the total number of rays is reduced

How to optimize the parameters?

- Tuning the parameters in Russian roulette and splitting is often done empirically
- Automatic control?
  - EARS: efficiency-aware russian roulette and splitting, Rath et al. 2022

Outline

- Russian Roulette and Splitting
- Importance Sampling
- Multiple Importance Sampling
- Adaptive Rendering (Sampling and Reconstruction)
- Correlated Sampling (Antithetic Variates, Common Random Numbers, Control Variates)
- Gradient-Domain Rendering

Importance Sampling

- A variance reduction technique for Monte Carlo estimators
  - Allocate more samples to the important region where the integrand's value is high

- A Monte Carlo estimator:
  - $F_N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p(X_i)}$

- e.g., $L_s(k_o) \approx L_e(k_o) + \frac{\rho(k_i,k_o)L_f(k_i)cos\theta_i d\sigma_i}{p(k_i)}$
  - An intuition is that if we take more samples in terms of k_i that makes $cos\theta_i$ high, we can reduce the variance of the estimator

Importance Sampling

- e.g., Evaluate an integral $\int f(x)dx$
- Note that we can choose an arbitrary pdf, $p(x)$
- What if we choose $p(x) \propto f(x)$ or $p(x) = cf(x)$
  - $c = \frac{1}{\int f(x)dx}$ a constant for normalization
  - $F_N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p(X_i)} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{c}$
  - $V(F_N) = 0$
- Notes
  - In practice, we cannot choose $p(x)$ in this way, but it provides some intuition
  - If we choose $p(x)$ similarly compared to the shape of $f(x)$, we are able to reduce variance

Importance Sampling

- A counterexample:
  - Evaluate an integral $\int f(x)dx$
  - $p(x) = \begin{cases} 99.01 & x \in [0,0.01) \\ 0.01 & x \in [0.01,1] \end{cases}$
  - $f(x) = \begin{cases} 0.01 & x \in [0,0.01) \\ 1.01 & x \in [0.01,1] \end{cases}$
  - $\int f(x)dx = 1$
  - Most of samples will be taken from $[0,0.01)$, and $\frac{f(X_i)}{p(X_i)} \approx 0.0001$ which is far from 1
  - In this case, the variance will increase a lot

- Note
  - In practice, it is easy to apply an important sampling to the rendering, by considering only some terms
  - Taking account for all terms is ideal but this can be very challenging

Outline

- Russian Roulette and Splitting
- Importance Sampling
- Multiple Importance Sampling
- Adaptive Rendering (Sampling and Reconstruction)
- Correlated Sampling (Antithetic Variates, Common Random Numbers, Control Variates)
- Gradient-Domain Rendering

Multiple Importance Sampling

- Evaluate an integral $\int f(x)g(x)dx$
- If we have each important sampling scheme for the functions $f(x)$ and $g(x)$, how can we combine the techniques?
- Multiple important sampling addresses this issue and this will be discussed later

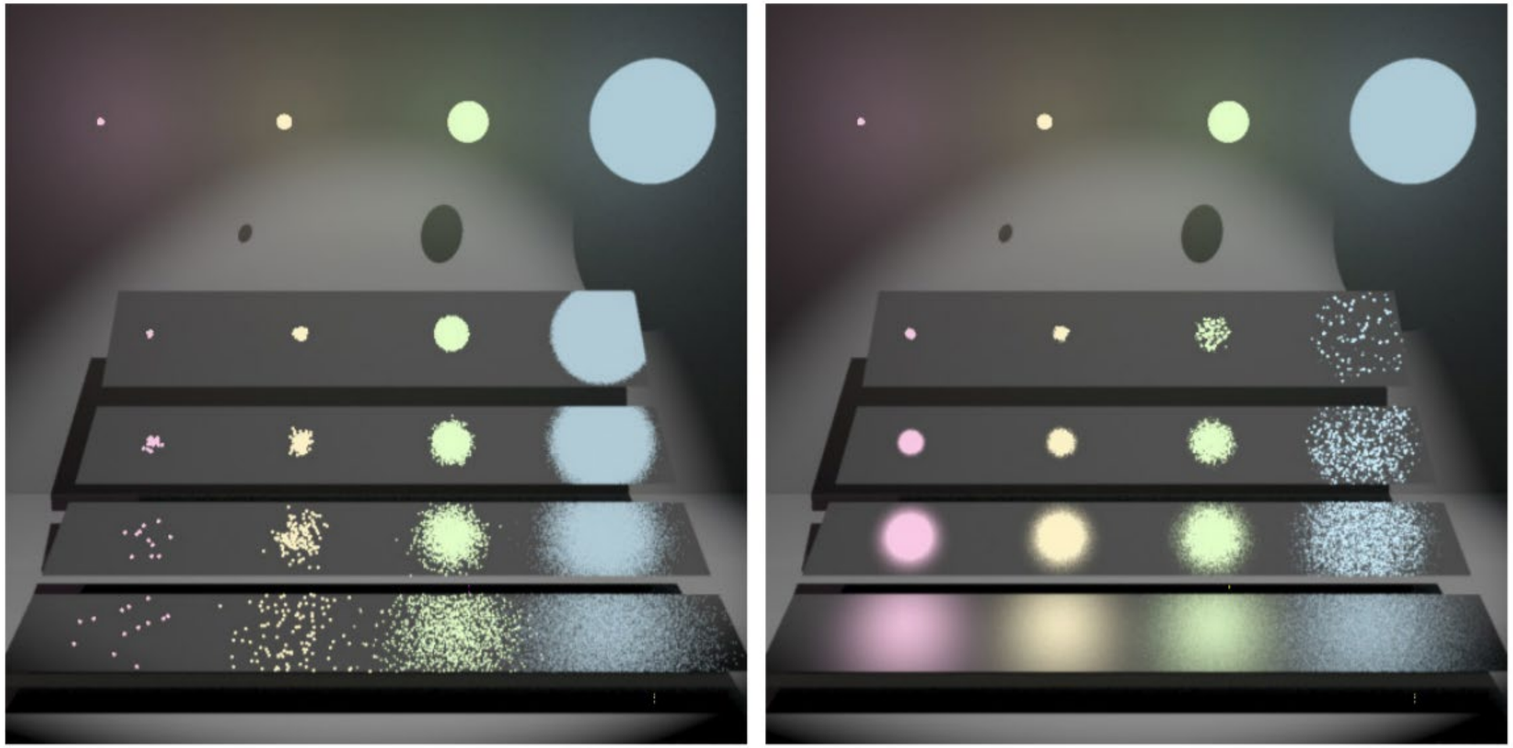
Multiple Importance Sampling

- Proposed by Eric Veach
- Assume we have multiple (more than one) sampling techniques
- Q. How do we combine the techniques?
- Motivation:
  - Light transport integral is complex (most terms are unknown and should be estimated)
  - Designing a sampling technique, which works well for a variety of situations, is difficult

Applications

- Glossy highlights from area light sources
- Common sampling techniques
  - Sampling the light sources
  - Sampling the BRDF
- Light transport for direct lighting
  - $L_s(x, k_o) = \int_{all\ x'} \frac{\rho(k_i, k_o) L_e(x', -k_i) v(x, x') \cos\theta_i \cos\theta'}{||x - x'||^2} dA'$

Applications

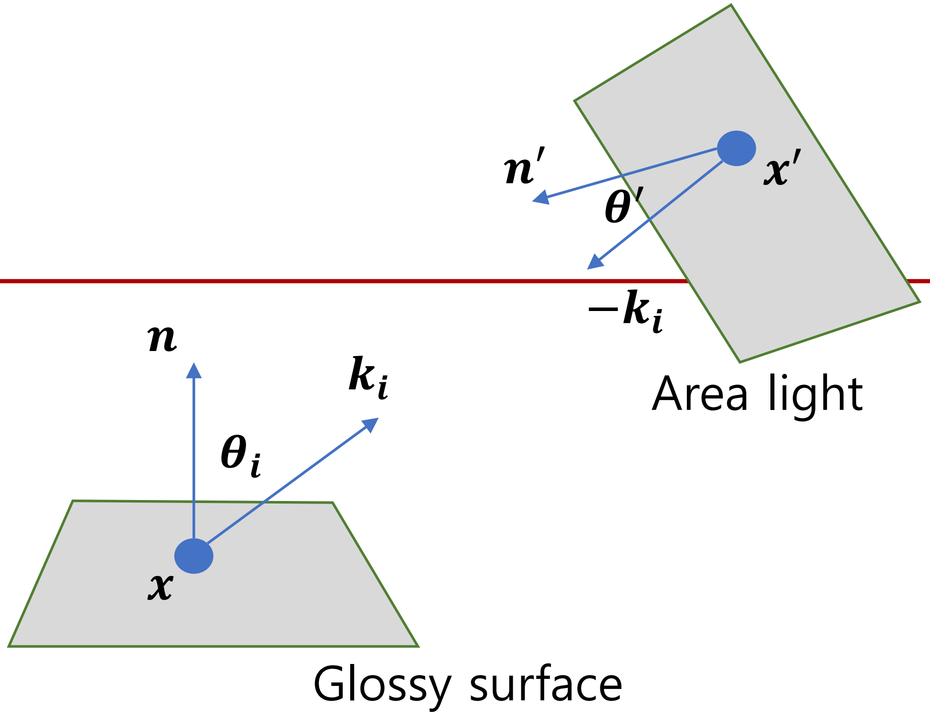


(a) Sampling the BSDF (b) Sampling the light sources

GIST Computer Graphics Laboratory

Images from the Veach's thesis

Applications



- Sampling the light sources
  - $p(x') \propto \frac{L_e(x', -k_i)\cos\theta_i \cos\theta'}{||x-x'||^2}$
- Sampling the BRDF
  - $p(k_i) \propto \rho(k_i, k_o)$
- Light transport for direct lighting
  - $L_s(x, k_o) = \int_{all\ x'} \frac{\rho(k_i, k_o) L_e(x', -k_i) v(x, x') \cos\theta_i \cos\theta'}{||x-x'||^2} dA'$

Multi-Sample Estimator

- A combination strategy to average samples from multiple sampling techniques

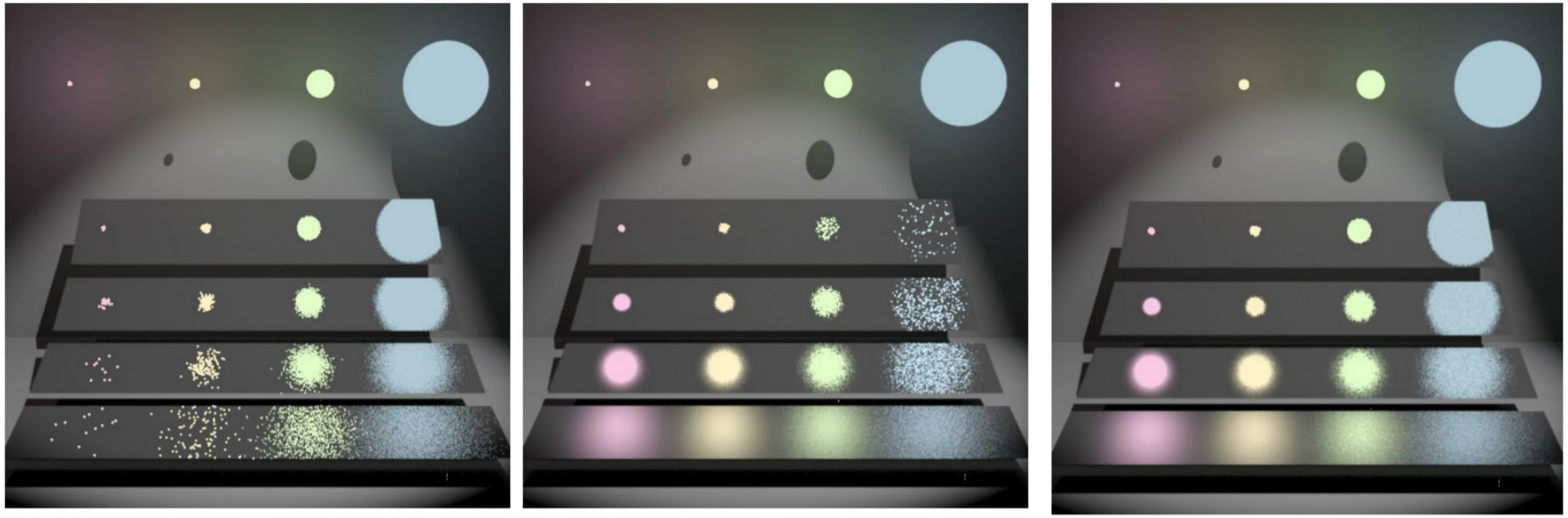
- $\int_{\Omega} f(x) d\mu(x)$
  - $f : \Omega \rightarrow R$

- Samples ($j = 1, \dots, n_i$) from i-th sampling
  - $X_{i,j}$

- Multi-sample estimator
  - $F = \sum_{i=1}^{n} 1/n_i \sum_{j=1}^{n_i} w_i(X_{i,j}) \frac{f(X_{i,j})}{p_i(X_{i,j})}$
  - Conditions for unbiasedness
    - $\sum_{i=1}^{n} w_i(x) = 1$ whenever $f(x) \neq 0$
    - $w_i(x) = 0$ whenever $p_i(x) = 0$
    - See the Veach's thesis for the proof.

Multi-Sample Estimator

- The balance heuristic
  - $\hat{w}_i(x) = \frac{n_i p_i(x)}{\Sigma_\mathrm{k} n_k p_k(x)}$



(a) Sampling the BSDF (b) Sampling the light sources

Images from the Veach's thesis

Outline

- Russian Roulette and Splitting
- Importance Sampling
- Multiple Importance Sampling
- Adaptive Rendering (Sampling and Reconstruction)
- Correlated Sampling (Antithetic Variates, Common Random Numbers, Control Variates)
- Gradient-Domain Rendering

Background

- Monte Carlo estimator
  - $\hat{\theta} = \frac{1}{\mathrm{N}} \Sigma_{\mathrm{i}} \mathrm{f}(\mathrm{X_i})$
  - $var(\hat{\theta}) = \frac{\sigma^2}{\mathrm{N}}$
- Homogeneous error
  - $\mathrm{var}\left(\mathrm{f}(\mathrm{X_i})\right) = \sigma^2$
  - Uniform sampling rate would make sense for the homogeneous error
- Heterogeneous error
  - Variance can vary across local regions
  - Light transport function is typically nonlinear and variance varies across regions

Classical Adaptive Sampling

- Image-space adaptive sampling using per-pixel variance
  - Per-pixel estimator
  - Imagine you have an unknown image-space function per pixel
  - $\widehat{\theta_p} = \frac{1}{\mathrm{N_p}} \Sigma_\mathrm{i} \mathrm{f_p}(\mathrm{X_i})$
  - $var(\widehat{\theta_p}) = \frac{\sigma_p^2}{\mathrm{N_p}}$
  - Assumption:
    - Random samples $\mathrm{f_p}(\mathrm{X_i})$ are iid and share a variance σ_p^2
    - Note that σ_p^2 can vary per pixel p
  - A simple approach
    - Control the number of samples per pixel $\mathrm{N_p}$ to be proportional to σ_p^2
    - High-error regions have high sample counts
  - How do we estimate σ_p^2?
    - $s_p^2 = \frac{1}{N_p - 1} \Sigma_i \left(f_p(X_i) - \overline{f_p}(X) \right)^2$

Classical Adaptive Sampling

- Two-step approach using the sample variance
  - Allocate a small number of samples (equal sample count) per pixel
  - Estimate the sample variance and decide the number of samples per pixel for the next stage
  - Repeat until we use a target sample count

- Potential problem of the simple approach
  - Per-pixel sample variances can be typically noisy unless a large number of samples are used.
  - Sample counts can be noisy and it typically leads to a noisy rendered image
  - A simple trick
    - Apply an image filter to the sample variance and use the smoothed variance

Multi-dimensional Adaptive Sampling

- The classical approach only controls the number of samples in image space
  - Sampling density for the other dimensions (lens, time, …) is still uniform.

- Multi-dimensional sampling
  - The random sample is in a high-dimensional space
  - Maintain a high-dimensional structure (kd-trees) that stores the samples
  - This idea was proposed initially in [Kajiya 86], but it was fully accomplished in [Hachisuka et al. 08]

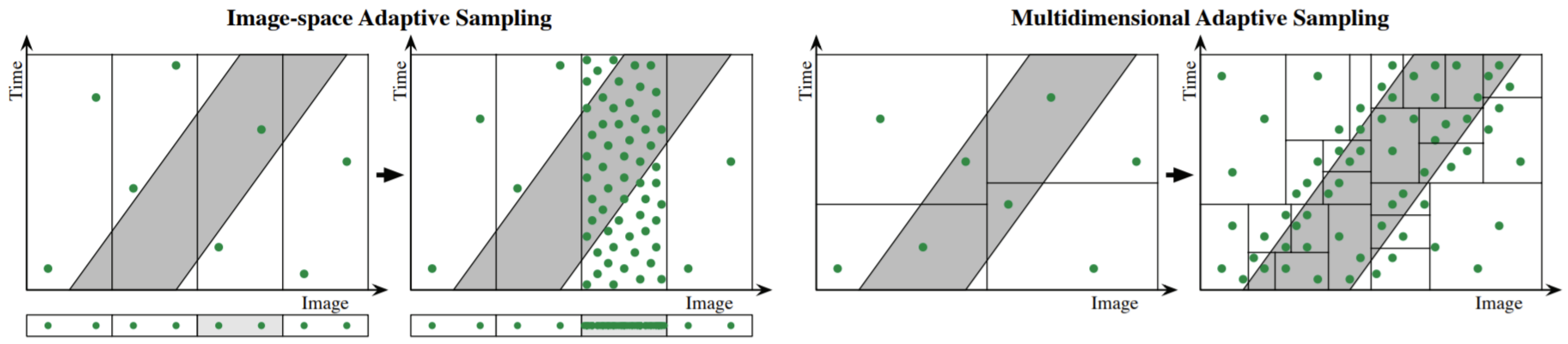


Image from [Hachisuka et al. 08]

Multi-dimensional Adaptive Sampling

- Iteratively allocate more samples to the kd-tree leafs whose variances are high
- Kd-tree leafs are subdivided if its number of samples are higher than a user-defined threshold
- The final image is reconstructed by projecting the sub-regions to the image plane (sub-regions are assumed to be constant)

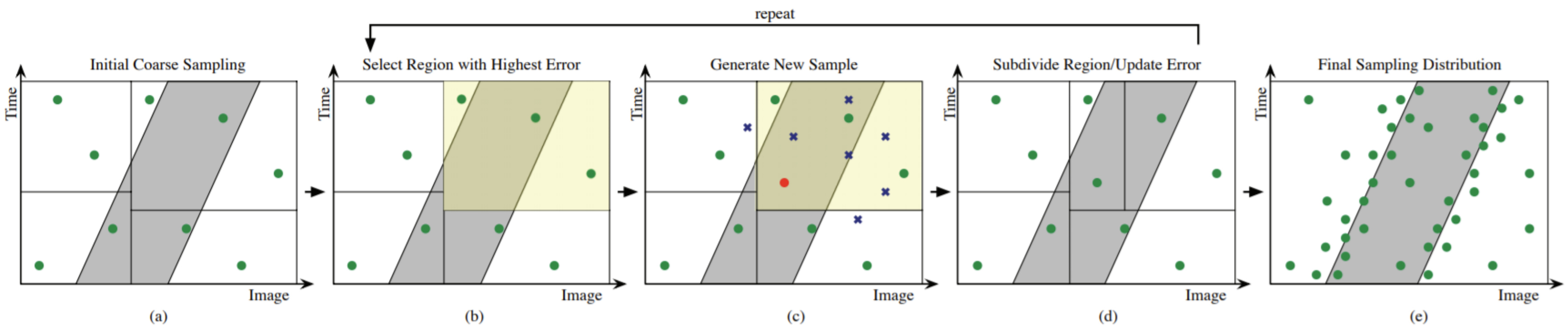
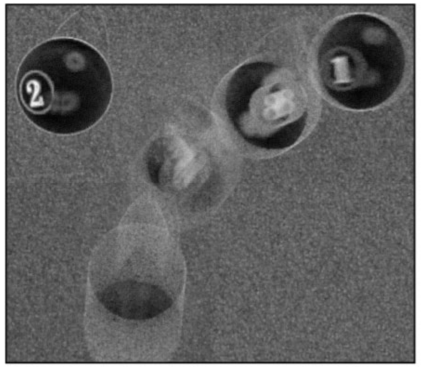


Image from [Hachisuka et al. 08]

Multi-dimensional Adaptive Sampling



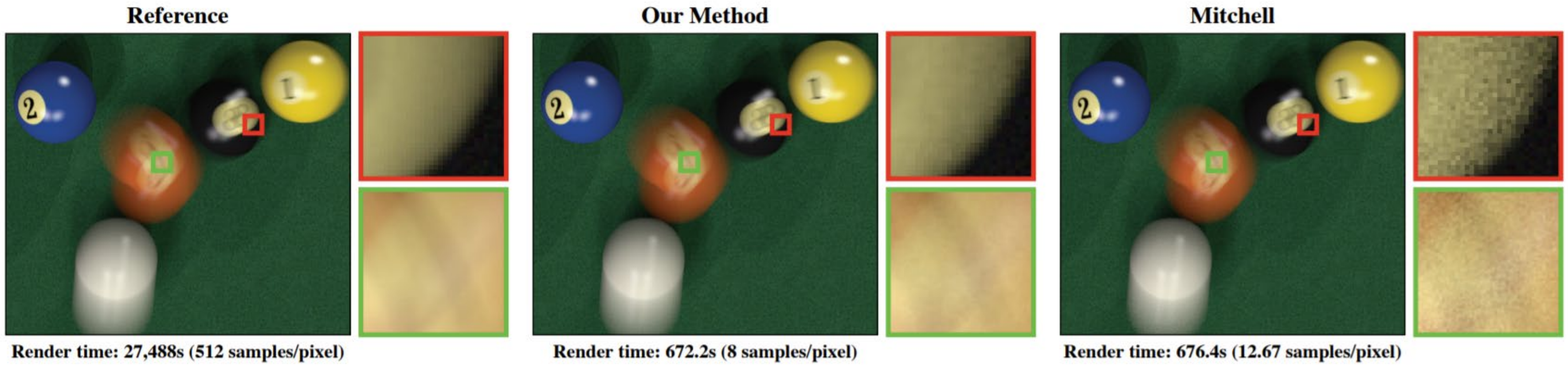


Image from [Hachisuka et al. 08]

Multi-dimensional Adaptive Sampling

- Pros.
  - Very effective for a moderate dimension (direct illumination, motion blur, and depth-of-fields)
- Cons.
  - Sample density becomes very sparse as increasing the dimensionality of samples (e.g., for global illumination)

Adaptive Sampling using Wavelet Space

- Adaptive Wavelet Rendering [Overbeck et al. 09]
- Properties
  - Image-space adaptive sampling
  - More robust than the classical approach
- Wavelet thresholding using variances

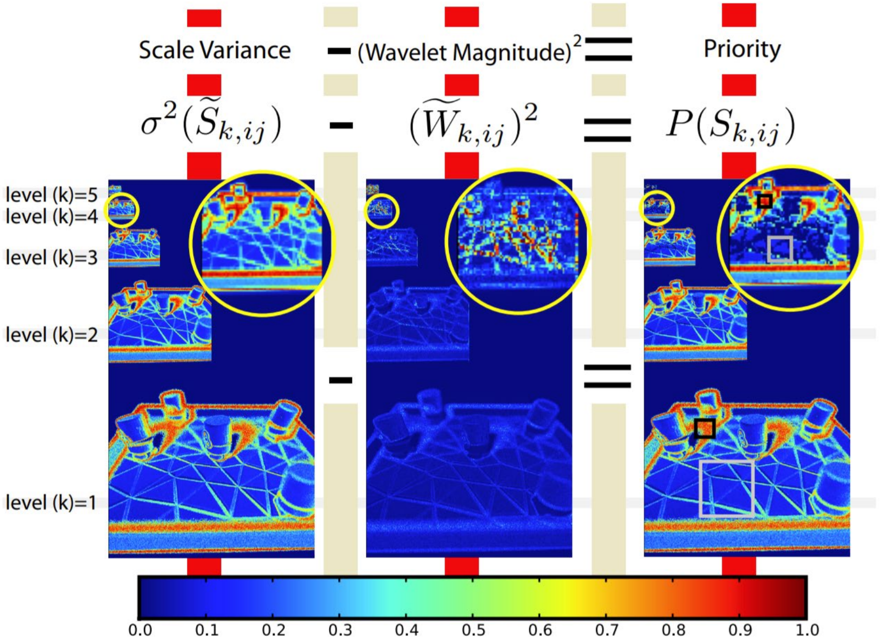


Image from [Overbeck et al. 09]

Adaptive Sampling using Wavelet Space

- Pros.
  - Works well for depth-of-field or motion blur

- Cons.
  - Produces ringing artifacts when GI simulates

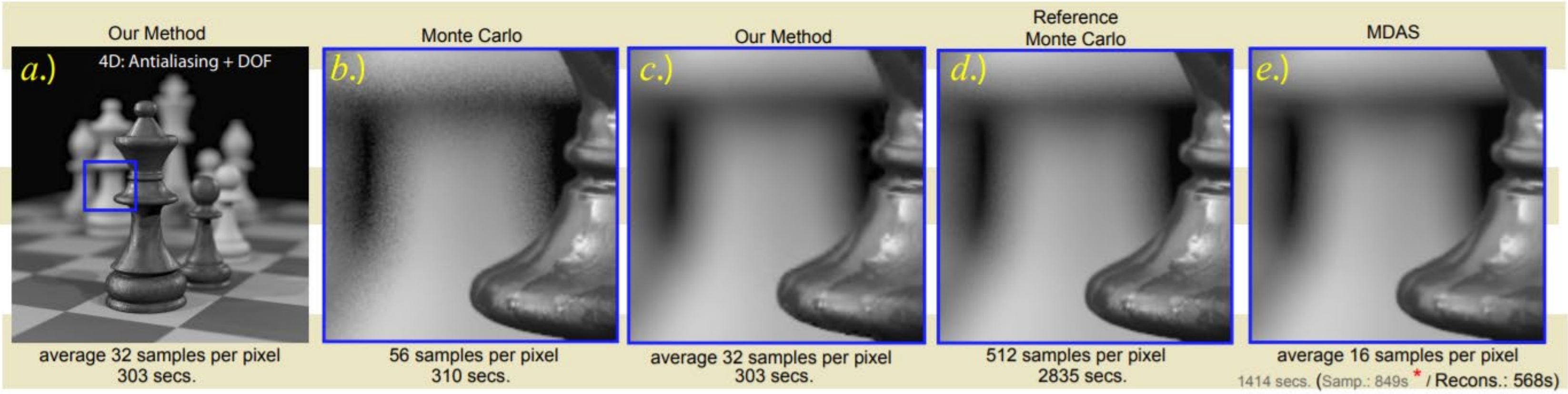


Image from [Overbeck et al. 09]

Classical Image Space Adaptive Rendering

- Controlling sample density using sample variances is not optimal

- Suppose a typical combination:
  - Path tracing + image denoising
  - The variance of the final image (reconstruction variance) can be very different from the sample variance
  - Adaptive sampling should allocate more samples to the regions with high reconstruction errors

Classical Image Space Adaptive Rendering

- Adaptive Sampling and Reconstruction using Greedy Error Minimization
  - [Rousselle et al. 11]
- Control two parameters
  - Sampling rate per pixel
  - Reconstruction parameter per pixel
- Target combination
  - Path tracing + Gaussian filter

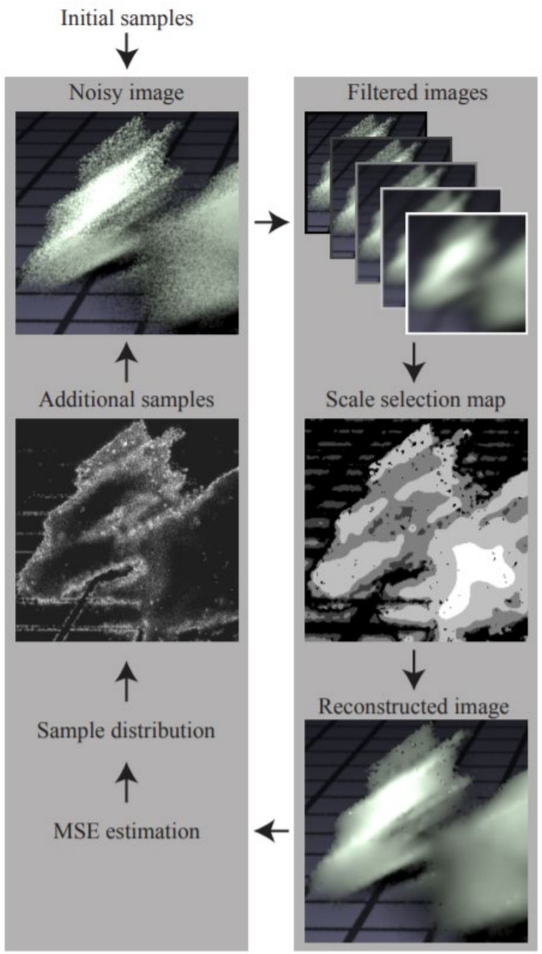


Image from [Rousselle et al. 11]

Classical Image Space Adaptive Rendering

- Adaptive Sampling and Reconstruction using Greedy Error Minimization
  - [Rousselle et al. 11]

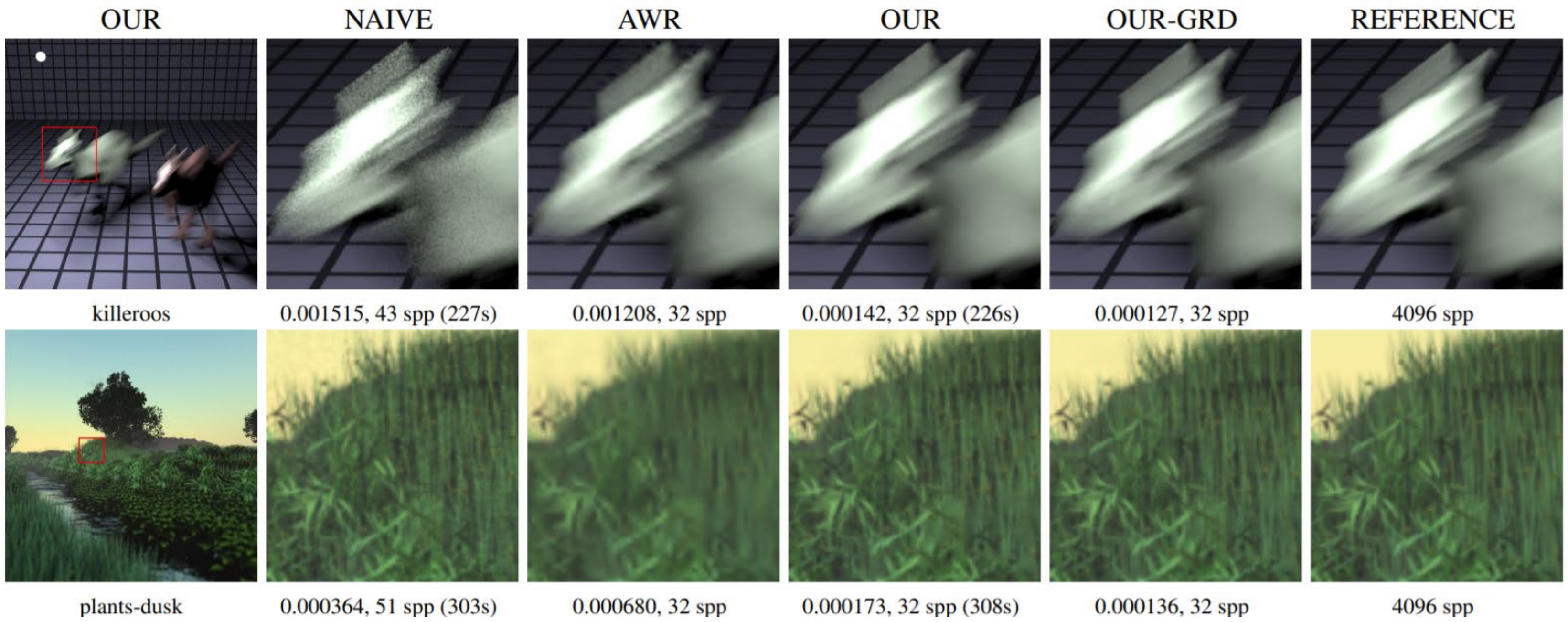


Image from [Rousselle et al. 11]

Classical Image Space Adaptive Rendering

- Adaptive Sampling and Reconstruction using Greedy Error Minimization
  - [Rousselle et al. 11]

- Pros.
  - Very simple and fast
  - Work reasonably well

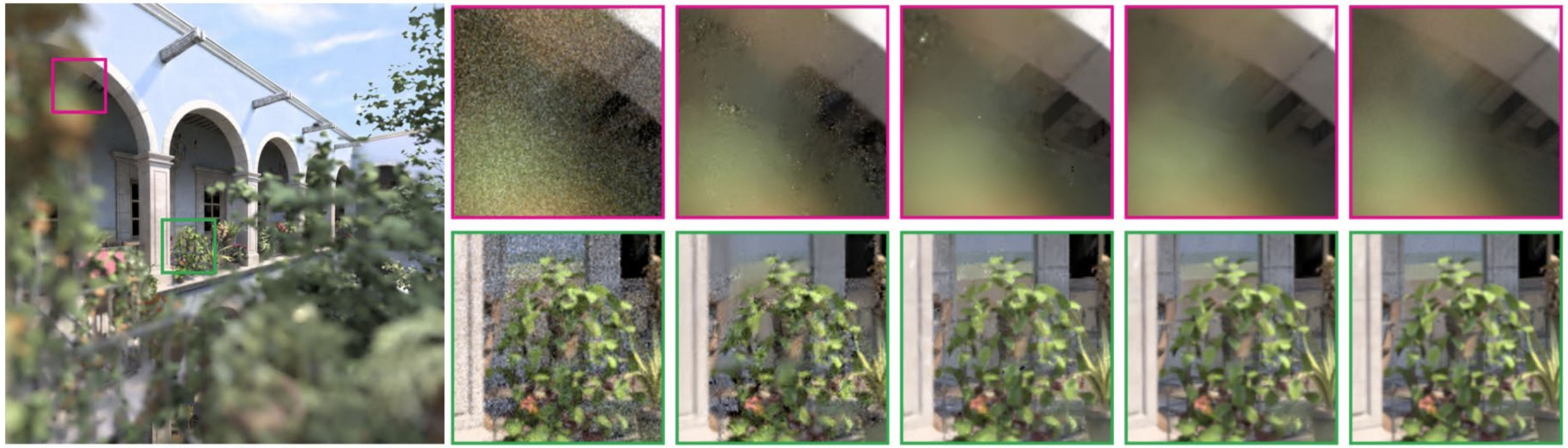
- Cons.
  - Isotropic filter (Gaussian) is not effective for non-linear functions (image edges)

Classical Image Space Adaptive Rendering

- Nonlinear reconstructions
  - Adaptive Rendering with Non-Local Means Filtering [Rousselle et al. 12]
    - Non-local means
  - On Filtering the Noise from the Random Parameters in Monte Carlo Rendering [Sen and Darabi 11]
    - Cross bilateral filtering
  - Adaptive Rendering based on Weighted Local Regression [Moon et al. 14]
    - Local regression
  - Nonlinearly Weighted First-order Regression for Denoising Monte Carlo Renderings [Bitterli et al. 16]
    - Local regression + non-local means

Classical Image Space Adaptive Rendering

- General procedure
  - Allocate sparse samples initially
  - Reconstruct the image using a non-linear filtering
    - Estimate optimal parameters for the reconstruction
    - Estimate MSE (bias^2 + variance) of the image
  - Allocate more samples to the image regions with high MSEs



(a) Ours 115 spp (660 s) rMSE 0.00448 | (b) LD 128 spp (665 s) rMSE 0.06288 | (c) NLM 115 spp (665 s) rMSE 0.01242 | (d) SURE 113 spp (665 s) rMSE 0.01521 | (e) Ours 115 spp (660 s) rMSE 0.00448 | (f) Reference 16K spp

Image from [Moon et al. 14]

Classical Image Space Adaptive Rendering

- Reconstruction performance depends on:
  - Nonlinearity of the chosen filter
  - Robust estimation of errors
  - Rendering-specific features to identify high-frequency image information (edges)
- Adaptive sampling is closely related to a chosen reconstruction method

Neural Image Space Adaptive Rendering

- So far:
  - Reconstruction filters are hand-crafted ones
  - Estimating MSE is often a tedious task (and also quite difficult to be robust)

- Deep learning based approaches
  - Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings [Bako et al. 17]
  - A general weighting can be represented as
    - $\hat{y}_c = \Sigma_i w_i y_i$
    - $w_i > 0$

  - Key advantage is that we don't need to assume a specific functional form for w_i

Neural Image Space Adaptive Rendering

- Deep learning based approaches
  - Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings [Bako et al. 17]
  - A general weighting can be represented as
    - $\hat{y}_c = \Sigma_i w_i y_i$
    - $w_i > 0$

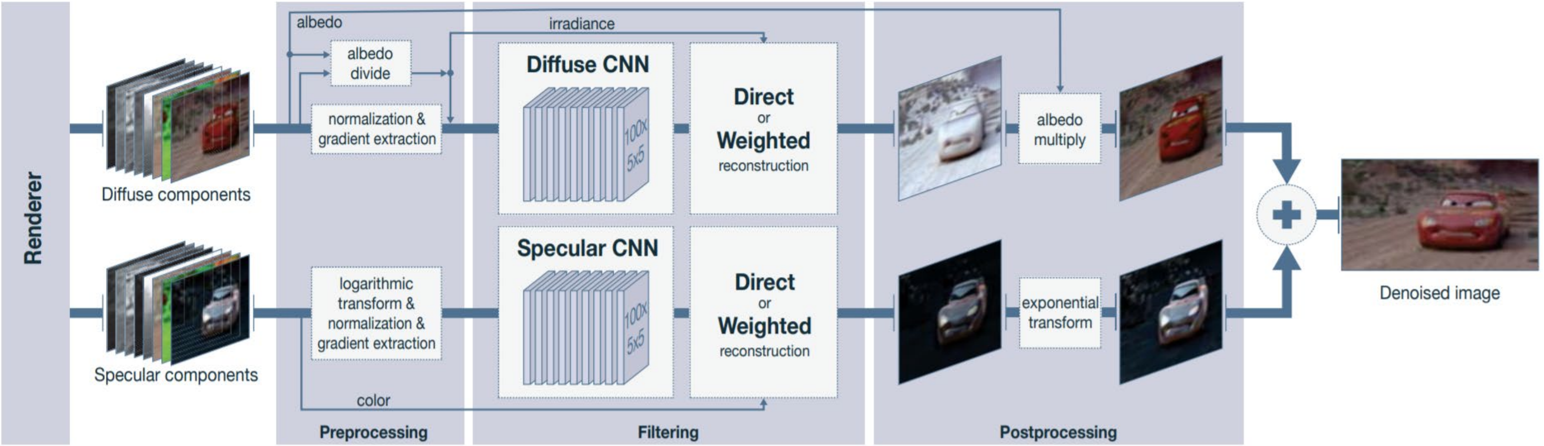


Image from [Bako et al. 17]

Neural Image Space Adaptive Rendering

- Deep learning based approaches
  - Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings [Bako et al. 17]

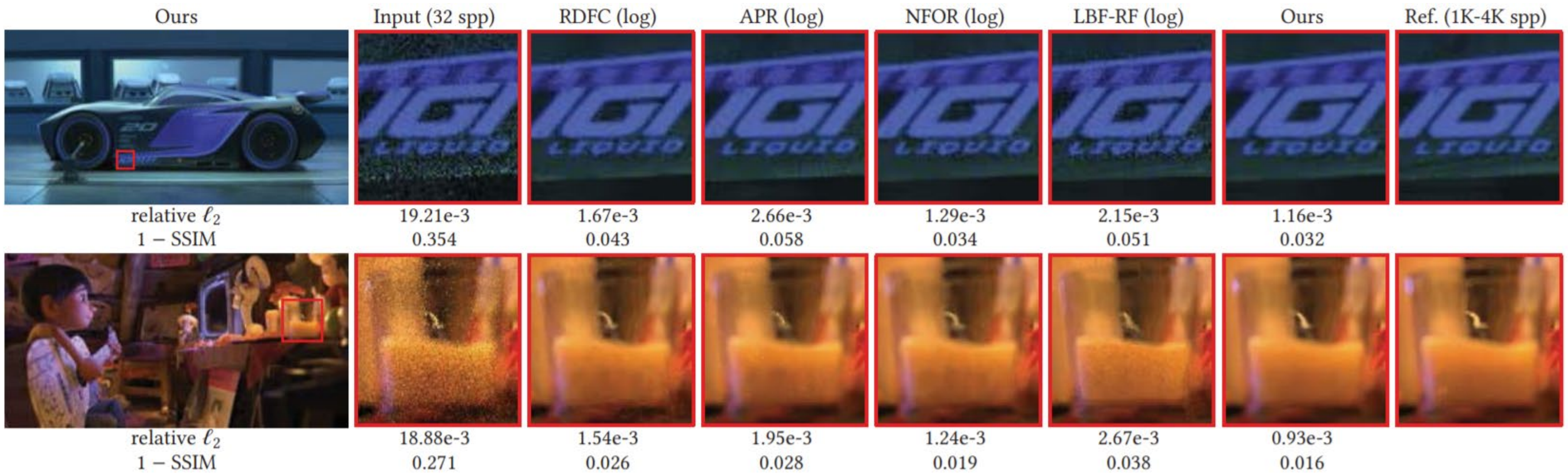


Image from [Bako et al. 17]

Neural Image Space Adaptive Rendering

- Deep learning based approaches
  - Sample-based Monte Carlo Denoising using a Kernel-Splatting Network [Gharbi et al. 19]
  - A general weighting in a sample space can be represented as
    - $\hat{y}_{c,j} = \Sigma_i \Sigma_j w_{i,j} y_{i,j}$

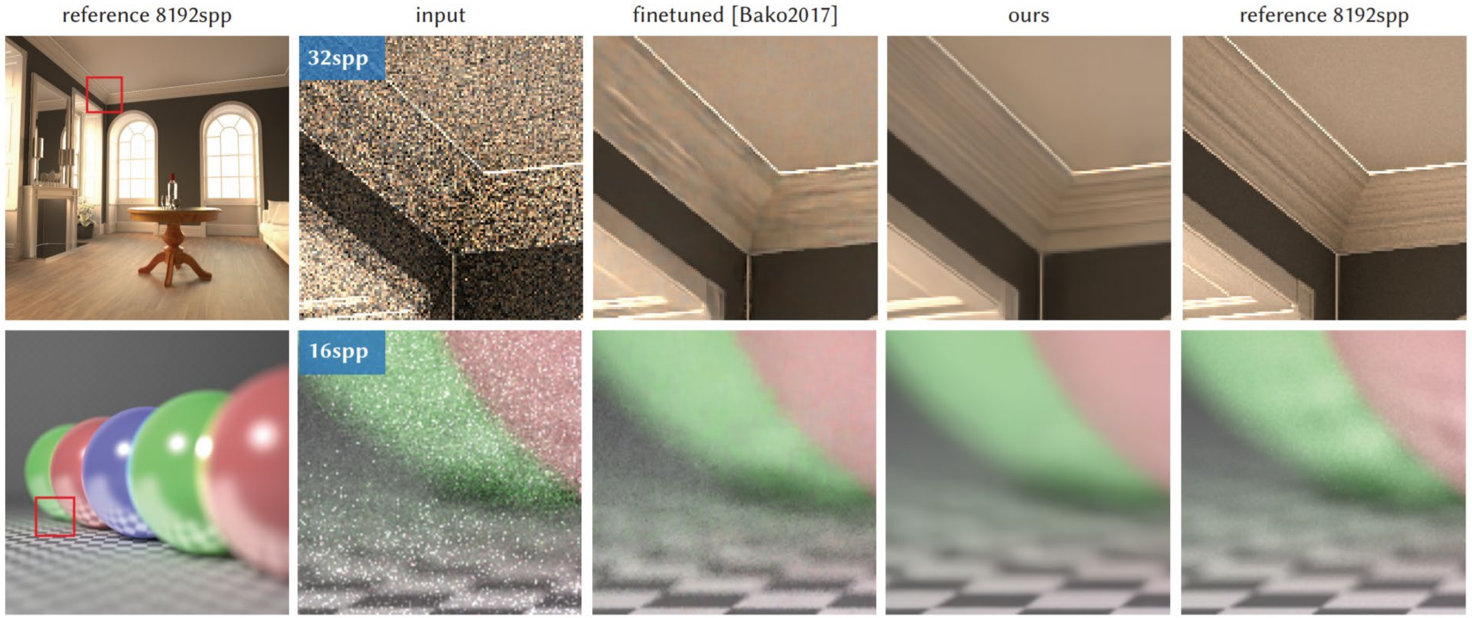


Image from [Gharbi et al. 19]

Recent Optimizations for Adaptive Rendering

- Larger receptive fields with advanced NNs
  - Monte Carlo Denoising via Auxiliary Feature Guided Self-Attention, Yu et al. 2021

- Post-correction via a neural network trained only using a test image (without any pretraining)
  - Self-Supervised Post-Correction for Monte Carlo Denoising, Back et al. 2022

Recent Optimizations for Adaptive Rendering

- Reconstruction bias is not directly controllable by neural image reconstruction
  - i.e., not consistent: Simply adding more samples does not guarantee error reduction

- Making an image denoiser consistent
  - Neural James-Stein Combiner for Unbiased and Biased Renderings, Gu et al. 2022

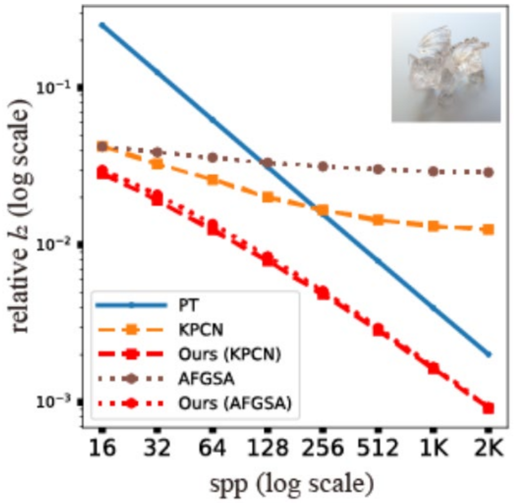


Image from [Gu et al. 2022]

Outline

- Russian Roulette and Splitting
- Importance Sampling
- Multiple Importance Sampling
- Adaptive Rendering (Sampling and Reconstruction)
- Correlated Sampling (Antithetic Variates, Common Random Numbers, Control Variates)
- Gradient-Domain Rendering

Antithetic Variates

- $\theta = \int_\Omega f(x) d\mu(x)$
- Monte Carlo estimator
  - $\hat{\theta} = \frac{1}{N} \Sigma_i f(X_i)$
- Suppose:
  - $pdf(X) = pdf(-X)$
  - i.e., a symmetric pdf of X
- Define a variable
  - $Y_i \equiv \frac{f(X_i) + f(-X_i)}{2}$
  - Y_i is an unbiased estimate of θ, $E[Y] = E[f(X)] = \theta$
  - $\hat{\theta}^{av} = \frac{1}{N} \Sigma_i Y_i$

Antithetic Variates

- (original) Monte Carlo estimator
  - $\hat{\theta} = \frac{1}{N}\Sigma_i f(X_i)$
- New estimator
  - $\hat{\theta}^{av} = \frac{1}{N}\Sigma_i Y_i$
- Suppose I.I.D:
  - $\mathrm{var}\left(f(X_i)\right) = \mathrm{var}\left(f(X_j)\right) = \sigma^2$
- Variance of estimators
  - $var\left(\hat{\theta}\right) = var\left(\frac{1}{N}\Sigma_i f(X_i)\right) = \frac{1}{N^2}\Sigma_i var\left(f(X_i)\right) = \frac{1}{N^2}\Sigma_i \sigma^2 = \frac{\sigma^2}{N}$
  - $var\left(\hat{\theta}^{av}\right) = var\left(\frac{1}{N}\Sigma_i Y_i\right) = \frac{1}{N^2}\Sigma_i var(Y_i)$

Antithetic Variates

- Variance of estimators
  - $var(\hat{\theta}) = var\left(\frac{1}{\mathrm{N}}\Sigma_{\mathrm{i}}\mathrm{f}(\mathrm{X}_{\mathrm{i}})\right) = \frac{1}{\mathrm{N}^2}\Sigma_i var\big(f(X_i)\big) = \frac{1}{N^2}\Sigma_i \sigma^2 = \frac{\sigma^2}{N}$
  - $var(\hat{\theta}^{av}) = var\left(\frac{1}{\mathrm{N}}\Sigma_{\mathrm{i}} Y_i\right) = \frac{1}{\mathrm{N}^2}\Sigma_i var(Y_i)$
  - $var(Y_i) = var\left(\frac{f(X_i)+f(-X_i)}{2}\right) = \frac{1}{4}\left[var\big(f(X_i)\big) + var\big(f(-X_i)\big) + 2cov\big(f(X_i), f(-X_i)\big)\right]$
  - Putting $var(Y_i)$ into $var(\hat{\theta}^{av})$:
  - $var(\hat{\theta}^{av}) = \frac{1}{4N^2}\Sigma_i\{2\sigma^2 + 2cov\big(f(X_i), f(-X_i)\big)\} = \frac{1}{2N^2}\Sigma_i\{\sigma^2 + cov\big(f(X_i), f(-X_i)\big)\}$
  - If there is no correlation
    - $var(\hat{\theta}^{av}) = \frac{\sigma^2}{2N} = \frac{var(\hat{\theta})}{2}$
    - No actual gain here since we use 2N samples instead of N samples
  - What if there is a negative correlation?
    - $var(\hat{\theta}^{av}) < \frac{var(\hat{\theta})}{2}$

Antithetic Variates

- Antithetic variates introduces a negative correlation for monotonically increasing functions
  - $cov\big(f(X_i), f(-X_i)\big) < 0$
  - e.g., linear functions – ideal case

- Properties
  - Very simple to implement it even for high-dimensional cases
  - Some applications in rendering:
    - Direct lighting
    - Pixel estimator in PSS?

Common Random Numbers (CRN)

- Suppose we want to estimate a difference between two functions
  - $\theta_1 = \int_\Omega f_1(x) d\mu(x)$
  - $\theta_2 = \int_\Omega f_2(x) d\mu(x)$
  - $\theta = \theta_1 - \theta_2$

- MC estimator
  - $\hat{\theta} = \frac{1}{\mathrm{N}} \Sigma_{\mathrm{i}} \mathrm{f}_1(\mathrm{X}_{\mathrm{i}}) - \frac{1}{\mathrm{N}} \Sigma_{\mathrm{j}} \mathrm{f}_2(\mathrm{X}_{\mathrm{j}})$

- CRN estimator
  - $\hat{\theta} = \frac{1}{\mathrm{N}} \Sigma_{\mathrm{i}} \mathrm{f}_1(\mathrm{X}_{\mathrm{i}}) - \frac{1}{\mathrm{N}} \Sigma_{\mathrm{i}} \mathrm{f}_2(\mathrm{X}_{\mathrm{i}}) = \frac{1}{N} \Sigma_{\mathrm{i}} \left(f_1(X_i) - f_2(X_i) \right)$

Common Random Numbers (CRN)

- CRN estimator
  - $\hat{\theta} = \frac{1}{\mathrm{N}}\Sigma_{\mathrm{i}}\mathrm{f}_1(\mathrm{X}_{\mathrm{i}}) - \frac{1}{\mathrm{N}}\Sigma_{\mathrm{i}}\mathrm{f}_2(\mathrm{X}_{\mathrm{i}}) = \frac{1}{N}\Sigma_{\mathrm{i}}\left(f_1(X_i) - f_2(X_i)\right)$
  - $var(\hat{\theta}) = var\left(\frac{1}{N}\Sigma_{\mathrm{i}}\left(f_1(X_i) - f_2(X_i)\right)\right) = 1/N^2\Sigma_i var\left(f_1(X_i) - f_2(X_i)\right)$
  - $var\left(f_1(X_i) - f_2(X_i)\right) = var\left(f_1(X_i)\right) + var\left(f_2(X_i)\right) - 2cov\left(f_1(X_i), f_2(X_i)\right)$
- What if $cov\left(f_1(X_i), f_2(X_i)\right) = 0$?
  - No actual gain over the ordinary MC estimator.

Common Random Numbers (CRN)

- When the two functions tend to increase (or decrease) together,
  - $cov\left(f_1(X_i), f_2(X_i)\right) > 0$
  - e.g., both functions are linear whose derivatives have the same sign.

- Applications in rendering
  - Estimating image gradients
  - (screened) Poisson reconstruction takes the image gradients to output a reconstructed image

- Q. can we decide whether or not we apply the CRN?
  - In practice, it is hard to know if there is such correlation in advance.
  - However, implementing and testing CRN are very easy.

CRN examples

Path tracing with CRN numbers, 76 samples per pixel

Control Variates

- $\theta = \int_{\Omega} f(x) d\mu(x)$

- Monte Carlo estimator
  - $\hat{\theta} = \frac{1}{\mathrm{N}} \Sigma_{\mathrm{i}} \mathrm{f}(\mathrm{X}_{\mathrm{i}})$

- Define a control variate $g(x)$ whose integration G is known.
  - $\theta = \int_{\Omega} f(x) - \alpha g(x) d\mu(x) + \alpha G$
  - $\hat{\theta}^{cv} = \frac{1}{\mathrm{N}} \Sigma_{\mathrm{i}} \left(\mathrm{f}(\mathrm{X}_{\mathrm{i}}) - \alpha \mathrm{g}(\mathrm{X}_{\mathrm{i}}) \right) + \alpha G$

Control Variates

- When $\alpha = 1$
  - $\hat{\theta}^{cv} = \frac{1}{\mathrm{N}} \Sigma_{\mathrm{i}} \big(\mathrm{f}(\mathrm{X_i}) - \mathrm{g}(\mathrm{X_i})\big) + G$
  - $var(\hat{\theta}^{cv}) = \frac{1}{\mathrm{N}^2} \Sigma_{\mathrm{i}} var\big(\mathrm{f}(\mathrm{X_i}) - \mathrm{g}(\mathrm{X_i})\big) = 1/N^2 \{\Sigma_{\mathrm{i}} var\big(f(X_i)\big) + var\big(g(X_i)\big) - 2cov\big(f(X_i), g(X_i)\big)\}$
  - Assume that:
    - $var\big(f(X_i)\big) = var\big(g(X_i)\big) = \sigma^2$
    - $cov\big(f(X_i), g(X_i)\big) = \sigma_{f,g}^2$
    - $corr\big(f(X_i), g(X_i)\big) = corr_{f,g} = \frac{\sigma_{f,g}^2}{\sigma_f \sigma_g} = \sigma_{f,g}^2 / \sigma^2$
  - $var(\hat{\theta}^{cv}) = \frac{\Sigma_{\mathrm{i}} 2\sigma^2}{N^2} - \frac{2\Sigma_i \sigma_{f,g}^2}{N^2} = \frac{2\sigma^2}{N} - \frac{2\sigma_{f,g}^2}{N}$
  - Variance of the original estimator with the same sample count N
    - $var(\hat{\theta}) = \frac{\sigma^2}{2N}$
  - Condition for $var(\hat{\theta}^{cv}) < var(\hat{\theta})$
    - $\frac{2\sigma^2}{N} - \frac{2\sigma_{f,g}^2}{N} < \frac{\sigma^2}{2N}$
    - $\frac{3\sigma^2}{4} < \sigma_{f,g}^2$
    - $\frac{3}{4} < corr_{f,g}$

Control Variates

- When $\alpha = \frac{\sigma_{f,g}^2}{\sigma_g^2} = \frac{\sigma_{f,g}^2}{\sigma^2}$
  - $\hat{\theta}^{cv} = \frac{1}{N}\Sigma_i\big(f(X_i) - \alpha g(X_i)\big) + \alpha G$
  - $var(\hat{\theta}^{cv}) = \frac{1}{N^2}\Sigma_i var\big(f(X_i) - \alpha g(X_i)\big) = 1/N^2\{\Sigma_i var\big(f(X_i)\big) + \alpha^2 var\big(g(X_i)\big) - 2\alpha cov\big(f(X_i), g(X_i)\big)\}$
  - $= \frac{\sigma^2}{N} + \frac{\alpha^2\sigma^2}{N} - \frac{2\alpha\sigma_{f,g}^2}{N}$
  - $= \frac{\sigma^2}{N} + \frac{corr_{f,g}^2\sigma^2}{N} - \frac{2corr_{f,g}^2\sigma^2}{N}$
  - $= \frac{\sigma^2}{N}\big(1 - corr_{f,g}^2\big)$
  - Condition for $var(\hat{\theta}^{cv}) < var(\hat{\theta})$
    - $\frac{\sigma^2}{N}\big(1 - corr_{f,g}^2\big) < \frac{\sigma^2}{2N}$
    - $1/2 < corr_{f,g}^2$
    - $\frac{1}{4} < |corr_{f,g}|$

Outline

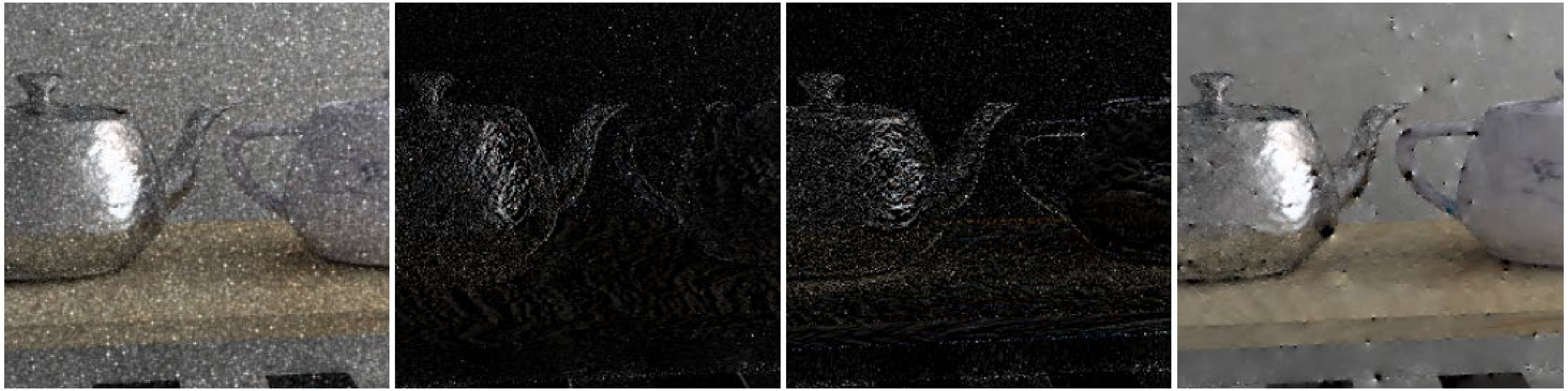
- Russian Roulette and Splitting
- Importance Sampling
- Multiple Importance Sampling
- Adaptive Rendering (Sampling and Reconstruction)
- Correlated Sampling (Antithetic Variates, Common Random Numbers, Control Variates)
- Gradient-Domain Rendering

Gradient-Domain Rendering

- Image gradients can be estimated via correlated sampling
  - $\mathrm{I}(\mathrm{x}+1,\mathrm{y})-\mathrm{I}(\mathrm{x},\mathrm{y})$
  - $\mathrm{I}(\mathrm{x},\mathrm{y}+1)\ -\ \mathrm{I}(\mathrm{x},\mathrm{y})$
- The variance of the estimated gradients can be smaller than the pixel color when then the covariance term is positive
  - $var(\mathrm{I}(\mathrm{x}+1,\mathrm{y})-\mathrm{I}(\mathrm{x},\mathrm{y}))=var\big(\mathrm{I}(\mathrm{x}+1,\mathrm{y})\big)+var(\mathrm{I}(\mathrm{x},\mathrm{y}))-2cov(\mathrm{I}(\mathrm{x}+1,\mathrm{y}),\mathrm{I}(\mathrm{x},\mathrm{y}))$

Gradient-Domain Rendering

- Rendering estimates three images:
  - Primal colors (e.g., standard path tracing)
  - Image gradients (e.g., correlated samplings such as CRNs, shift mapping, and path reusing)



(a) Input color image, 512 spp relMSE 0.1318 | (b) Input gradients in horizontal direction | (c) Input gradients in vertical direction | (d) L2 relMSE 0.0184

Images from [Ha et al. 2019]

Gradient-Domain Rendering

- Screened Poisson Reconstruction
  - $\hat{y} = \underset{\bar{y}}{\mathrm{argmin}} \sum_{i=1}^{n} \|\alpha(y_i - \bar{y}_i)\|^2 + \sum_{i=1}^{n} \left\|g_i^{dx} - D^{dx}\bar{y}_i\right\|^2 + \sum_{i=1}^{n} \left\|g_i^{dy} - D^{dy}\bar{y}_i\right\|^2$
- g_i^{dx}, g_i^{dy}: Estimated gradients at pixel i in x and y directions
- y_i: Pixel color at i
- α: user-parameter (e.g., 0.2)
- D^{dx}, D^{dy}: differential operator in x and y directions (i.e., finite differences)
- Has a closed-form solution (i.e., normal equation) when the norm is $L2$

Gradient-Domain Rendering

- When L2 reconstruction is used, the output is unbiased.
- One may use a neural network that takes the three inputs
  - e.g., Deep Convolutional Reconstruction for Gradient-Domain Rendering, Kettunen et al. 2019
- More information:
  - EG STAR paper 2019: A Survey on Gradient-Domain Rendering, Hua et al. 2019