Adaptive Rendering based on Weighted Local Regression

Bochang Moon[1], Nathan Carr[2], Sung-Eui Yoon[1]
[1]KAIST, [2]Adobe

Monte Carlo ray tracing is considered one of the most effective techniques for rendering photo-realistic imagery, but it requires a large number of ray samples to produce converged or even visually pleasing images. We develop a novel image plane adaptive sampling and reconstruction method based on local regression theory. A novel local space estimation process is proposed for employing the local regression, by robustly addressing noisy high dimensional features. Given the local regression on estimated local space, we provide a novel two-step optimization process for selecting bandwidths of features locally in a data-driven way. Local weighted regression is then applied using the computed bandwidths to produce a smooth image reconstruction with well preserved details. We derive an error analysis to guide our adaptive sampling process at the local space. We demonstrate that our method produces more accurate and visually pleasing results over the state-of-the-art techniques across a wide range of rendering effects. Our method also allows users to use an arbitrary set of features including noisy features, and robustly computes a subset of them by ignoring noisy features and decorrelating them for higher quality.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Raytracing*

General Terms: Algorithms

Additional Key Words and Phrases: Adaptive rendering, image-space reconstruction, Monte Carlo ray tracing

1. INTRODUCTION

Methods for adaptively rendering and reconstructing images for Monte Carlo (MC) ray tracing have a long history [Mitchell 1987]. Recent work has made further advances in this space, but the primary goal remains the same: to improve image quality using a reduced number of ray samples in order to gain efficiency. Images generated with too few ray samples are plagued with noise, and convergence to a smooth image is quite slow. The key ingredient of adaptive sampling and reconstruction is to locally perform error analysis, which guides ray budgets on high error regions while controlling smoothing for image reconstruction, to produce numerically and visually pleasing rendering results.

DOI 10.1145/XXXXXXX.YYYYYYY
http://doi.acm.org/10.1145/XXXXXXX.YYYYYYY

At a high level, adaptive rendering techniques can be classified as integrand- and image-space approaches. Powerful integrand methods [Hachisuka et al. 2008] have been proposed, but recent research efforts have been focusing on designing effective image-space, adaptive sampling and reconstruction techniques. This is mainly because the image-space approach is efficient, is simple to integrate into existing rendering systems, and handles a wide variety of complex rendering effects such as motion blur, depth-of-field, etc.

Recent image-space reconstruction techniques use well-known filtering methods proposed in the image processing field such as Gaussian filters [Rousselle et al. 2011], non-local means [Rousselle et al. 2012; Li et al. 2012], joint-bilateral filtering [Ritschel et al. 2009; Li et al. 2012; Sen and Darabi 2012], and wavelets [Overbeck et al. 2009]. A key difference between reconstruction techniques developed in image processing and those used in rendering is that filtering methods used in rendering are tailored to leverage different types of available features such as normals, textures, and depths. Since these features serve as less noisy cues for denoising MC rendering results, a lot of prior image-space techniques have utilized some types of features and been able to achieve impressive rendering results.

These features, unfortunately, can be considered as double-edged swords. Even features can be very noisy, especially when we have scenes with complex motion, geometry, textures, and light paths. Furthermore, different types of features can have varying importance on the overall filtering process, as demonstrated by a random parameter filtering [Sen and Darabi 2012]. Nonetheless, these issues have received little or no attention in prior image-space adaptive rendering methods that heavily utilize these features.

Contributions. In this paper we propose a novel, weighted local regression based adaptive reconstruction and sampling technique to effectively handle a wide variety of MC rendering effects. Departing from prior techniques, we estimate an unknown output function based on local weighted regression (Sec. 3). This approach enables us to derive error analysis in a robust way, measure importance of different types of features, and mitigate the curse of dimensionality as we consider more features in a consistent and principled way. Specifically, we provide the following technical contributions:

—We identify a reduced, local feature space that effectively guides our reconstruction based on a truncated SVD (Singular Value Decomposition) and perturbation theory (Sec. 4).

—The Mean Squared Error (MSE) of our reconstruction method is decomposed into bias and variance terms (Sec. 5.1), and we robustly estimate them based on parametric error analysis (Sec. 5.3).

—We use partial derivatives to estimate the importance of each feature type and compute different filtering widths for features (Sec. 5.2), resulting in effective anisotropic filtering.

—We use the reduced dimension from the local feature space analysis to optimally distribute rays (Sec. 6).

According to our best knowledge, weighted local regression in a high dimensional feature space has not been applied to image-space adaptive rendering, and our adaptive rendering technique based on

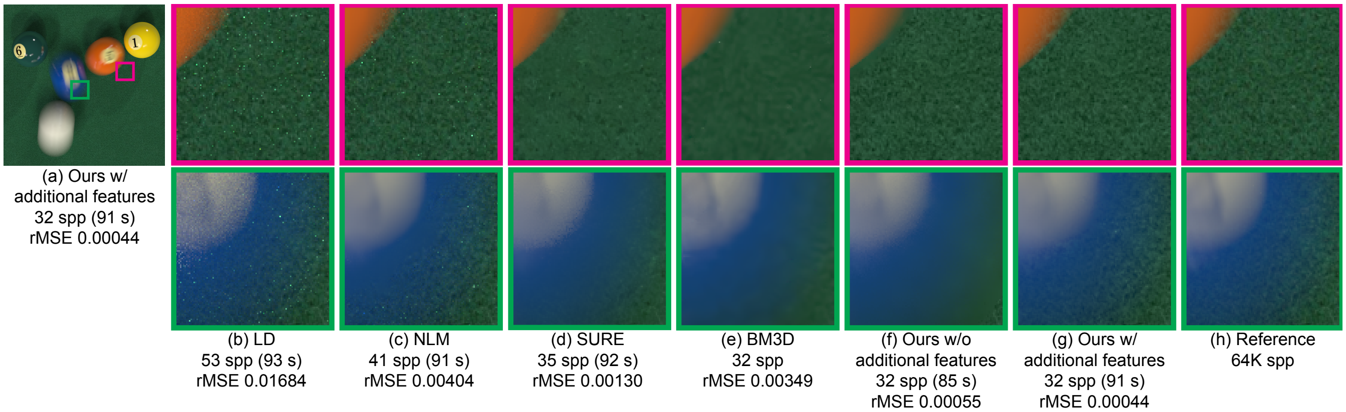


Fig. 1. Equal-time comparisons in the pool scene. The image (b) is generated by 53 uniform low discrepancy samples per pixel (spp). NLM [Rousselle et al. 2012] leaves high-frequency noise. SURE [Li el al. 2012] and BM3D [Kalantari and Sen 2013] show over-blurred results. As a quantitative measure for comparisons, we use the relative MSE (rMSE) [Rousselle et al. 2011]. Our method shows a lower rMSE compared to previous methods. In our result (g), extended feature buffers (i.e., geometries from moving and non-moving objects) are included, and filter bandwidths for the features are automatically chosen. Model courtesy of Toshiya Hachisuka.

local regression is the first technique integrating robust error analysis, automatic feature bandwidth selection, and a tightly coupled approach between reconstruction and sampling in a consistent and principled way.

To demonstrate benefits of our method we have applied our method to a variety of benchmarks with different rendering effects. Our method automatically chooses filtering bandwidths in a reduced local space given a arbitrary set of geometries (e.g., normal, texture, and depth). Compared to the state-of-the-art methods [Rousselle et al. 2012; Li et al. 2012; Kalantari and Sen 2013], our method shows visually pleasing and numerically better results across tested benchmarks, thanks to automatic bandwidth selections and adaptive sampling based on an estimated local feature space per pixel. In addition our framework is naturally extended to handle animations in a principled way, which minimizes flickering artifacts as well as numerical errors.

2. PREVIOUS WORK

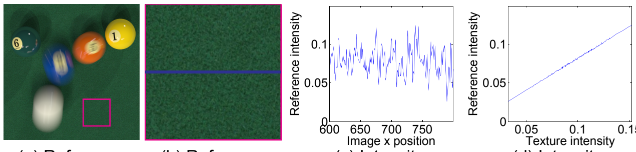
Image space reconstruction methods. Image filtering has been a popular approach to remove noise in images generated by MC ray tracing, because of its simplicity and efficiency. Rushmeier and Ward [1994] proposed a nonlinear filter that distributes the energy of outliers to neighboring pixels in an energy preserving way. Meyer and Anderson [2006] applied principle component analysis to removing noise in an animation sequence. Our method also employ a dimensionality reduction method (i.e., truncated SVD) for reducing noise in geometric buffers. It has been recognized that geometric information can play an important role for predicting edges in rendered images. For example, McCool [1999] proposed an anisotropic diffusion process guided by geometric features. The geometric features have also been widely used for quickly producing reasonable image quality in interactive rendering techniques [Segovia et al. 2006; Ritschel et al. 2009; Dammertz et al. 2010; Bauszat et al. 2011; Shirley et al. 2011].

Sen *et al.* [2012] recently showed that by analyzing the functional relationships between MC inputs and outputs, feature weights can be computed to achieve high quality image reconstructions with a low number of samples. While their proposed method of *random parameter filtering* (RPF) produces exceptional reconstruction results, it comes at a high computation and storage cost. In contrast our method is designed to have both a low storage and computation overhead, while producing better filtered results efficiently.

Our method departs from earlier work in that it is derived from local regression theory appearing in the statistics literature [Cleveland and Loader 1996; Ruppert and Wand 1994]. We reconstruct a surface, allowing us to efficiently estimate bias, variance, and partial derivatives, which play a crucial role in our error analysis. This in turn allows us to select ideal filter widths across multiple feature dimensions.

Multi-dimensional reconstruction and sampling. Numerous methods have been proposed that operate in integrand space, where reconstruction and sampling is performed in high dimensions. Hachisuka *et al.* [2008] proposed a general multi-dimensional adaptive rendering method that reconstructs smooth images using Riemann sums. This method works well in a low dimensional space, however its effectiveness rapidly degrades as the dimension increases. Similar approaches have been proposed to examine a reduced set of effects to mitigate the so-called *curse of dimensionality*. The shared reconstruction filters based on frequency analysis were designed for efficiently rendering depth-of-field effects [Soler et al. 2009], motion blur [Egan et al. 2009], soft shadows [Egan et al. 2011], and ambient occlusions [Egan et al. 2011]. Recently, Belcour *et al.* [2013] proposed a 5D frequency analysis for efficiently handling depth-of-field and motion blur effects. Lehtinen *et al.* [2011] proposed a reconstruction method by using depth and motion information of per sample to synthesize depth-of-field, motion blur, and soft shadows. Lehtinen *et al.* [2012] developed an advanced filter that reuses rays across pixels for a high quality reconstruction of indirect illuminations. While multi-dimensional reconstruction and sampling methods have shown exceptional performance even with a small number of samples per pixel, these approaches often focused on specific rendering effects. The generality of our method allows us to work across a wide range of effects. In addition, we are able to determine an appropriate, reduced dimensional feature subspace locally, avoiding an unnecessary growth in dimension and improving efficiency.

Image space adaptive rendering. We now detail the techniques that are most closely related to our method. Overbeck *et al.* [2009] developed a framework that treats sampling and reconstruction as a coupled iterative process. They decompose the image into wavelets and apply shrinkage to the coefficients to reduce noise. The same iterative framework has been adopted by more recent works [Rous-



(a) Reference (b) Reference (c) Intensity on x position (d) Intensity on texture

Fig. 2. The reference image (a) with its zoomed image (b) within the magenta box is generated with 64K samples per pixel in the pool scene. Intensity plots (c) and (d) are generated based on data along the blue horizontal line in the magenta box. The intensity plot (c) of image position shows a strong variation due to a noisy texture, but this non-linearity in the image position becomes an almost linear function in the texture space (d).

selle et al. 2011; 2012; Li et al. 2012; Kalantari and Sen 2013], and is also a foundation upon which we build our research.

Rousselle *et al.* [2011] demonstrated improved image quality by greedily selecting an appropriate isotropic filtering bandwidth locally across the image. This work was further improved by introducing state-of-the-art anisotropic filtering methods such as the cross bilateral and non-local means filters [Rousselle et al. 2012; Li et al. 2012]. Li *et al.* [2012] adopted Stein's unbiased risk estimator for sampling and bandwidth selection. Using this estimator their system is capable of supporting a wide range of anisotropic filters including the cross bilateral and non-local means. One limitation of their work is that their bandwidth selection process is over the spatial dimension and relies on a set of fixed *global* parameters for controlling the influence of other geometric features such as depth, texture, and normals. As shown by Sen *et al.* [2012] analyzing the functional relationship between the geometric feature information and output intensity can significantly improve image quality. Our work addresses this problem by performing an optimal bandwidth selection locally for different feature types.

3. LOCAL REGRESSION BASED FILTERING

In this section we give a brief background on local regression and its application to reconstruction. Local regression [Cleveland and Loader 1996; Ruppert and Wand 1994] is a smoothing method for fitting parametric curves or surfaces, $f(\mathbf{x})$, based on a neighborhood of $\mathbf{x}$. Its underlying statistical model is:

$$y = f(\mathbf{x}) + \epsilon, \tag{1}$$

where y and $\mathbf{x}$ denote $\mathbb{R}^1$-valued response variable and $\mathbb{R}^D$-valued predictor variables, respectively. The noise ϵ is modeled by additive random noise that has zero mean and bounded variance.

In our problem $y \in \mathbb{R}^1$ and $f(\mathbf{x}) \in \mathbb{R}^1$ denote a noisy MC input image and an unknown ground truth image, respectively. Given color images as the input, we apply our method to each channel independently. The feature vector $\mathbf{x} \in \mathbb{R}^D$ has D dimensions that includes image positions as well as any arbitrary set of additional geometric information including textures, depth, and normals. To compute the feature vector at a pixel we average out geometries computed from multiple primary rays. The statistical model assumes that $\mathbf{x}$ is a noise-free $\mathbb{R}^D$ vector, but it is well-known that the feature vector $\mathbf{x}$ can be noisy due to distributed effects such as depth of fields. We address this problem by a regularization method (Sec. 4) that is a crucial part for employing local regression in rendering.

The unknown intensity $f(\mathbf{x})$ at $\mathbf{x}$ nearby a center feature vector $\mathbf{x}^c$ can be approximated locally based on a Taylor polynomial of order one as follows:

$$f(\mathbf{x}) \approx f(\mathbf{x}^c) + \nabla f(\mathbf{x}^c)^T(\mathbf{x} - \mathbf{x}^c). \tag{2}$$

For simplicity let the coefficients of the local linear model for the unknown image value $f(\mathbf{x}^c)$ and its gradient $\nabla f(\mathbf{x}^c)$ be α and β respectively. A weighted least squares minimization can be formulated to determine the unknown coefficients α and β as follows:

$$[\hat{\alpha}, \hat{\beta}] = \min_{\alpha,\beta} \sum_{i=1}^{n} \left(y^i - \alpha - \beta^T(\mathbf{x}^i - \mathbf{x}^c)\right)^2 \Pi_{j=1}^{D} w\left(\frac{\mathbf{x}_j^i - \mathbf{x}_j^c}{h\mathbf{b}_j}\right), \tag{3}$$

where $\mathbf{x}^i$ and y^i are the feature vector and intensity at pixel i, respectively. The value of n denotes the number of pixels within a square window region centered around pixel c. For filtering of a center feature vector $\mathbf{x}^c$ we define its neighboring feature vectors as $\mathbf{x}^i$ in a filtering window. The computed coefficients $\hat{\alpha}$ and $\hat{\beta}$ then correspond to the filtered image and estimated gradient for our reconstruction problem, respectively. The term $\Pi_{j=1}^{D} w\left(\frac{\mathbf{x}_j^i - \mathbf{x}_j^c}{h\mathbf{b}_j}\right)$ is a chosen multi-dimensional kernel based on the product of 1-D kernels. The kernel $w(\cdot)$ is commonly chosen to be a symmetric decreasing function (e.g., Gaussian or Epanechnikov kernel) for allocating high weights to close samples from $\mathbf{x}^c$. Note that the distance is computed in the feature space $\mathbb{R}^D$. The bandwidth term $h\mathbf{b}_j$ controls filtering bandwidth for the j-th feature. In our formulation we have chosen to separate the bandwidth component into two separate terms: $\mathbf{b}_j$ that varies per feature, and a scaling factor h that is shared across all feature types. This breakdown aids in the process of selecting optimal bandwidths for reconstruction and will be discussed in more detail in Sec. 5.

Note that the minimization of Eq. 3 is also used in filtering methods based on a linear regression with uniform weighting [Bauszat et al. 2011; He et al. 2013]. The main difference between these prior approaches and our optimization goal (Eq. 3) is in the kernel function $w(\cdot)$. The uniform kernel, i.e. $w(\cdot) = 1$, is used in the prior filtering methods, since it is simple and efficient. On the other hand, our formulation uses a general kernel function that allows a set of bandwidths $h\mathbf{b}_j$ enabling a higher reconstruction quality.

Local linear approximation in the feature space. Our approach locally approximates an unknown image function $f(\mathbf{x})$ with a low order polynomial in a feature space $\mathbf{x} \in \mathbb{R}^D$. This approach could be seen as counter-intuitive, since the unknown function of a rendered image has discontinuities introduced by a variety of rendering effects and is thus typically nonlinear. Our key observation is that the non-linearity in the 2D image space can be well approximated by the local linear model in a high dimensional feature space, as shown in Fig. 2. In rendering, some effects can be nonlinear even in the presence of a high dimensional feature space. In such cases our local linear model can create a bias due to the curvature of the signal being reconstructed. To minimize bias as well as variance we propose a two-step optimization method (Sec. 5). We have considered other alternatives such as logistic regression for handling such discontinuities as well as higher order local regression, but found that our weighted linear local regression strikes an excellent balance between denoising quality and computational efficiency, while preserving feature edges well.

Normal equation. The optimization problem shown in Eq. 3 has the following closed form solution, i.e. normal equation:

$$[\hat{\alpha}, \hat{\beta}]^T = (X^T W X)^{-1} X^T W Y, \tag{4}$$

where X is the $n \times (D+1)$ design matrix whose i-th row is set as $[1, (\mathbf{x}^i - \mathbf{x}^c)^T]$, and Y is the $n \times 1$ output vector $Y = [y^1, ..., y^n]^T$. The matrix W is an $n \times n$ diagonal matrix, whose elements are defined by the weight function $\Pi_{j=1}^{D} w(\frac{\mathbf{x}_j^i - \mathbf{x}_j^c}{h\mathbf{b}_j})$.

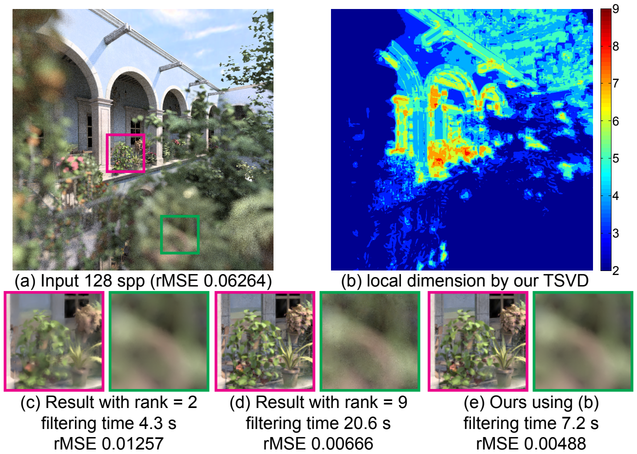


Fig. 3. Given an input image (a), we compute local dimensions (b) based on our TSVD. The reconstructed images by using a global small (c) or large rank (d) show over-blurred artifacts in the in-focused region (magenta box) or noise in defocused region (green box), respectively. Our method (e) based on locally determined ranks is visually and numerically better than the results (c) and (d). In addition, our method (e) has a lower filtering overhead compared to the method using the large rank (d). Model courtesy of Guillermo M. Leal Llaguno.

Intuitively speaking, the bandwidths $h\mathbf{b}_j$ act as smoothing parameters; controlling the trade-off between bias and variance of our local regression based reconstruction method. We present our optimization process in Sec. 5 to choose bandwidth parameters such that we minimize the Mean Squared Error (MSE) of our reconstruction.

The normal equation with feature vectors defined in a global space $\mathbb{R}^D$ unfortunately can be unstable and require a high cost of computation as we consider more types of features. Instead of working directly in the D dimensional feature space, we propose using truncated singular value decomposition (TSVD) to generate an appropriate reduced dimensional local coordinates to solve our weighted least squares optimization problem (Sec. 4).

By using local regression and performing error analysis for our problem in a principled way, our approach has some significant differences when compared to prior methods. First, our method locally reconstructs a parametric function and minimizes its reconstruction error more robustly over prior approaches [Rousselle et al. 2011; 2012; Li et al. 2012] that perform point estimation on $f(\mathbf{x})$. Second, our method can measure the importance of different types of features by estimating second partial derivatives for features and even ignore noisy input features. Finally, our error analysis is naturally combined with sampling, to guide available ray sample budgets.

4. TSVD BASED LOCAL REGRESSION

Common local regression methods assume that input predictor variables are not noisy, while response variables have noise. Unfortunately, our feature vectors can also carry a significant amount of noise. To address this problem we construct a reduced feature space using truncated SVD (TSVD) for solving Eq. 3.

Before solving the optimization problem (Eq. 3) at every pixel, we use singular value decomposition (SVD) as a means to construct a reduced feature space locally defined in a filtering window. The feature vector $\mathbf{x}$ in a global space is transformed into a vector $\mathbf{z}$ in the local space. We then perform our optimization in the reduced feature space.

The coordinate transformation provided by SVD gives us an elegant means to robustly handle rank deficient systems commonly encountered. When combined with perturbation theory, we are able to pre-filter the matrix space for reducing noise that would lead to poor conditioning and failed reconstruction. At a high level, we can consider this process as a pre-filtering process that reduces noise contained in feature vectors, and it does not depend on the response variable (i.e., intensity). Furthermore, the coordinate transformation enables us to identify new orthogonal feature types, which are linearly combined with some of original feature types. For example, when a 3D input feature vector $[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3]$ is given, the reduced space computed by SVD can be a 2D vector $[\mathbf{x}_1, 0.5\mathbf{x}_2 + 0.5\mathbf{x}_3]$.

Let us define a $n \times D$ matrix Z, whose i-th row is set as $[(\mathbf{x}^i - \bar{\mathbf{x}})^T]$, where $\bar{\mathbf{x}}$ is a mean vector whose j-th component is computed as $\bar{\mathbf{x}}_j = 1/n \sum_{i=1}^{n} \mathbf{x}_j^i$. The SVD on Z leads to a factorization $Z = USV^T$, where U and V are $n \times n$ and $D \times D$ unitary matrices. Also, S is the $n \times D$ diagonal matrix, where diagonal entries, σ_m, are ordered singular values in the non-increasing order.

To solve the noise problem we apply truncated SVD based on perturbation theory [Hansen 1987], which filters out small singular values that fall below a threshold τ by setting them to be zero. This is based on the observation that small singular values are often a result of corruption from noise, and these small corruptions lead to significant changes in the least squares solutions [Hansen 1987]. To adaptively select the threshold τ, we rely on perturbation theory to provide us with a principled approach for selecting an appropriate threshold value of τ.

From a perturbation theory perspective, a perturbation matrix Z is expressed by $Z = \mu(Z) + E$, where $\mu(Z)$ is the ground truth matrix and E is the perturbation matrix containing error and noise. We then have the following Weyl's perturbation bound [Stewart 1993]:

$$|\sigma_m - \hat{\sigma}_m| \leq \|E\|_2, \tag{5}$$

where σ_m and $\hat{\sigma}_m$ are the m-th singular value of $\mu(Z)$ and Z, respectively. Also, $\|E\|_2$ is the spectral norm of E.

In our problem $\mu(Z)$ can be obtained with an infinite number of ray samples, and E indicates the perturbation structure on Z introduced by MC ray tracing. By the central limit theorem and independence property of MC ray samples [1], we can estimate the noisy matrix E. Specifically, we assume each element in Z follows a normal distribution, $Z_{ij} \sim \mathcal{N}\left(\mathbf{x}_j^i - \bar{\mathbf{x}}_j, var(\mathbf{x}_j^i)\right)$. In this model, $var(\bar{\mathbf{x}}_j)$ is assumed to be zero, since the term is defined as $var(\bar{\mathbf{x}}_j) = 1/n^2 \sum_{i=1}^{n} var(\mathbf{x}_j^i)$ and tends to be very small due to the $1/n^2$ term. We then model each element in E as $E_{ij} = \left(var(\mathbf{x}_j^i)\right)^{1/2}$. We estimate the unknown variance $var(\mathbf{x}_j^i)$ as the variance of the sample mean of j-th feature at pixel i, and it is computed at each pixel by using the samples generated by Monte Carlo ray tracing.

The Weyl's perturbation bound in Eq. 5 gives us upper and lower bounds for a range of σ_m given observed values $\hat{\sigma}_m$ and E. According to these lower and upper bounds, potential values for σ_m can be negative or zero, when $\hat{\sigma}_m$ is equal to or lower than $\|E\|_2$. Since σ_m cannot be negative or zero, we set τ conservatively to be $C\|E\|_2$. In our test scenes, we set C as 2 to remove the perturbation effects as long as possible. For example, we observed that using a relatively large constant $C = 2$ tends to generate visually smooth results in defocused regions (e.g., Fig. 3). On the other hand, when a smaller constant $C = 1$ is used, the MSE can be further improved.

[1] Based on the central limit theorem, the distribution of sample means goes to a normal distribution as sample count increases. In addition the approximation quality depends on the number of samples [Moon et al. 2013].

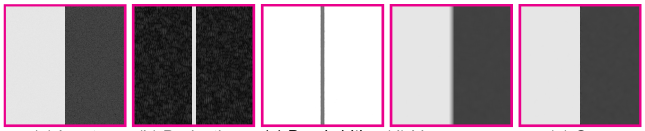


Fig. 4. Given a noisy step function (a), estimated derivatives (b) and bandwidths (c) computed for image position x are visualized. In this example, we do not use any geometries. While the linear approximation (d) with a large global bandwidth produces over-blurred results, our method (e) guided by the estimated bandwidths (c) preserves the strong edge.

However, this often comes at the expense of increased noise. We found that the relative difference in MSE between using $C = 1$ and $C = 2$ in practice is not high (e.g. from 0.00488 to 0.00431). Thus we have chosen $C = 2$ for making more visually pleasing images.

After performing SVD and identifying the k biggest singular values by filtering out small ones based on perturbation theory, we then approximate the matrix Z as those k largest singular values and corresponding singular vectors. This can be expressed by a compact form $Z \approx U_k S_k V_k^t$, where U_k and V_k are $n \times k$ and $D \times k$ reduced unitary matrix respectively, and S_k is the diagonal singular matrix that has k non-zero singular values. In our problem, $\|E\|_2$ tends to monotonically decrease as a function of the number of samples. As a result, the bias introduced by the k-rank approximation goes to zero.

Given a input vector $\mathbf{x} \in \mathbb{R}^D$ defined in a global space, the vector is transformed to a local space by using a vector-matrix multiplication $\mathbf{z} = V_k^T \mathbf{x} \in \mathbb{R}^k$. Since each column of the unitary matrix V_k is an orthonormal basis geometrically, this transformation can be considered as a projection on the basis. The optimization problem (Eq. 3) is then modified as follows:

$$\min_{\alpha,\beta} \sum_{i=1}^{n} \left(y^i - \alpha - \beta^T(\mathbf{z}^i - \mathbf{z}^c)\right)^2 \Pi_{j=1}^{k} w\left(\frac{\mathbf{z}_j^i - \mathbf{z}_j^c}{h\mathbf{b}_j}\right). \quad (6)$$

Note that the coefficients α and β estimated by the optimization correspond to the unknown image value $f(\mathbf{z}^c)$ and its gradient $\nabla f(\mathbf{z}^c)$, respectively, in the transformed space $\mathbb{R}^k$. Also, the i-th row in the design matrix X of the normal equation (Eq. 4) is replaced with $[1, (\mathbf{z}^i - \mathbf{z}^c)^T]$.

Fig. 3 shows reconstruction results computed by using a global rank or locally adaptive ranks determined by our TSVD. The local dimension selection shows visually pleasing images as well as numerically correct images compared to the user-specified dimensions. In addition, since most regions in our method have relatively small dimensions (e.g., 2 to 4), our filtering time (e.g., 7.2 sec.) is more than two times faster than the time required to filter using the full rank. Furthermore, even in the focused areas, the selected dimension can be lower than the input dimension (e.g., 9). Technically, when there is strong dependency between input features, our method can decorrelate them and generate a lower dimension thanks to TSVD. As a result, users can use many features for high quality reconstruction, while our method can efficiently perform filtering with a proper set of features in the reduced local space. For examples, even in the focused scene of killeroo-gold (Fig. 5), the average rank is low (e.g., 3.5 for 32 spp), which results from removing the dependency between input features.

To solve the modified optimization problem (Eq. 6), optimizing bandwidths $h\mathbf{b}_j$ should be determined locally, and is described in the next section (Sec. 5). Note that the bandwidths determines relative importance of each orthogonal feature $\mathbf{z}_j$, not for the input feature $\mathbf{x}_j$.

Algorithm 1 Local Regression based Adaptive Rendering

Input: Feature vector $\mathbf{x}^i$ and intensity y^i generated by renderer
Output: Sampling map for the next rendering pass or final output
for each pixel center c **do**
  Compute transformed feature $\mathbf{z}^i$ and $\mathbf{z}^c$ using TSVD (Sec. 4)
  Estimate $\mathbf{b}_j$ based on second partial derivatives (Sec. 5.2)
  for each $h \in \{h_{min}, h_{max}\}$ **do**
    Compute $\hat{f}_{h\mathbf{b}}(\mathbf{z}^c)$ using local linear regression
    Estimate bias and variance of $\hat{f}_{h\mathbf{b}}(\mathbf{z}^c)$ (Sec. 5.3)
  end for
  Compute h_{opt} using parametric error analysis (Sec. 5.3)
  Compute $\hat{f}_{h_{opt}\mathbf{b}}(\mathbf{z}^c)$ using local linear regression
end for
if Sampling budget remains **then**
  return a sampling map based on $\Delta rMSE(\mathbf{x})$ (Sec. 6)
else
  return $\hat{f}_{h_{opt}\mathbf{b}}(\mathbf{z}^c)$
end if

5. ADAPTIVE RECONSTRUCTION

Given the transformed features by using TSVD (Sec. 4), we solve the optimization problem (Eq. 6) to reconstruct a high quality image. As shown in Fig. 4, the linear approximation fails to preserve edges when the geometric information does not help predict the edges. However, the linear approximation guided by local bandwidth selections preserves the edge thanks to the small bandwidths selected on the edges. In this section, we detail a novel two-step bandwidth selection algorithm that estimates optimal feature bandwidths $\mathbf{b}_j$ and shared bandwidth h separately. Pseudocode for our overall algorithm is provided in Algorithm 1. We begin by formalizing the goal of our reconstruction method and provide the high level overview of our approach (Sec. 5.1), followed by our optimization process selecting optimal filter bandwidths (Sec. 5.2 and Sec. 5.3).

5.1 Optimization Goal

The goal of our local regression based filtering is to minimize MSE, which consists of the bias and variance terms: $MSE = bias(\hat{f}_{h\mathbf{b}}(\mathbf{z}))^2 + var(\hat{f}_{h\mathbf{b}}(\mathbf{z}))$ by adjusting reconstruction bandwidths $h\mathbf{b}_j$ in Eq. 6, where h is a shared bandwidth for modulating the magnitude of all the feature bandwidths and $\mathbf{b}_j$ controls j-th feature bandwidth for feature types in the local feature space. We seek a set of bandwidths $h\mathbf{b}_j$ that reduce MSE of each pixel; h and $\mathbf{b}_j$ are defined for each pixel, but we do not explicitly encode that information in their notations for simplicity.

One may consider well-known estimators such as a local cross validation [Cleveland and Loader 1996] and Stein's unbiased risk [Li et al. 2012] that work as a surrogate of MSE. Given one of these estimators, we could test all the possible combinations of bandwidth values within a user-defined set (e.g., $h\mathbf{b}_j \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$). We can then select an optimal bandwidth that minimizes the error. This approach unfortunately requires an exponentially growing number of tests, as we consider including a greater number of features which lead to higher quality reconstructions. Furthermore, the number of features can be high, since different features such as geometry introduced by secondary rays [Sen and Darabi 2012] can be also used for high quality reconstruction. As an alternative, one can solve only a subset (e.g., the spatial filter's bandwidth in cross bilateral) of all the feature bandwidths [Li et al. 2012], but this approach results in a sub-optimal solution.

Departing from these alternatives, we use well-established bias and variance estimators developed from the local regression literature [Ruppert and Wand 1994]. The bias term has the following asymptotic relationship:

$$bias(\hat{f}_{h\mathbf{b}}(\mathbf{z})) \propto 0.5h^2 trace(B\mathbf{H}\hat{f}_{h\mathbf{b}}(\mathbf{z})), \quad (7)$$

where a diagonal matrix B is set as $diag(\mathbf{b}_1^2, ..., \mathbf{b}_k^2)$, $trace(.)$ is the matrix trace, and $\mathbf{H}\hat{f}_{h\mathbf{b}}(\mathbf{z})$ is the Hessian matrix. In addition,

$$var\left(\hat{f}_{h\mathbf{b}}(\mathbf{z})\right) \propto \frac{1}{n(\mathbf{z})h^k \Pi_{j=1}^k \mathbf{b}_j}, \quad (8)$$

where $n(\mathbf{z})$ denotes the number of samples at $\mathbf{z}$.

One noticeable advantage of using these expressions compared to the general estimators is that the bias and variance term are functions of bandwidths. Unfortunately, the challenging problem of optimizing bandwidths even with these expressions remains in practice, since the asymptotic expressions merely gives the asymptotic relationship; the optimal bandwidths are not directly computable from the asymptotic expression. This is the main reason why we propose a novel two-step optimization process for estimating optimal bandwidths by considering the asymptotic theory as a crucial hint. Specifically, based on the functional relationship, we propose an estimation process of per-feature bandwidths $\mathbf{b}_j$ by estimating their corresponding second partial derivatives (Sec. 5.2). We then optimize the shared bandwidth h given the estimated feature bandwidths (Sec. 5.3).

5.2 Estimating Feature Bandwidths $\mathbf{b}_j$

Since the matrix B is a diagonal matrix, the asymptotic bias term (Eq. 7) is reduced to $0.5h^2 \sum_{j=1}^k \mathbf{b}_j^2 \frac{\partial^2 f}{\partial \mathbf{z}_j^2}$. The asymptotic bias term consists of multiple bias terms, each of which is introduced from each feature subspace. The amount of bias caused by considering j-th feature $\mathbf{z}_j$ is proportional to $(h\mathbf{b}_j)^2 |\frac{\partial^2 f}{\partial \mathbf{z}_j^2}|$. Intuitively, selecting a small bandwidth for a feature with a strong second partial derivative is desirable to reduce its corresponding bias, according to the asymptotic bias expression. Small bandwidths, however, lead to an increase in the asymptotic variance expression (Eq. 8). In other words, the second derivatives tell us the relative impact that each feature has on our reconstruction error. Our key idea, two-step bandwidth selection, is to estimate $\mathbf{b}_j$ by using the relation between the bias term and the second derivatives, followed by adjusting the shared bandwidth term h by considering both the asymptotic bias and variance expression.

Given h the bias (Eq. 7) is computed as a weighted sum of $\mathbf{b}_j^2$ and $|\frac{\partial^2 f}{\partial \mathbf{z}_j^2}|$. We therefore choose $\mathbf{b}_j$ to be $|\frac{\partial^2 f}{\partial \mathbf{z}_j^2}|^{-0.5}$. Given this equation, our next goal is to estimate the second partial derivatives of the unknown function $f(\mathbf{z})$.

Estimating derivatives or the Hessian matrix has been a crucial part for predicting or minimizing errors for various rendering problems, and it is often tailored to target problems. For example, the irradiance Hessian is estimated on surfaces for irradiance caching [Schwarzhaupt et al. 2012], and derivatives of radiances are estimated using derivatives of kernel functions in progressive photon mapping [Hachisuka et al. 2010]. We also estimate the Hessian $\mathbf{H}$ within our framework. One of the fundamental benefits of our formulation based on local regression is that we can easily estimate second partial derivatives using quadratic local regression at each pixel, and analytic second derivatives are directly computable from the quadratic solution form.

In our feature bandwidth selection, we do not use off-diagonal terms in the Hessian $\mathbf{H}$ since those terms do not affect the asymptotic bias expression (Eq. 7). In order to lessen the computational burden of fitting a full quadratic form, we use a partial quadratic fitting ignoring the off-diagonal terms in the Hessian. This estimation process corresponds to the following optimization:

$$\min_{\alpha_q, \beta_q, \gamma} \sum_{i=1}^{n} \left(y^i - \alpha_q - {\beta_q}^T \Delta\mathbf{z} - \gamma^T \Delta^2 \mathbf{z}\right)^2 \Pi_{j=1}^k w\left(\frac{\mathbf{z}_j^i - \mathbf{z}_j^c}{h_q}\right), \quad (9)$$

where $\Delta\mathbf{z} = (\mathbf{z}^i - \mathbf{z}^c)$ and $\Delta^2\mathbf{z} = \Delta\mathbf{z}(\Delta\mathbf{z})^T$. In addition, h_q is an additional bandwidth for controlling a bias-variance tradeoff. In order to solve this optimization problem, we use the normal equation (Eq. 4). The i-th row in the design matrix X is set with $[1, (\Delta\mathbf{z})^T, (\Delta^2\mathbf{z})^T]$. After computing the equation, the estimated $2\hat{\gamma}_j$ is used as an estimator for $\frac{\partial^2 f}{\partial \mathbf{z}_j^2}$.

In order to perform the quadratic local regression we also need to select an additional bandwidth h_q; note that the bandwidth in this derivative estimation is different from $h\mathbf{b}_j$ used for our main linear regression that we aim to optimize. Estimating the bandwidth h_q is far more difficult than $h\mathbf{b}_j$, since it is related to estimating higher order derivatives [Ruppert and Wand 1994]. In addition, estimating second derivative information given samples can be quite noisy, compared to estimating the unknown function $f(\mathbf{z})$. Fortunately, as shown in Table I, we found that different choices for h_q for quadratic local quadratic regression have little impact on the final MSE results obtained. This can be explained by the fact that h_q is only used to compute the relative importance between each $\mathbf{b}_j$. In our optimization, the feature bandwidth $\mathbf{b}_j$ mainly estimates the relative magnitude of bandwidths based on estimated derivatives, and the global bandwidth h modulates all the magnitudes of the feature bandwidths $\mathbf{b}_j$.

Table I. rMSE of our local linear regression as a function of the bandwidth h_q used in our local quadratic fitting

h_q	0.4	0.6	0.8	1.0
San Miguel (128 spp)	0.00476	0.00462	0.00472	0.00488
killeroo-gold (32 spp)	0.00174	0.00167	0.00169	0.00176

We found that estimated derivatives can be noisy when the bandwidth h_q is too small (e.g., 0.4). Noisy derivatives then lead to noise in our estimated feature bandwidths. Fortunately since we are using the derivatives to estimate only the relative bandwidth between features, we have found it sufficient to set h_q to be reasonably large (i.e., $h_q = 1$). This choice suppresses noise in the estimated derivatives while still capturing the relative importance between features.

Alternatively, one may attempt to use an additional optimization process to estimate an optimal h_q using well-known local risk estimators such as a local cross validation [Cleveland and Loader 1996]. In practice we believe this approach is undesirable since it requires significant computational overhead while yielding only modest gains over our fixed choice of bandwidth.

Fig. 5 shows estimated optimal bandwidths for the first three features $\mathbf{z}_j$. As shown in the first row, each bandwidth is differently selected by considering estimated second derivatives. Considering our feature bandwidths gives not only visually pleasing, but also numerically accurate results, since its introduced bias is minimized thanks to choosing a small bandwidth on the dimension that has high curvatures (i.e., high second partial derivatives).

5.3 Parametric Error Estimation

Given feature bandwidths $\mathbf{b}_j$ estimated in the last section we now focus on optimizing the shared bandwidth h, in terms of minimizing MSE. Our error analysis first fits parametric curves for bias and

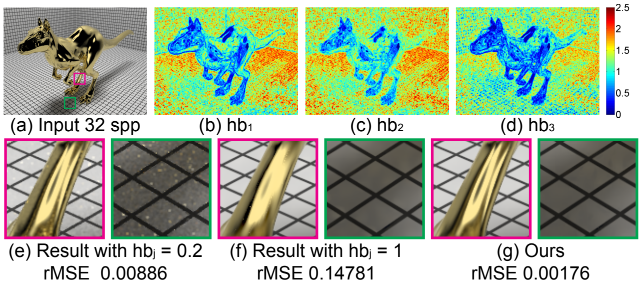


Fig. 5. Input image (a) is generated with 32 uniform spp, and estimated bandwidths (b), (c) and (d) on first three dimensions are visualized. The result images (e) and (f) are generated without our two-step optimization process, and thus we use a user-defined small bandwidth (i.e., $h\mathbf{b}_j = 0.2$) and a large bandwidth (i.e., $h\mathbf{b}_j = 1$). By using our two-step optimization, our result image (g) preserves the highlights (magenta box) thanks to small bandwidths $h\mathbf{b}_j$ on the region. In addition, our method removes the spike noise on the floor (green box) owing to large bandwidths. Model courtesy of headus/Rezard.

variance terms of MSE, as a function of the shared bandwidth parameter h, and then analytically optimizes h to minimize MSE.

One may attempt to directly use the asymptotic expressions for estimating optimal bandwidths. This approach is referred to as plug-in bandwidth selection [Cleveland and Loader 1996]. The caveat of this approach is the poor behavior with a small number of samples, because the asymptotic expression itself assumes a large number of samples. Our key idea is to utilize only functional relationships between $\mathbf{b}_j$ and second partial derivatives (Sec. 5.2), also between h and errors (Sec. 5.3).

One can easily show that our weighted local regression is a linear smoother in the output $f(\mathbf{z})$. The estimation result $\hat{f}_h(\mathbf{z})$ at $\mathbf{z}$ can be then expressed by $\hat{f}_h(\mathbf{z}) = \sum_{i=1}^{n} l_h^i(\mathbf{z})y^i$, where $l_h^i(\mathbf{z})$ is a weight for the noisy output y^i given the shared bandwidth h. In our notation, we drop the feature bandwidths $\mathbf{b}_j$ since these are given values in this stage. Based on this linear property, we can derive the bias and variance expression for the small sample count. The bias of $\hat{f}_h(\mathbf{z})$ is then reformulated as follows:

$$\begin{aligned} E\left(\hat{f}_h(\mathbf{z}) - f(\mathbf{z})\right) &= \sum_{i=1}^{n} l_h^i(\mathbf{z})E(y^i) - f(\mathbf{z}) \\ &= \sum_{i=1}^{n} l_h^i(\mathbf{z})f(\mathbf{z}^i) - f(\mathbf{z}) \ (\because \text{Eq. 1}) \\ &= \sum_{i=1}^{n} l_h^i(\mathbf{z})y^i - y. \end{aligned} \tag{10}$$

The observed values y^i and y from the pixel filter (e.g., box filter) provided by users are typically considered as unbiased estimation of the unknown $f(\mathbf{z}^i)$ and $f(\mathbf{z})$ [Rousselle et al. 2011]. As a result, the unbiased estimation for the bias term is then achieved as $\sum_{i=1}^{n} l_h^i(\mathbf{z})y^i - y$, when we plug-in y^i and y to the unknown $f(\mathbf{z}^i)$ and $f(\mathbf{z})$, respectively. The variance of our reconstruction method can be represented in a similar manner:

$$var(\hat{f}_h(\mathbf{z})) = \sum_{i=1}^{n} \left(l_h^i(\mathbf{z})\right)^2 var(y^i), \tag{11}$$

where $var(y^i)$ is the variance of the sample mean at pixel i.

As the first step for optimizing h we apply our linear local regression method with different bandwidth values for h in a range of $[h_{min}, h_{max}]$. This process provides us with a paired list of h values and corresponding bias values according to the unbiased bias estimator shown in Eq. 10; note that we do not use the asymptotic bias equation for this process.

As the second step we fit a curve based on ordinary least squares with observed pairs, to generate a parametric bias function, $bias(\hat{f}_h(\mathbf{z}))$, as a function of h. To use ordinary least squares, a proper model for $bias(\hat{f}_h(\mathbf{z}))$ should be chosen in terms of h. For the parametric bias function, we employ the functional relationship between bias and h shown in the asymptotic bias equation (Eq. 7). As a result, we use the following quadratic form:

$$bias(\hat{f}_h(\mathbf{z})) = \lambda_0 + \lambda_1 h^2. \tag{12}$$

Coefficients λ_0 and λ_1 then can be easily estimated by ordinary least squares.

To construct a parametric curve for the variance term, we also utilize the functional relationship captured in the asymptotic variance expression (Eq. 8). A polynomial model of the variance term is set as:

$$var(\hat{f}_h(\mathbf{z})) = \left(\kappa_0 + \frac{\kappa_1}{h^k}\right)\frac{1}{n(\mathbf{z})}. \tag{13}$$

The coefficients κ_0 and κ_1 are also computed using ordinary least squares.

Based on estimated two parametric curves $bias(\hat{f}_h(\mathbf{z}))$ and $var(\hat{f}_h(\mathbf{z}))$, and feature bandwidths $\mathbf{b}_j$, we can parametrize our MSE as $MSE(\hat{f}_h(\mathbf{z}))$, with respect to the shared bandwidth h. Since two parametric curves are monotonic functions as a function h, one can analytically compute the optimal bandwidth as $h_{opt} = {\frac{k\kappa_1}{4\lambda_1^2 n(\mathbf{z})}}^{\frac{1}{k+4}}$ based on $\frac{\partial MSE_h(\mathbf{z})}{\partial h} = 0$. In addition, an estimated MSE of the reconstructed image with h_{opt} is set as $MSE_{h_{opt}}(\mathbf{z})$.

Our shared bandwidth selection idea can be considered as a smoothing process based on the asymptotic expressions, and our estimated MSE is not an unbiased estimator. Introduced bias from our curve fitting is, however, asymptotically small since we use leading terms of the asymptotic expression.

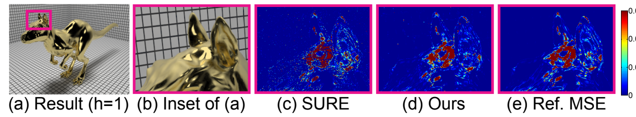
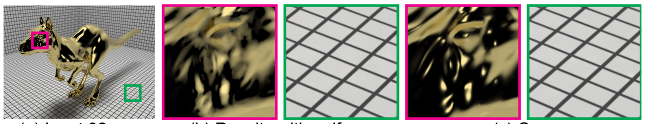


Fig. 6. Given a filtering result ((a) and (b)), our estimated MSE (d) is compared with SURE (c) and reference (e). Our estimation is less noisy compared to the result estimated by SURE, and shows a similar pattern with its reference.

We have tested the error estimation quality of our method against its ground truth MSE. To estimate the ground truth, we repeat our filtering performed with a fixed shared bandwidth, e.g., $h = 1$, on many input images (e.g., 1000 images) generated by uniformly sampled 32 rays per pixel. Fig. 6 shows our estimated MSE, Stein's unbiased risk (SURE) [Li et al. 2012], and the ground truth MSE. The error map estimated by SURE contains more noises compared to our estimation, and thus it requires an additional filtering on the map to achieve a smooth bandwidth and sampling, as demonstrated in the previous work [Li et al. 2012]. Our MSE map shows slightly under-estimated results in the head of the killeroo compared to the reference map. Our estimated error, however, shows a similar pattern compared to the ground truth, especially in highlights of the killeroo. As a result, our estimated shared bandwidth h can produce more numerically accurate results. For example, our result (Fig. 5



(a) Input 32 spp rMSE 0.02370 (b) Results with uniform weights, rMSE 0.50889 (c) Ours rMSE 0.00176

Fig. 7. Comparisons with the simple local linear regression [Bauszat et al. 2011]. While the result based on uniform weight kernels (b) preserves the checkerboard texture (green box) well, the highlights (magenta box) are completely removed. Our result (c) with a general weight kernel based on our two-step bandwidth optimization shows better results visually and numerically.

(g)) with the estimated shared bandwidth h shows a lower MSE (e.g., 0.00176) compared to the MSE (e.g., 0.00604) of the result (Fig. 6 (a)) without using the shared bandwidth (i.e., $h = 1$). Note that the two methods use our feature bandwidths $\mathbf{b}_j$.

One may prefer using a simple local regression without our two-step optimization for simplicity and efficiency, as adopted for interactive rendering [Bauszat et al. 2011]. For example, we can solve the optimization (Eq. 6) with an uniform weight kernel (i.e., $w(.) = 1$). In this case we do not need to compute optimal bandwidths, while we still use our TSVD. As shown in Fig. 7, this simpler approach preserves checkerboard textures well, since those edges are well approximated in a local linear function. It, however, fails to preserve the detailed highlights. Instead, our method with non-uniform and general kernels adapts bandwidths locally based on our estimated error, making it possible to reduce reconstruction error and thus achieve a higher quality.

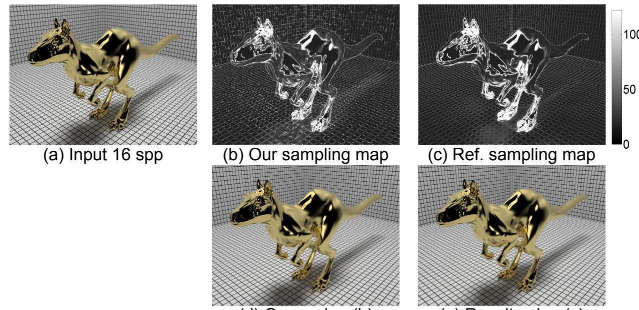
6. ADAPTIVE SAMPLING

We use a common iterative approach [Rousselle et al. 2011; Li et al. 2012] to allocate available ray samples to regions with high errors. As an initial iteration we uniformly distribute a small number of ray samples (e.g., four ray samples per pixel). In subsequent iterations we predict an error reduction $\Delta MSE(\mathbf{z})$ for a pixel $\mathbf{z}$, when the pixel would receive one additional sample. We then decide the number of samples, $\Delta n(\mathbf{z})$, according to relative values of $\Delta MSE(\mathbf{z})$.

The main difference in our method over the existing iterative sampling methods is that we use our error metric, which is dependent on the local, reduced feature subspace k, not the original feature space D.

Since we perform our reconstruction with the computed optimal shared bandwidth h_{opt} its reconstruction error is estimated as $MSE_{h_{opt}}(\mathbf{z})$, where $h_{opt} = \frac{k\kappa_1}{4\lambda_1^2 n(\mathbf{z})}^{\frac{1}{k+4}}$. One can easily show that bias and variance terms of our method with h_{opt} have the same reduction rate of $n(\mathbf{z})^{-\frac{4}{k+4}}$; deriving these terms can be done by putting h_{opt} into bias and variance terms $bias_h(\mathbf{z})$ and $var_h(\mathbf{z})$ fitted in Sec. 5.3. $MSE(\mathbf{z})$ of our reconstruction method in turn reduces in the same rate. $\Delta MSE(\mathbf{z})$ is defined as the reduction of MSE, and this term is computed as $MSE(\mathbf{z}) \times n(\mathbf{z})^{-\frac{4}{k+4}}$ by using the reduction rate.

In order to consider human visual perception that is more sensitive to darker areas we use the relative MSE [Rousselle et al. 2011], dubbed rMSE, and then $\Delta rMSE(\mathbf{z})$ is defined as the following: $\Delta rMSE(\mathbf{z}) = \frac{\Delta MSE(\mathbf{z})}{\hat{f}^2_{h_{opt}} + \epsilon}$, where ϵ is used to avoid the divide-by-zero and is set to be 0.001 in practice. We then set the number of samples $\Delta n(\mathbf{z})$ for a pixel $\mathbf{z}$, according to its relative reduction rate over the sum from all the pixels. In other words, $\Delta n(\mathbf{z}) = \frac{\Delta rMSE(\mathbf{z})}{\sum_t \Delta rMSE(\mathbf{z}^t)}$. We generate $\Delta n(\mathbf{z})$ samples for the pixel $\mathbf{z}$ by low discrepancy sampling, a common practical choice.



(a) Input 16 spp (b) Our sampling map (c) Ref. sampling map (d) Ours using (b) rMSE 0.00121 (e) Result using (c) rMSE 0.00111

Fig. 8. Given an input image (a), we adaptively allocate more ray samples until the average number of samples is 32. Our sampling map (b) and a reference sampling map (c) visualize the average sample count per pixel, and those sampling maps are computed by our sampling metric and by ground truth MSE metric, respectively. The correlation between two sampling map is 0.78. In terms of the numerical accuracy, our result using (b) is slightly worse (e.g., 9%) than the reference result using (c).

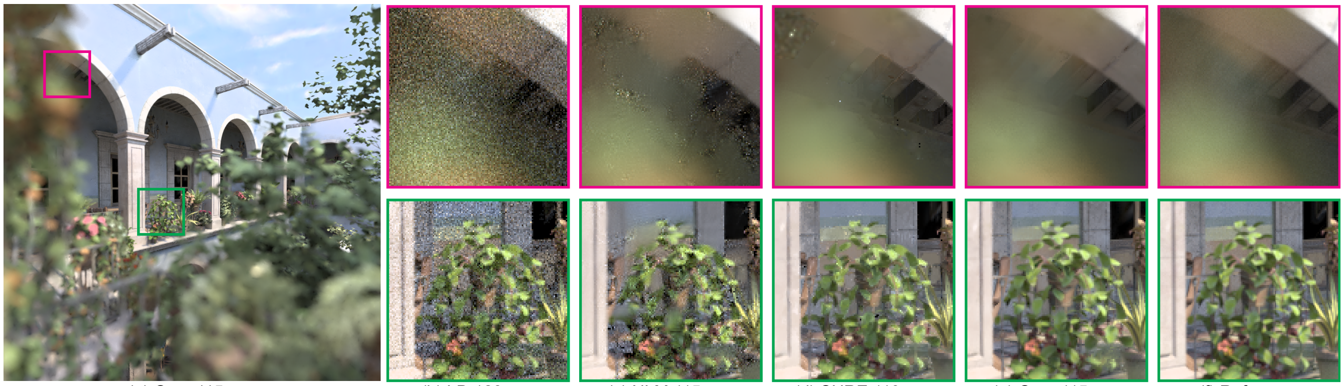
Fig. 8 shows our sampling map generated by our sampling metric, and a reference sampling map generated by the ground truth rMSE, $rMSE_g(\mathbf{z})$. $rMSE_g(\mathbf{z})$ can be computed by repeating our filtering process on randomly generated input images in N times; the exact computation is only possible with $N = \infty$. We set $N = 1000$, and the reference image generated by 16 K ray samples is used as $f(\mathbf{z})$ in the rMSE equation as: $\Delta rMSE_g(\mathbf{z}) = \frac{\Delta MSE_g}{f(\mathbf{z})^2 + \epsilon}$. ΔMSE_g is hard to be exactly computed, but is simply defined by multiplying the reduction rate $n(\mathbf{z})^{-\frac{4}{k+4}}$ to MSE_g, since we took the same approach for computing ΔMSE. We then allocate ground truth ray samples, $\Delta n(\mathbf{z})_g = \frac{\Delta rMSE_g(\mathbf{z})}{\sum_t \Delta rMSE_g(\mathbf{z}^t)}$. Our sampling map shows a high correlation, 0.78, with the reference sampling map. The filtered results in comparison with the reference sampling map are both visually and numerically similar, demonstrating the high quality approximation of our error estimation method.

7. IMPLEMENTATION DETAILS

We have implemented our method on top of pbrt2 [Pharr and Humphreys 2010], and our reconstruction is implemented by using CUDA. We accumulate each feature and color buffers in our sampling phase, and store the buffers in the texture memory of GPU for running our reconstruction. We uses a CUDA implementation of the Jacobi iterations [Nash 1975] to compute SVD.

We consider four feature types resulting in a 9 dimensional space for images: 2D coordinates, 3D normals, 1D depths, and 3D textures of primary rays. We normalize features since the range of features can be different [Sen and Darabi 2012]. Specifically, we find maximum and minimum values per feature within a filtering window, and original values are mapped linearly to the range $[0, 1]$.

We use a box filter with a small width (e.g., 0.5) for the pixel filter, which is one of the commonly used filters [Pharr and Humphreys 2010]. We use the optimal Epanechnikov kernel $w(t) = \frac{3}{4}(1 - t^2)$ for $|t| < 1$, as the kernel function in Eq. 3. Until we reach the average sample count provided by users, we use a small number of iterations (e.g., 5), where additional samples are allocated by our adaptive sampling.



(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

Fig. 9. Equal-time adaptive rendering results in the San Miguel scene. Our method shows more visually pleasing results on both defocused (the top row) and focused regions (the bottom row), and has numerically better results, about 3:1 reduction ratios on average, over NLM and SURE.

For all the tests we use a 11×11 filtering window for intermediate iterations, and use 19×19 filtering window for the final reconstruction. We test five different values for h in the range of $[h_{min}, h_{max}]$, as mentioned in Sec. 5.3. These five values are defined as follows: $[0.2 = h_{min}, 0.4, 0.6, 0.8, 1.0 = h_{max}]$.

Pixel-based local regression. We have implemented our method as a pixel-based filter which is common to many image filtering methods [Rousselle et al. 2011; 2012; Li et al. 2012]. We note, however, that our regression method can be performed directly using stored ray samples, potentially giving better results at the expense of both memory and computational overhead. Instead, we store only the mean and variance for each feature type per pixel. We use those values stored at pixels as our samples under our filtering window. As a result, our method's required memory and performance overhead is independent from the number of ray samples. When we use the GPU to perform our reconstruction method on a 1K by 1K image (e.g., San Miguel), it takes about 2 s and 7 s of processing time in intermediate stages and final stage, respectively. The main bottleneck of our method is in its various matrix operations. For complex benchmarks such as the San Miguel scene, generating 128 spp takes approximately 665 s. As a result, the overhead of our reconstruction method takes a small portion compared to the overall rendering process, and our method produces visually pleasing results much more efficiently than generating more samples, because of its effectiveness for distributing samples in high error regions. Computation overheads of different methods can vary depending on the benchmark. On average our method has a 14% and 4% overhead per sample over SURE and NLM, respectively. Nonetheless, robust error estimation and better reconstruction ability of our method results in higher efficiency in terms of reducing errors, as demonstrated in equal-time comparisons. When very detailed edges (e.g., hairy objects) need to be reconstructed, pixel-based approaches guided by geometries may miss the details especially with a relatively low sample count. As a compromise between the sample- and pixel-based approaches, one can directly apply our method to sub-pixels, where a pixel is divided into multiple sub-pixels, as demonstrated previously [Rousselle et al. 2011].

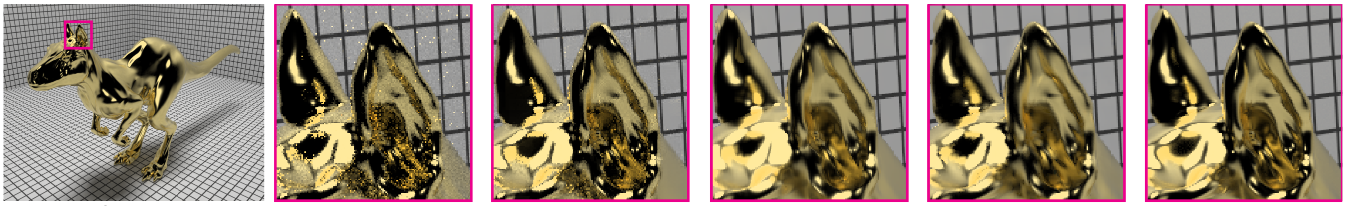
8. RESULTS AND DISCUSSIONS

For all the tests we use an Intel i7-3960X CPU machine with 3.3 GHz, and NVIDIA GeForce GTX TITAN for a CUDA implementation of our reconstruction method. We compare our method with the state of the art adaptive rendering methods [Rousselle et al. 2012; Li et al. 2012; Kalantari and Sen 2013]. For the non-local means (NLM) adaptive rendering [Rousselle et al. 2012], we use the CUDA code provided by the authors. We have implemented cross bilateral filtering based on Stein's unbiased risk estimator (SURE) as described in Li et al. [2012]. For the block-matching and 3D filtering (BM3D) based adaptive rendering [Kalantari and Sen 2013], we use the code provided by the authors. We uses the parameters recommended by authors, and the numbers are reported in our supplementary report.

As a quantitative measure for comparisons we use the relative MSE (rMSE) [Rousselle et al. 2011] that is computed as an average of $(\hat{f}_{h\mathbf{b}}(\mathbf{z}) - f(\mathbf{z}))^2/(f(\mathbf{z})^2 + 0.01)$, where 0.01 is introduced to avoid a divide by zero.

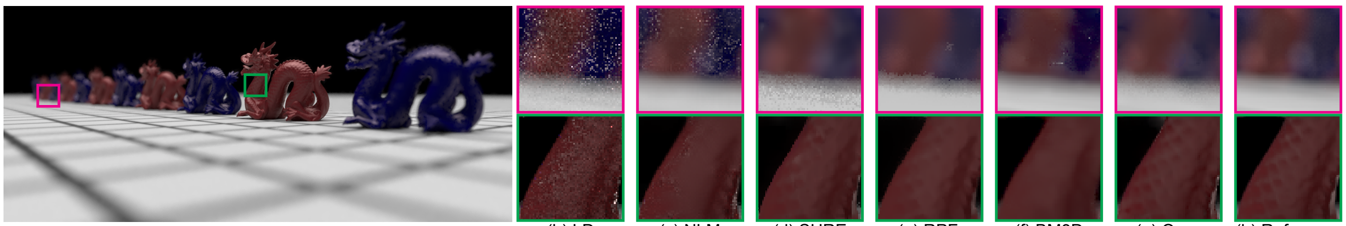
Benchmarks. To test behaviors of all the tested methods in a variety of rendering effects we have chosen the following well-known benchmarks: 1) San Miguel (1024×1024), 2) killeroo-gold (1368×1026), 3) dof-dragons (1500×636), 4) pool (1024×1024), and 5) conference scene (1024×1024). The number in each parenthesis denotes the image resolution used for each scene. The San Miguel scene has complex textured geometry, and a large portion of its image is defocused due to strong depth of field (DOF) effects. The DOF on complex geometry is a challenging benchmark for filtering methods using geometric information, because their corresponding G-buffers are quite noisy. In the killeroo-gold, the killeroo model has the highly glossy gold material that makes spike noise and strong highlights. The dof-dragons scene consists of instanced multiple dragons with complex geometry, and each dragon has very different DOF effects. In the pool scene each billiard ball with a plastic material has a different motion that leads to different motion blurs. The conference scene has glossy materials that introduce a significant amount of spike noise. All the scenes are rendered by path tracing. We have tested two more scenes, and results and comparisons with them are reported in the supplementary report; we have observed similar trends to scenes tested in the main body of the paper.

Comparisons. In the San Miguel scene (Fig. 9), NLM shows numerous high frequency artifacts on both defocused (the first row) and focused regions (the second row). NLM does not utilize G-buffers (i.e., normal, depth, and textures) and thus it runs fast. It shows, however, noisy results, when the input color information is highly noisy. On the other hand, SURE provides a relatively better filtering result in the focused region, where the G-buffers can guide some edge information, but noticeable artifacts are generated in the defocused region, where the geometry is not helpful. SURE



(a) Ours 16 spp (78 s) rMSE 0.00235 | (b) LD 32 spp (79 s) rMSE 0.02384 | (c) NLM 24 spp (79 s) rMSE 0.00417 | (d) SURE 22 spp (79 s) rMSE 0.00576 | (e) Ours 16 spp (78 s) rMSE 0.00206 | (f) Reference 16K spp

Fig. 10. Equal-time comparisons in the killeroo-gold scene. The zoom-in inset visualizes highlights on the head. NLM (c) leaves noise on the head. SURE (d) and our method (e) show visually pleasing results, but SURE gives over-blurred results.



(a) Ours 16 spp (66 s) rMSE 0.00316 | (b) LD 39 spp (66 s) rMSE 0.01350 | (c) NLM 17 spp (66 s) rMSE 0.00447 | (d) SURE 15 spp (66 s) rMSE 0.00678 | (e) RPF 16 spp rMSE 0.00547 | (f) BM3D 16 spp rMSE 0.00443 | (g) Ours 16 spp (66 s) rMSE 0.00316 | (h) Reference 64K spp

Fig. 11. Equal-time comparisons in the dof-dragons. Previous methods show either over-smoothed results or noise. Our method not only preserves the detailed geometry, but also provides smooth results on the defocused area. Dragon model courtesy of Stanford 3D repository.

attempted to ameliorate this problem by applying prefiltering using a cross bilateral filter to compute smooth bandwidths and a sampling map. However, the prefiltering fails to smooth out some pixels where the geometry is highly noisy, and spike noise remains since the energy of spike noise is not well distributed. In addition, a few null radiance pixels remained when they failed to generate more rays on the pixels. The San Miguel benchmark was also used in a multi-dimensional reconstruction method [Lehtinen et al. 2011], but the previous work only demonstrated reducing noise from distribution effects (e.g., depth-of-field). On the other hand, our method removes noise from both indirect illumination and the distribution effects simultaneously.

In the killeroo-gold scene (Fig. 10), our method preserves the strong highlights of the head and shows the lowest MSE. The NLM produces slightly noisy results and SURE generates over-blurred results.

In Fig. 11, we can verify the robustness of tested methods against DOF effects with multiple dragons that have quite different depths. Given the dof-dragons scene we use equal-sample count (e.g., 16) for RPF. RPF stores all the samples to identify sources of noise, and thus its runtime overhead is much higher compared to pixel-based approaches including ours. We also use the same sample count for BM3D, since its filtering is a MATLAB based implementation that can be slower than the CUDA implementation used in our method. For NLM and SURE, we conduct equal-time comparisons. In the defocused area (the first row), NLM and SURE show under-blurred results on the defocused dragons and textured floor. In the focused area (the second row), all the tested previous methods (NLM, SURE, RPF and BM3D) generate over-blurred results. On the other hand, our method preserves detailed curvatures thanks to our feature bandwidth selection. RPF considers the relative importance of feature types like our approach, but the main advantage of our method over RPF is error analysis based bandwidth selection and adaptive sampling. As a result, our method shows more accurate results than RPF.

We have measured convergence rates of the relative MSE in the San Miguel, killeroo-gold, dof-dragons, and pool scenes. As shown in Fig. 12, our method consistently shows lower rMSEs across all the ray sample counts over NLM and SURE. In the killeroo, the rMSE reduction of SURE is poor, although their method produces a quite smooth filtered image. Specifically, the strong highlights on the killeroo are not preserved well, which leads to a high error. Technically, this error is mainly from bias, and smaller bandwidths are required for reducing the bias. Unfortunately, the bandwidth for each feature in SURE is set as a globally fixed user parameter. One may reduce the user parameter to reduce such bias, but it typically generates more noise on other areas. In the pool scene, our method shows significantly lower rMSEs over the previous techniques. This is mainly because our method preserves noisy textures well, while removing MC noise thanks to the strong correlation between textures and output intensity, as shown in Fig. 2. Specifically, the rMSE (e.g., 0.00057) of our method with 16 spp is even lower than the rMSEs (e.g., 0.00139 and 0.00082) of NLM and SURE with 128 spp.

Extended feature vectors. The pool scene (Fig. 1) with motion blur effects is a challenging benchmark for filtering methods guided by the geometry. For example, geometry buffers can be very noisy on motion blurred regions, and it leads to over-blurred results on the noisy textures. Since additional features can be freely included for our method, we use two different textures buffers, each of which is dedicated for moving or non-moving objects, to have separate texture information for these different types of objects. Specifically, we check intersection points between primary rays and the scene, and then determine whether the points are from moving objects or not. In the case of moving objects, the texture buffer for moving objects is updated based on the texture information of intersected moving objects. For the other case of non-moving objects, the texture buffer for non-moving objects is updated.

As shown in the Fig. 1, our method with this simple extension generates a high quality reconstruction result on the motion blurred region with noisy textures (the second row). On the other hand, the previous methods (NLM, SURE, and BM3D) either leave high frequency noises or show over blurred results even in the noisy texture on the floor (the first row). One may reduce the bias of SURE by reducing the bandwidth for textures, but it typically increases noise

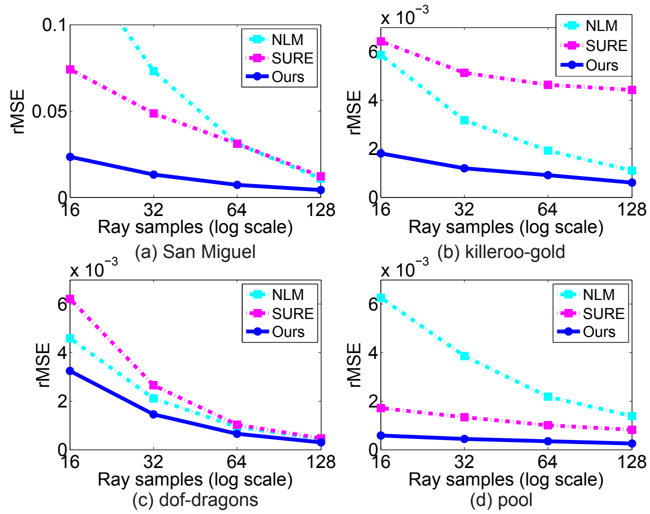
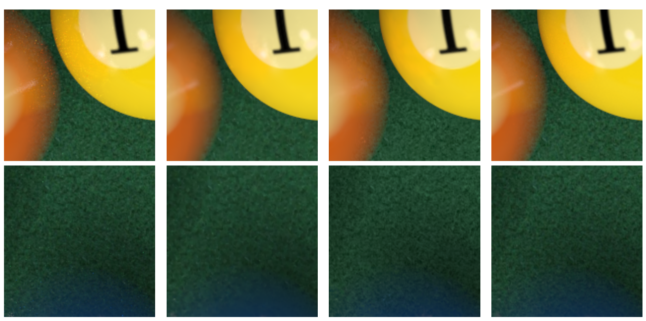


Fig. 12. Convergence plots in different methods. Our method outperforms previous methods across all the tested sample counts.



(a) NLM 512 spp (1064 s) rMSE 0.00047 | (b) SURE 1K spp (2202 s) rMSE 0.00063 | (c) Ours 32 spp (91 s) rMSE 0.00044 | (d) Reference 64K spp

Fig. 13. Equal-quality comparisons in the pool scene.

on the motion blurred textures of the blue ball (second row). In addition, it is clear that the noise texture in the first row cannot be preserved by using the additional texture buffer. Our local linear regression gives much better results on the noisy textures since the non-linear intensity function has a strong linear correlation on texture space, as shown in Fig. 2. Specifically, our method shows 9×, 3×, and 8× rMSE reduction ratios over NLM, SURE, and BM3D, respectively.

SURE can use the additional texture buffer for the moving objects with a user-defined, global standard deviation parameter. For two buffers we use the same parameter (i.e., 0.25) used in the original texture buffer of SURE. Unfortunately, we found that this extension of SURE provides a numerically similar result (e.g., rMSE = 0.00129) to the one without using the extended buffer.

The extended feature is a naïve one and does not fully address more complex scenarios. For example, even within a single pixel, multiple moving objects with different speeds can exist. Note that our result with the extended buffer is provided for verifying whether our method supports additional features without introducing the bandwidth selection problem for the features.

Equal-quality comparisons. We conduct equal-quality comparisons in the pool scene (Fig. 13). For our method, we use the extended feature buffer as described in the previous paragraph. NLM with 512 ray samples shows a similar MSE compared to ours with 32 ray samples. Even with the large number of ray samples, NLM leaves high-frequency noise especially on the orange ball (first row). Since NLM does not utilize geometry information, it is fundamentally difficult in certain regions to discern image features (e.g., noisy texture) from MC noise. SURE shows over-blurred results on the noisy texture (second row). Although SURE uses geometries, automatic bandwidth selection for the geometries is not resolved. One may reduce the bandwidths used in SURE manually, but it can generate noise in motion blurred regions. Because of the fixed, global bandwidth for geometries, SURE does not preserve detailed noisy textures. As a result, SURE with 1024 ray samples shows a higher MSE compared to ours. In this challenging benchmark, our method achieves more than an order of magnitude performance improvement over the tested state-of-the-art methods.

Table II. rMSE comparisons of animated results

San Miguel	frame 50	frame 100	frame 150	frame 200
LD (143 spp)	0.05433	0.05127	0.04894	0.04934
NLM (128 spp)	0.00814	0.00845	0.00843	0.00962
SURE (126 spp)	0.01228	0.01448	0.01402	0.01548
Ours (128 spp)	0.00525	0.00546	0.00528	0.00556

Outliers. In rendering, outliers are often defined as spike noise with extremely high energy. If outliers are not addressed well, sampling budgets are drained because pixels with outliers typically have a very large variance [Rousselle et al. 2011]. Furthermore, a reconstruction process requires a larger window size to spread out its energy [Rousselle et al. 2012]. We use a simple heuristic to suppress its influence, by dividing the computed weights of samples by the variance of the sample mean. This approach works with our local regression approach and has a negligible overhead. Nonetheless we have observed that this simple trick produces visually pleasing results. This is mainly because our method robustly measures bias even with outliers and our reconstruction method restores energy loss of outliers, as the number of samples is increased. Note that high bias values for pixels with outliers cause those pixels to receive appropriate samples and thus the variance of the sample mean tends to decrease. As a result, our reconstruction method puts higher weights on outliers, reducing the energy loss.

Animations. Our method can naturally be extended to handle animated images, since our technique selects bandwidths across an arbitrary set of features including ones derived from the temporal domain. To demonstrate our extension to the temporal domain, we first generate each frame using our adaptive rendering method, and these images can be considered as volumetric pixels. The animation directly computed from the volumetric pixels can have many flickering artifacts unless enough samples are used for each frame. To further reduce noise by using temporal coherence, we apply our method again as a post-process to the volumetric pixels. Specifically, in each volumetric pixel, we apply our reconstruction with an additional feature, a frame number t. The input variance of each volumetric pixel is estimated using the variance estimation equation (Eq. 11). In our implementation, we use a $7 \times 7 \times 5$ filtering window to utilize both temporal and spatial coherence. Given the filtering window size, the runtime overhead of our additional filtering per frame takes 7 s for the San Miguel benchmark. Note that all the bandwidths of features including the frame number t is automatically selected in terms of minimizing MSE.

In our accompanying video, we conduct the equal-time comparisons with SURE and NLM. For the temporal extension of SURE, we add an additional Gaussian function to the cross bilateral filter, and its distance function is defined based on the difference on frame numbers of pixels; details are included in our supplementary report. For NLM, we extended their non-local means filtering to consider neighboring pixels in adjacent frames (e.g., 5 frames), as described in their paper.

Our method shows visually pleasing animation results across all the tested models. In some cases, our result still shows noticeable flickering, but its level is significantly lower than those of other tested methods. In addition, we measure rMSEs from a few regularly spaced frames in the accompanied video, based on reference images rendered by 8 K samples per pixel. As shown in Table II, our method produces numerically more accurate filtered images compared to all the other tested methods; rMSEs in other scenes are included in our supplementary report.

One may prefer to apply our method directly to the volumetric pixels. We found that it produces a slightly lower rMSE compared to the current approach, since working directly with the volumetric pixels does not produce any additional bias. Nonetheless, we chose to use additional filtering that reapplies our reconstruction to the images filtered by our method, mainly because the additional filtering tends to have less flickering compared to the alternative. Overall, the additional filtering generates an additional bias, but as shown in Fig. 14, the introduced bias from additional filtering is not high compared to the amount of reduced variances by the same filtering. This is mainly because our additional filtering method also optimizes for the reduction of MSE.

Limitations. As shown in the conference scene (Fig. 15), all the tested methods do not provide satisfactory results because of the frequent presence of spike noise. Our method is visually better than the other methods, but still exhibits low-frequency noise. We may lessen the problem by using a larger window size [Rousselle et al. 2012], but this approach can introduce more bias in other scenes. Before applying our method, outliers can be reduced in a pre-processing step [DeCoro et al. 2010]. Estimating bias introduced in the pre-processing step, however, is not trivial. Efficiently addressing outliers without energy loss (i.e., bias) still remains a challenging problem in adaptive rendering techniques. In addition, our error analysis is derived by assuming that input samples satisfy the independence property. When low discrepancy samples are used, our method can overestimate variances of filtered images. The overestimation can lead to over-blurring in our reconstruction and overshooting of our sampling budgets, but we found that it does not significantly lower the quality of our reconstruction; our supplementary material includes more details. Furthermore, our TSVD based pre-filtering for reducing noise contained in feature vectors can lead to over-blurred results of detailed edges in the regions with very noisy geometries (e.g., Fig. 1 (f)), since local dimensions estimated by the TSVD can be underestimated especially with a small number of ray counts. Technically, the loss in details goes to zero as the number of samples increases, since our threshold value τ used in TSVD also goes to zero. Finally, our filtering tends to generate over-blurring results in some regions (e.g., ears of the killeroo in Fig. 10) when output intensity is nonlinear on features and also variances of the regions are very high. Nonetheless, our method shows visually pleasing and numerically better results over the state-of-the-art adaptive rendering methods.

9. CONCLUSION

We have proposed a novel, weighted local regression based adaptive rendering technique for efficiently and robustly handling a wide variety of rendering effects. We locally identify noisy features based on our TSVD and perform our reconstruction with a reduced feature space. We then parameterize the bias and variance of our reconstruction error and find the optimal bandwidths for feature types. We also adopt an iterative sampling process that distributes the available ray budgets to regions with high errors according to our error metric. We have also demonstrated robustness and efficiency of our method with a diverse set of benchmark models under different rendering effects, and showed that our method produces a robust and consistent improvement over the state-of-the-art techniques.

(a) Ours 128 spp w/ addi. filtering rMSE 0.00546 (b) Ours 128 spp w/o addi. filtering rMSE 0.00555 (c) Ours 128 spp w/ addi. filtering rMSE 0.00546 (d) Reference 8K spp

Fig. 14. Our results with and without running an additional filtering that utilizes temporal coherence, given the 100th frame from the accompanying video of the San Miguel.

Many interesting future research directions lie ahead. Robustly handling outliers in the context of rendering continues to be a challenging problem. We plan to investigate a number of different approaches (e.g., spreading out the loss energy into the reconstructed function) within our adaptive rendering framework in a robust way. We would like to extend our TSVD based pre-filtering to be performed by analyzing the source of noise contained in feature vectors such as noisy textures or motion blur, since it may lead to a higher reconstruction result, when the noise comes from multiple sources simultaneously. Another interesting question is to investigate fundamental differences and efficiency between our local regression based image-space adaptive method and photon mapping, since the density estimation and our local regression stem from the same statistical assumptions. For example, Kaplanyan and Dachsbacher [2013] demonstrated that high quality reconstruction can be achieved in a progressive photon mapping framework by using local regression. Finally we would like to investigate our bandwidth selection for a wider set of features (e.g., virtual flash images proposed in [Moon et al. 2013]).

ACKNOWLEDGMENTS

We are thankful to Soheil Darabi, Min H. Kim, and the anonymous reviewers for their valuable feedback. We would like to thank Fabrice Rousselle for the discussion on the 3D extension of NLM. This work was supported in part by NRF-2013R1A1A2058052, DAPA/ADD (UD110006MD), MEST/NRF (2013-067321), IT R&D program of MOTIE/KEIT [10044970], as well as Adobe. Sung-Eui Yoon is a corresponding author of the paper.

REFERENCES

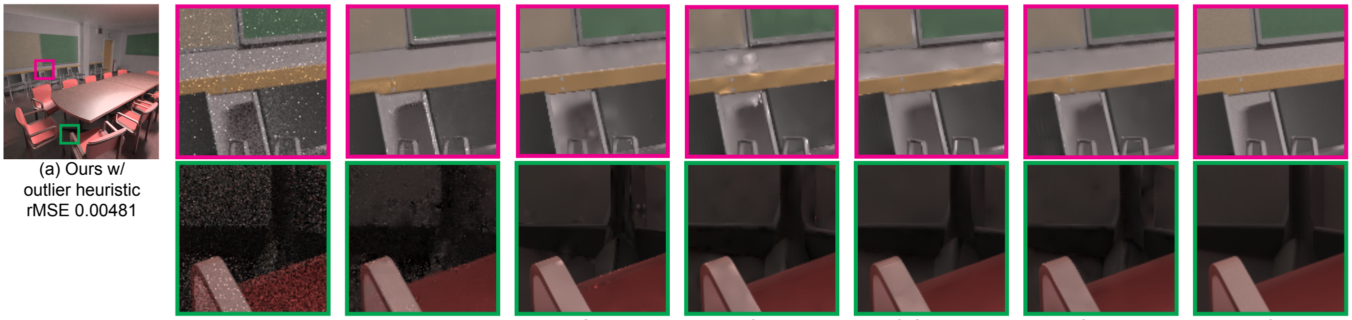
BAUSZAT, P., EISEMANN, M., AND MAGNOR, M. 2011. Guided image filtering for interactive high-quality global illumination. *Computer Graphics Forum 30,* 4, 1361–1368.

BELCOUR, L., SOLER, C., SUBR, K., HOLZSCHUCH, N., AND DURAND, F. 2013. 5d covariance tracing for efficient defocus and motion blur. *ACM Trans. Graph. 32,* 3, 31:1–31:18.

CLEVELAND, W. S. AND LOADER, C. L. 1996. Smoothing by Local Regression: Principles and Methods. In *Statistical Theory and Computational Aspects of Smoothing*. Springer, New York, 10–49.

DAMMERTZ, H., SEWTZ, D., HANIKA, J., AND LENSCH, H. P. A. 2010. Edge-avoiding A-Trous wavelet transform for fast global illumination filtering. In *High Performance Graphics*. 67–75.

DECORO, C., WEYRICH, T., AND RUSINKIEWICZ, S. 2010. Density-based outlier rejection in Monte Carlo rendering. *Computer Graphics Forum 29,* 7, 2119–2125.



(a) Ours w/ outlier heuristic rMSE 0.00481

(b) LD rMSE 0.65747 (c) NLM rMSE 0.00708 (d) SURE rMSE 0.00655 (e) Ours w/o outlier heuristic rMSE 0.01151 (f) Ours w/ a large window rMSE 0.00497 (g) Ours w/ outlier heuristic rMSE 0.00481 (h) Reference 64K spp

Fig. 15. Failure case of the conference room. This scene contains a large number of outliers (i.e., spike noise). All images are rendered with 32 samples per pixel. NLM (c) uses a large filtering window (41×41) than its original parameters (21×21). Our method without a simple heuristic (e) does not distribute the energy of outliers (first row). The artifacts can be reduced with a large filtering window (31×31) (f), but it can introduce more bias in other scenes. Our method with a heuristic (g) shows better results than all the tested methods, without changing our filtering window size. Model courtesy of Anat Grynberg and Greg Ward.

EGAN, K., DURAND, F., AND RAMAMOORTHI, R. 2011. Practical filtering for efficient ray-traced directional occlusion. *ACM Trans. Graph. 30,* 6, 180:1–180:10.

EGAN, K., HECHT, F., DURAND, F., AND RAMAMOORTHI, R. 2011. Frequency analysis and sheared filtering for shadow light fields of complex occluders. *ACM Trans. Graph. 30,* 2, 9:1–9:13.

EGAN, K., TSENG, Y.-T., HOLZSCHUCH, N., DURAND, F., AND RAMAMOORTHI, R. 2009. Frequency analysis and sheared reconstruction for rendering motion blur. *ACM Trans. Graph. 28,* 3 (July), 93:1–93:13.

HACHISUKA, T., JAROSZ, W., AND JENSEN, H. W. 2010. A progressive error estimation framework for photon density estimation. *ACM Trans. Graph. 29,* 6, 144:1–144:12.

HACHISUKA, T., JAROSZ, W., WEISTROFFER, R. P., DALE, K., HUMPHREYS, G., ZWICKER, M., AND JENSEN, H. W. 2008. Multidimensional adaptive sampling and reconstruction for ray tracing. In *ACM SIGGRAPH*. 33:1–33:10.

HANSEN, P. 1987. The truncated svd as a method for regularization. *BIT Numerical Mathematics 27,* 4, 534–553.

HE, K., SUN, J., AND TANG, X. 2013. Guided image filtering. *IEEE Trans. Pattern Anal. Mach. Intell. 35,* 6, 1397–1409.

KALANTARI, N. K. AND SEN, P. 2013. Removing the noise in Monte Carlo rendering with general image denoising algorithms. *Computer Graphics Forum (Proceedings of Eurographics 2013) 32,* 2, 93–102.

KAPLANYAN, A. S. AND DACHSBACHER, C. 2013. Adaptive progressive photon mapping. *ACM Trans. Graph. 32,* 2, 16:1–16:13.

LEHTINEN, J., AILA, T., CHEN, J., LAINE, S., AND DURAND, F. 2011. Temporal light field reconstruction for rendering distribution effects. *ACM Trans. Graph. 30,* 4, 55:1–55:12.

LEHTINEN, J., AILA, T., LAINE, S., AND DURAND, F. 2012. Reconstructing the indirect light field for global illumination. *ACM Trans. Graph. 31,* 4, 51:1–51:10.

LI, T.-M., WU, Y.-T., AND CHUANG, Y.-Y. 2012. Sure-based optimization for adaptive sampling and reconstruction. *ACM Trans. Graph. 31,* 6 (Nov.), 194:1–194:9.

MCCOOL, M. D. 1999. Anisotropic diffusion for Monte Carlo noise reduction. *ACM Trans. Graph. 18,* 2, 171–194.

MEYER, M. AND ANDERSON, J. 2006. Statistical acceleration for animated global illumination. *ACM Transactions on Graphics (TOG) 25,* 3, 1075–1080.

MITCHELL, D. P. 1987. Generating antialiased images at low sampling densities. In *ACM SIGGRAPH*. Vol. 21. 65–72.

MOON, B., JUN, J. Y., LEE, J., KIM, K., HACHISUKA, T., AND YOON, S.-E. 2013. Robust image denoising using a virtual flash image for monte carlo ray tracing. *Computer Graphics Forum 32,* 1, 139–151.

NASH, J. C. 1975. A one-sided transformation method for the singular value decomposition and algebraic eigenproblem. *The Computer Journal 18,* 1, 74–76.

OVERBECK, R. S., DONNER, C., AND RAMAMOORTHI, R. 2009. Adaptive wavelet rendering. In *ACM SIGGRAPH Asia 09*. 140:1–140:12.

PHARR, M. AND HUMPHREYS, G. 2010. *Physically Based Rendering: From Theory to Implementation 2nd*. Morgan Kaufmann Publishers Inc.

RITSCHEL, T., ENGELHARDT, T., GROSCH, T., SEIDEL, H.-P., KAUTZ, J., AND DACHSBACHER, C. 2009. Micro-rendering for scalable, parallel final gathering. In *ACM SIGGRAPH Asia*. 1–8.

ROUSSELLE, F., KNAUS, C., AND ZWICKER, M. 2011. Adaptive sampling and reconstruction using greedy error minimization. In *SIGGRAPH Asia*. 159:1–159:12.

ROUSSELLE, F., KNAUS, C., AND ZWICKER, M. 2012. Adaptive rendering with non-local means filtering. *ACM Trans. Graph. 31,* 6, 195:1–195:11.

RUPPERT, D. AND WAND, M. 1994. Multivariate locally weighted least squares regression. *The annals of statistics 22,* 3, 1346–1370.

RUSHMEIER, H. E. AND WARD, G. J. 1994. Energy preserving non-linear filters. In *ACM SIGGRAPH*. 131–138.

SCHWARZHAUPT, J., JENSEN, H. W., AND JAROSZ, W. 2012. Practical hessian-based error control for irradiance caching. *ACM Trans. Graph. 31,* 6, 193:1–193:10.

SEGOVIA, B., IEHL, J. C., MITANCHEY, R., AND PÉROCHE, B. 2006. Non-interleaved deferred shading of interleaved sample patterns. In *Graphics Hardware*. 53–60.

SEN, P. AND DARABI, S. 2012. On filtering the noise from the random parameters in monte carlo rendering. *ACM Trans. Graph. 31,* 3, 18:1–18:15.

SHIRLEY, P., AILA, T., COHEN, J., ENDERTON, E., LAINE, S., LUEBKE, D., AND MCGUIRE, M. 2011. A local image reconstruction algorithm for stochastic rendering. In *I3D*. 9–14.

SOLER, C., SUBR, K., DURAND, F., HOLZSCHUCH, N., AND SILLION, F. 2009. Fourier depth of field. *ACM Trans. Graph. 28,* 2, 18:1–18:12.

STEWART, G. 1993. On the early history of the singular value decomposition. *SIAM review 35,* 4, 551–566.