Multidimensional Adaptive Sampling and Reconstruction for Ray Tracing

Toshiya Hachisuka* Wojciech Jarosz* Richard Peter Weistroffer† Kevin Dale‡
Greg Humphreys† Matthias Zwicker* Henrik Wann Jensen*

*UC San Diego ‡Harvard University †University of Virginia

Abstract

We present a new adaptive sampling strategy for ray tracing. Our technique is specifically designed to handle multidimensional sample domains, and it is well suited for efficiently generating images with effects such as soft shadows, motion blur, and depth of field. These effects are problematic for existing image based adaptive sampling techniques as they operate on pixels, which are possibly noisy results of a Monte Carlo ray tracing process. Our sampling technique operates on samples in the multidimensional space given by the rendering equation and as a consequence the value of each sample is noise-free. Our algorithm consists of two passes. In the first pass we adaptively generate samples in the multidimensional space, focusing on regions where the local contrast between samples is high. In the second pass we reconstruct the image by integrating the multidimensional function along all but the image dimensions. We perform a high quality anisotropic reconstruction by determining the extent of each sample in the multidimensional space using a structure tensor. We demonstrate our method on scenes with a 3 to 5 dimensional space, including soft shadows, motion blur, and depth of field. The results show that our method uses fewer samples than Mittchell's adaptive sampling technique while producing images with less noise.

CR Categories: I.3.7 [Computing Methodologies]: Computer Graphics—Three-Dimensional Graphics and Realism

Keywords: Sampling and Reconstruction, Ray Tracing, Rendering Equation, Global Illumination

1 Introduction

Most image synthesis applications rely heavily on sampling and reconstruction. Errors introduced by these processes can be visually distracting, so a large number of samples are typically drawn. However, there is a tension between computational expense and image fidelity, as computing the radiance at a sample can be costly. Adaptive sampling techniques strive to focus effort only where it is likely to be needed: in areas of rapid change. This approach is justified because regions where the signal is changing quickly have high local bandwidth; therefore, more samples are needed in these regions to adequately capture the higher frequencies (see Landau and Pollak for a formal analysis of the use of local bandwidth in sampling [Landau and Pollak 1962]).

Typically, adaptive techniques detect high frequencies of the radiance function only in the image plane; however, adaptive sampling only in the image plane poses problems in the context of Monte Carlo ray tracing. These approaches ignore discontinuities present between samples that fall into the same pixel, where additional samples would be of great benefit. Furthermore, image-space algorithms are often difficult to generalize to ones that consider features of the input signal in non-image dimensions (e.g., time for motion blur). Although this does not make them incompatible with a Monte Carlo ray tracing, the resulting images require more samples to prevent artifacts in the reconstruction.

In this paper, we present a rendering algorithm based on multidimensional adaptive sampling and integration. Our technique adaptively distributes a set of samples in the full, multidimensional sampling domain. These samples will tend to group around areas of rapid change in all dimensions, which could be either sharp edges in the scene, or image areas that are dominated by distribution effects (e.g., soft shadows, motion blur, depth of field) where more samples are required to adequately reconstruct the correct color for display. We also present a novel image reconstruction technique designed for these multidimensional samples. Our approach integrates a piecewise constant function defined by samples to project the multidimensional result onto the image plane. Our combined approach allows us to render realistic images with substantially less visual and quantitative error than previous approaches in the same amount of time.

2 Related Work

Sampling and reconstruction are ubiquitous in computer graphics, and we will only describe the most relevant prior work here. For a more comprehensive overview we refer to several excellent surveys that describe the different algorithms [Mitchell 1990; Glassner 1995; Dutré et al. 2006; Pharr and Humphreys 2004].

Most adaptive sampling algorithms operate exclusively in the image plane. Whitted's pioneering paper on recursive ray tracing suggested the use of an adaptive sampling scheme, followed by a reconstruction step [Whitted 1980]. Whitted's algorithm first coarsely samples the image on a regular grid, and then recursively subdivides the squares of this grid if the samples at the corners of the square are sufficiently different. Although Whitted's scheme is adaptive, it is not stochastic, and the rendered images exhibit structured aliasing. To eliminate aliasing a number of approaches use adaptive stochastic sampling [Mitchell 1987; Mitchell 1991; Bolin and Meyer 1998; Rigau et al. 2003], where the samples are generated stochastically with higher density in regions of the image that exhibit the largest error (e.g., high contrast or variance). These image based approaches provide a powerful method for reducing the number of samples required by normal ray tracing, but they fall short when considering effects such as motion blur and depth of field. The reason for this is that pixel values are noisy unless a large number of samples are used to estimate the true radiance through each pixel. Mitchell [1991] demonstrated how to compute better sampling patterns for higher dimensional problems, but the adaptive part of his algorithm still used the samples in the image plane.

In his seminal paper, Kajiya [1986] described several approaches for using adaptive and hierarchical sampling for evaluating the rendering equation. He also proposed the use of a k-d tree for sample placement when performing numerical estimation of high dimen-

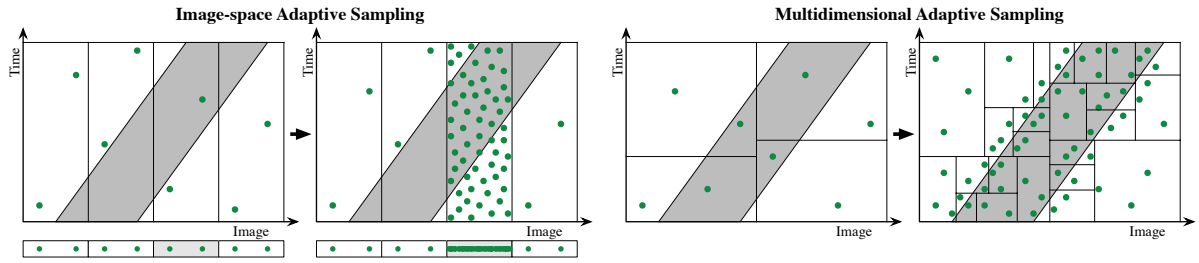

Figure 1: *A 2-D example defined by a 1-D image and time. Image space adaptive sampling approaches (left) collapse the multidimensional function onto the image before detecting discontinuities. However, nearby image samples my span large regions in the remaining dimensions, easily missing important features such as fast moving objects. Moreover, pixels chosen for supersampling have additional samples distributed uniformly within the multidimensional domain. Our approach (right) detects discontinuities in the multidimensional domain, adapting to the behavior of the full multidimensional function.*

sional integrals. However, he ended up not using this approach since his "experiments in finding adaptive criteria have not been terribly successful." Our method builds on Kajiya's work and we show how to make this idea of multidimensional sampling and reconstruction practical for rendering.

In the context of Monte Carlo ray tracing [Cook et al. 1984] there are numerous techniques that use importance sampling in which knowledge of the reflectance functions or lighting is incorporated into the sampling scheme (see Dutre et al. [2006] for an overview). Recent work includes BRDF importance sampling [Lawrence et al. 2004], and BRDF and environment map sampling [Clarberg et al. 2005]. Our method can be used in combination with many of these techniques and we already use BRDF importance sampling in our system. One caveat to using adaptive sampling compared with pure importance sampling methods is that we introduce bias in the final rendered images [Kirk and Arvo 1991]. In practice this means that errors due to insufficient sampling appear as artifacts in the image such as blurry patches rather than the noise seen in pure Monte Carlo ray tracing based approaches. This is similar to other biased approaches such as photon mapping [Jensen 2001] and irradiance caching [Ward et al. 1988].

Metropolis light transport (MLT) [Veach and Guibas 1997] is a Monte Carlo ray tracing algorithm that uses the history of the previous ray to sample the bright regions of the scene more densely. MTL can be thought of as an adaptive sampling technique within the multidimensional sample space [Kelemen and Szirmay-Kalos 2001]. One key difference between our sampler and MLT is that we use the entire sample history to generate new samples, which means that our algorithm is better at detecting multiple disjoint regions of importance. Furthermore, MLT does not provide any advantages when it comes to simulating blurry effects such as motion blur and depth of field. Our sampler adapts based on contrast and it naturally detects edges in (for example) the space-time domain in scenes with motion blur.

Multidimensional lightcuts [Walter et al. 2006] is a powerful method for accelerating lighting calculation in the presence of motion blur, depth of field, participating media and more. It works by representing the light transport within the scene by a large number of *light* and *gather* points. The performance gain in multidimensional lightcuts comes from efficiently culling the lighting computations between light and gather points. Unfortunately, the generation of the full set of sample points is still computed in a brute-force fashion, which may be too costly if a scene requires a large collection of samples (e.g., due to significant motion blur or depth of field). In contrast, our sampler does not cull samples, but instead reduces the total number of generated samples. Although the current implementation of our sampler does not support indirect illumination, it is possible to combine our method with other global illumination methods such as photon mapping.

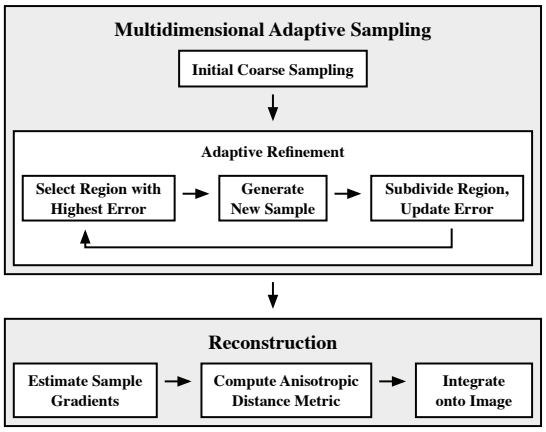

Figure 2: *Our algorithm is split up into two main stages: multidimensional adaptive sampling (described in Section 4) followed by anisotropic reconstruction and integration (described in Section 5).*

Outside computer graphics, the CUHRE technique is a numerical integration algorithm introduced by Berntsen et al. [1991]. They adaptively perform axis-aligned subdivision of a hypercubical integration domain to find a good set of evaluation locations for numerical integration. While we use a similar technique, our sampling is ultimately guided towards image synthesis by using contrast and we use a reconstruction step that is optimized for generating two dimensional images.

The MISER [Press and Farrar 1990] method is closely related to our approach and has been successfully used in the context of rendering [Leeson 2003]. In essence, the MISER method is a stratified Monte Carlo integration technique where the integration domain is adaptively subdivided in order to reduce the overall variance. Evaluation of each subdomain is performed using a standard Monte Carlo method. Though we also use adaptive Monte Carlo sampling, our method employs a deterministic integration technique combined with an anisotropic reconstruction to evaluate the subdomains.

Most adaptive ray tracing algorithms face the problem of reconstructing the image for a non-uniform distribution of samples. This is challenging, as most computer graphics scenes contain discontinuities that require a dense sampling to reconstruct. Whitted's sampling technique [Whitted 1980] reconstructed an image by summing the average color of each recursively generated square, weighted by that square's proportional area. This scheme is easy to implement, but it can assign very different weights to sample points that are close together. Mitchell identified this problem and improved this

technique by using the areas of the Voronoi cell around each sample as the filter weights [Mitchell 1990]. Although using Voronoi cells is impractical, this suggestion directly motivated the reconstruction ideas described in this paper. Mitchell also proposed a new technique for reconstructing images from sample points with spatially varying density properties [Mitchell 1987]. He applied multiple box filters at varying resolutions, yielding good results as long as the variation in density is not too large. These multi-stage techniques are hard to generalize to higher-dimensional sample sets unless we can assume that the non-image dimensions are uncorrelated with the image itself. While this assumption is true of traditional adaptive sampling techniques, our proposed algorithm does not have this property, so a new reconstruction algorithm is required.

3 Overview

Image formation for raytracing evaluates the radiance $L(x, y)$ arriving at a set of discrete sample points on the image plane. In the absence of any distribution effects, each image-space coordinate (x, y) uniquely determines a light path, allowing L to be precisely evaluated by tracing a single ray. Traditional adaptive sampling techniques attempt to carefully choose the samples by detecting areas of interest in the function L (e.g., high local contrast or brightness) based on local image-space behavior. Assuming a reasonable initial coarse sampling, this process works well because L is easy to evaluate exactly for any given image location.

Physically-based light transport, however, includes effects such as motion blur, depth-of-field, and soft shadows. When these effects are considered, there are many light paths that can contribute incoming radiance to a given image location. These paths can be interpreted as points in a higher dimensional domain, giving a location on the aperture, in time, etc. For any class of light paths, the function $L(x, y)$ is itself an integral:

$$L(x,y) = \int \dots \int f(x,y,u_1,\dots,u_n)\; du_1 \dots du_n, \qquad (1)$$

where the dimensions $u_1, \dots, u_n$ define the class of light paths, and f is the contribution of each path.

Monte Carlo ray tracing samples $f(x, y, u_1, \dots, u_n)$ and numerically integrates these samples to approximate the desired image-space function $L(x, y)$. This approximation is often very noisy, which hampers image-space adaptive sampling techniques because features and noise are indistinguishable during adaptive sample refinement. Furthermore, image-space metrics inherently fail to maintain enough information to properly capture the discontinuities of the true multidimensional function. Instead, they consider only the radiance of nearby samples on the image plane, which may actually represent a large region in the remaining dimensions. If neighboring image samples do not provide a good estimate of the integral in Equation 1, the renderer may miss important features. This limitation leads to sub-optimal sampling of the multidimensional domain (Figure 1).

Our approach performs adaptive sampling by placing samples near areas of high frequency in the full multidimensional domain, allowing it to more robustly detect and sample such features. We then use these samples to create an analytic approximation of the high dimensional function. After sampling, the image is generated by integrating this function over all non-image dimensions.

Conceptually, we perform this integration by splitting up the sampling domain into disjoint subdomains. For each continuous coordinate (x, y) on the image plane we construct a partition $P(x, y)$ of the remaining n dimensions $u_1 \dots u_n$. Given this decomposition, the radiance at (x, y) can be computed as

$$\begin{aligned} L(x,y) &= \int \dots \int f(x,y,u_1,\dots,u_n)\; du_1 \dots du_n \\ &= \sum_{\Omega \in P(x,y)} \int_{\Omega} f(x,y,u_1,\dots,u_n)\; du_1 \dots du_n. \end{aligned} \qquad (2)$$

By assuming that the function value within each subdomain is constant, we can estimate the integral with a simple product

$$\int_{\Omega} f(x,y,u_1,\dots,u_n)\; du_1 \dots du_n \approx V_\Omega f_\Omega, \qquad (3)$$

where V_Ω is the volume of domain Ω and f_Ω is the constant reconstruction of the function within Ω. The radiance on the image plane can then be computed as a Riemann sum of these estimates

$$L(x,y) \approx \sum_{\Omega \in P(x,y)} V_\Omega f_\Omega. \qquad (4)$$

Note that we did not explicitly define subdomains and constant reconstructions here. We will discuss more details in Section 5.

Our algorithm is split up into two main steps: *adaptively sampling* the multidimensional function f, and analytically integrating the resulting approximation to *reconstruct* the image radiance L (Figure 2). The number and distribution of subdomains Ω for each pixel is determined during sampling, and the function approximations f_Ω are calculated during reconstruction. The goal of our algorithm is to distribute the samples such that the approximation in Equation 4 is as close as possible to the true solution while using as few samples as possible.

4 Multidimensional Adaptive Sampling

The guiding principle of any adaptive sampling algorithm is to place samples in areas of high local bandwidth while not spending too much time making decisions about sample locations. We have developed a multidimensional version of Mitchell's contrast function [Mitchell 1987] that we use to guide our adaptive sample technique.

Algorithm Overview. Our approach starts by placing an initial coarsely distributed set of samples into an acceleration structure for fast neighbor searching. We then use a greedy algorithm to repeatedly add samples in regions with high local error. We illustrate our algorithm in Figure 3 and explain each step in the following sections.

4.1 Initial Sampling

In this step we generate an initial coarse sampling by sparsely distributing sample points within the entire multidimensional domain (Figure 3a). We use a uniform random sample distribution in all dimensions, though any technique, such as a quasi-random sequence or poisson disk points, could be used instead. Each sample stores its multidimensional coordinate $\mathbf{s} = (x, y, u_1, \dots, u_n)$ and its associated function value $f(\mathbf{s})$. This value is computed by tracing the ray specified by the sample coordinate.

After all coarse sample points have been computed, we insert them into a kD-tree for fast neighbor queries. Our kD-tree stores samples only in leaf nodes which may contain a small, user-defined, maximum number of samples. We typically use a maximum of four samples per node. We discuss the effects of this parameter in Section 6.

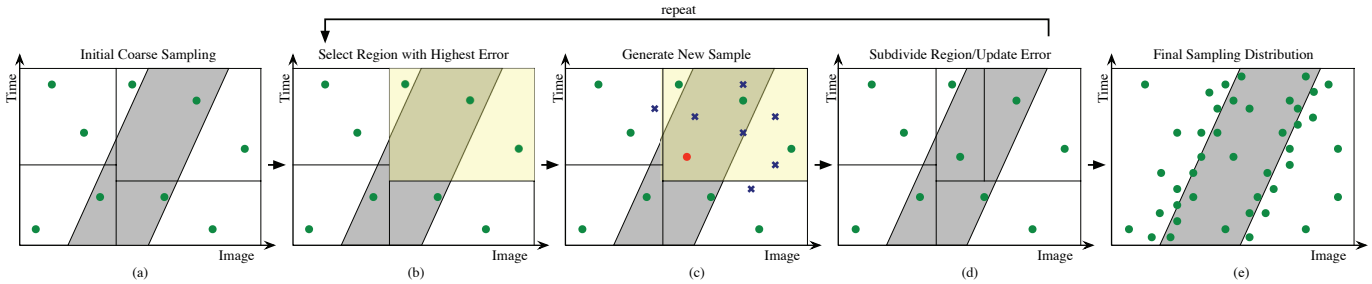

Figure 3: *To illustrate adaptive sampling in the high-dimensional domain, we use a two-dimensional example with a time axis and a one-dimensional image axis. Initially, a coarse sampling is distributed in the 2-D domain (a) and stored in a kD-tree data structure. The kD-tree leaf with the largest sample variance is chosen (b) and several candidate sample points are generated (c). The best-candidate (shown in red) is selected and the leaf is split if the node capacity is exceeded. This process (steps b through d) repeats until a desired number of samples is generated.*

4.2 Adaptive Refinement

Once the initial samples are computed, we estimate the error in each leaf node in order to find locations in the sampling domain where the multidimensional function would benefit most from additional samples. We define the total error within a leaf node, $E(N_j)$, as the integral of an error function ϵ:

$$E(N_j) = \int_{N_j} \epsilon(f(\mathbf{x}), f'_j(\mathbf{x}))\, d\mathbf{x}, \tag{5}$$

where ϵ is a function of the actual value $f(\mathbf{x})$ and the reconstructed value $f'_j(\mathbf{x})$ within the node. For the purposes of this metric we assume a constant reconstruction $f'_j(\mathbf{x}) = \bar{f}_j$ and use the average of all the samples in the node for $\bar{f}_j$. We approximate the above integral using the computed samples as

$$\begin{aligned} E(N_j) &\approx \frac{V_j}{n_j} \sum_{\mathbf{s}\in N_j} \epsilon(f(\mathbf{s}), f'_j(\mathbf{s})) \\ &= \frac{V_j}{n_j} \sum_{\mathbf{s}\in N_j} \epsilon(f(\mathbf{s}), \bar{f}_j), \end{aligned} \tag{6}$$

where V_j is the volume of kD-tree node N_j, and n_j is the number of samples contained in that node. For the error function ϵ, we use a contrast metric [Mitchell 1987] generalized to multidimensional neighborhoods. This is defined as[1]

$$\epsilon(f(\mathbf{s}), \bar{f}_j) = \frac{\left|f(\mathbf{s}) - \bar{f}_j\right|}{\bar{f}_j}. \tag{7}$$

Once an error is computed for all leaves, we place the leaf nodes into a priority queue based on their estimated error to rapidly find the node with maximum error.

To insert a new sample, we first pick the leaf node with maximum error (Figure 3b) and then employ a best-candidate technique [Mitchell 1991] to generate the sample (Figure 3c). We generate several candidate samples near the node and compute distances between each candidate and all of the existing samples. The candidate with the largest distance to all other samples is selected as the best-candidate, its contribution is evaluated, and it is added to the kD-tree. This procedure locally approximates a poisson-disk distribution.

In order to decrease the likelihood of under-sampling important features, we generate the candidate points in a region slightly larger than the node's extent. For simplicity we use the bounding-sphere of the node. This allows samples to spill into neighboring nodes and enables our approach to crawl along edges when sharp discontinuities have been detected.

If the containing node exceeds capacity when a new sample has been added, we median-split the node along its longest dimension. After inserting, we recompute the affected error values and update the priority queue. We repeat this procedure until a user-specified number of samples are computed. At the end of this process, we have a distribution of sample points and function values within the multidimensional domain which is concentrated around edges and other high-frequency regions of the function (Figure 3d).

5 Reconstruction and Integration

After the sampling step, we must compute a two-dimensional image from the samples we have evaluated. Conceptually, we first obtain a continuous reconstruction of the high-dimensional function, and then compute the value at each image location as the integral of that functon over all other dimensions. We illustrate the procedure in Figure 4 and describe each of these steps separately in the following sections.

5.1 Reconstruction

The goal of the reconstruction step is to convert the sampled values into a continuous function that matches the true multidimensional function as closely as possible. To do this, it is important to account for the anisotropic behavior of the function being approximated. For instance, consider a scene rendered with motion blur consisting of one stationary object and one moving object. The three dimensional function in this scene will have the stationary object's radiance extruded along the entire time axis. The moving object, on the other hand, will look like a sheared extrusion, with the angle of shear dependent on velocity[2]. Depth of field and other distribution effects also create highly anisotropic multidimensional functions. In order to capture this behavior accurately with the samples available, we use a reconstruction approach that blurs along edges but not across sharp boundaries. To handle such highly anisotropic functions, we perform anisotropic nearest neighbor interpolation. This interpolation uses a warped distance metric throughout the sampling domain which stretches each sample's area of influence along edges and contracts it across discontinuities. This warping metric is defined by examining the local behavior of gradients in the neighborhood around each sample point.

[1] For numerical reasons we add a small constant epsilon to the denominator to avoid division by zero.

[2] This is only conceptual; spatial variation in lighting and camera distortion will make the contribution near moving objects more complex.

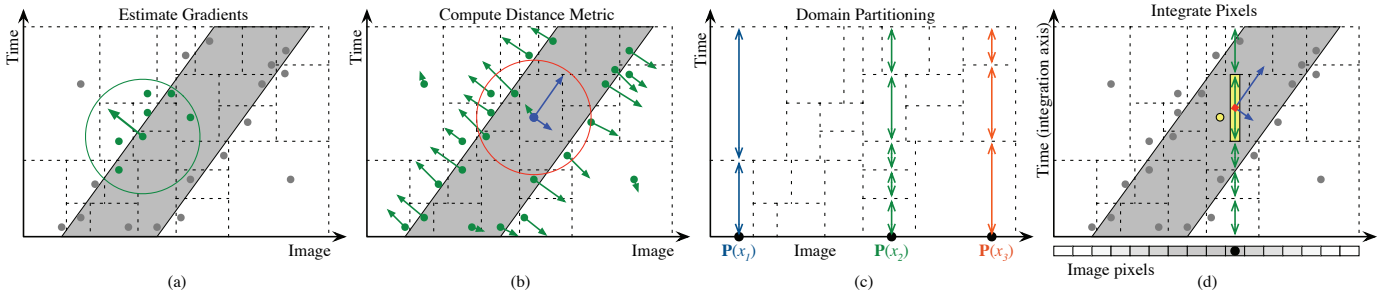

Figure 4: *During reconstruction, we first estimate a per-sample gradient by performing a least squares fit to the k nearest neighbors (a). Next we define an anisotropic distance metric (shown in blue) within each kD-tree leaf node by computing a structure tensor using the k nearest neighbor gradient vectors (b). The integral of the reconstructed function is split up into disjoint partitions P defined by the cross section of the image coordinate and the kD-tree cells. In (c) we show three example image coordinates and their corresponding partitions $P(x_1)$, $P(x_2)$, and $P(x_3)$. For each pixel in the image, we assume the function is constant over each subdomain in the partition and compute the value using anisotropic nearest neighbors at the centroids of these intervals (d). The values are multiplied by the lengths (hypervolumes) of their corresponding subdomains and accumulated for each pixel.*

We start this process by subdividing the kD-tree so that each leaf node contains exactly one sample point. We then estimate the gradient of the multidimensional function at each sample point $\mathbf{s}$ by performing a k-nearest neighbor lookup and solving a linear least squares fit to these sample values (Figure 4a). If $\mathbf{s}_i$ is the coordinate of the ith nearest neighbor of $\mathbf{s}$, where $(i \in 1 \ldots k)$, then the gradient is computed as the optimal solution to the overconstrained system

$$\begin{bmatrix} \mathbf{s}_1 - \mathbf{s} \\ \vdots \\ \mathbf{s}_k - \mathbf{s} \end{bmatrix} \nabla f(\mathbf{s}) = \begin{bmatrix} f(\mathbf{s}_1) - f(\mathbf{s}) \\ \vdots \\ f(\mathbf{s}_k) - f(\mathbf{s}) \end{bmatrix}. \tag{8}$$

These gradient estimates are then used to construct warped distance metrics, where distances are magnified across regions of rapid change. To capture this distortion property we use local gradient structure tensors, which are commonly used to perform anisotropic filtering of 2-D image data and 3-D MR images [San Jose Estepar 2005]. The structure tensor G is defined as the average of the outer products of the gradient vectors in the local neighborhood:

$$\begin{aligned} G(\mathbf{s}) &= \frac{1}{k} \sum_{i=1}^{k} \nabla f(\mathbf{s}_i) \nabla f(\mathbf{s}_i)^T \qquad (9) \\ &= \frac{1}{k} \left[\nabla f(\mathbf{s}_1) \cdots \nabla f(\mathbf{s}_k) \right] \begin{bmatrix} \nabla f(\mathbf{s}_1)^T \\ \vdots \\ \nabla f(\mathbf{s}_k)^T \end{bmatrix}. \end{aligned}$$

For a d-dimensional domain, the structure tensor G is a $d \times d$ matrix. We compute $G(\mathbf{s})$ for each sample and store it in the leaf node containing $\mathbf{s}$ (Figure 4b).

We use the structure tensor G to compute Mahalanobis distances between sample points. That is, the difference vector $v = (b - a)$ is scaled so that its squared distance is $v^T G v$. The matrix G can be thought of as specifying a general ellipsoid where the eigenvectors of G serve as the ellipsoid axes and the corresponding eigenvalues are the amount of stretch along each axis (visualized as blue arrows in Figure 4b). When G is identity it specifies a sphere and the Mahalanobis distance is equivalent to regular Euclidean distance. G can also be interpreted as computing the average squared scalar projection of the vector v onto each of the local gradients; hence, distances are magnified in orientations which contain large gradients. Two points are considered far apart if the function changes significantly between them, and close together if the function is slowly varying.

We reconstruct the multidimensional function using nearest neighbor interpolation in combination with the anisotropic distance metric. We use nearest neighbor interpolation because it is robust to rapidly changing sample densities [Amidror 2002]. To evaluate the function at any point $\mathbf{x}$, we find the sample closest to this point, using distances defined by the structure tensor G for the node containing $\mathbf{x}$. The function value is then simply the value of this closest sample. As an optimization, we compute k candidate nearest neighbors within the kD-tree using Euclidean distance. We then choose the nearest neighbor from among these candidates using the Mahalanobis distance as defined by G.

5.2 Integration

Given the continuous representation of the sampled function over the multidimensional domain, we evaluate the image-space function $L(x, y)$ using the Riemann sum from Equation 4. At each location (x, y), we partition the integration domain into disjoint regions Ω, where the partitioning, $P(x, y)$, is determined by the continuous image coordiates (x, y). The number of regions in $P(x, y)$ is equal to the number of kD-tree leaves spanning the coordinate (x, y) (see Figure 4c for an illustrative 1-D example). Using this partitioning, the radiance at each (x, y) is computed as a Riemann sum with Equation 4. The integral of the function over the cross section spanned by an image coordinate (x, y) and a leaf node is the product of the function value f_Ω and the hypervolume of the cross section V_Ω. We evaluate f_Ω at the centroid of region Ω using the anisotropic nearest neighbor interpolation described in the previous section. Because the leaf nodes are disjoint, we can compute the full integral by summing over all nodes spanning the coordinate (x, y). The final pixel intensity is an integral of Equation 4 over the pixel filter; however, for simplicity we use a box filter and evaluate this equation once at the center of each pixel. This approach is illustrated in Figure 4d.

6 Implementation and Results

We have implemented the multidimensional adaptive sampler as two plugins for the PBRT rendering system [Pharr and Humphreys 2004]; one plugin for sampling and one for reconstruction. In this section, we explain the parameters used by our implementation as well as the settings used in our results.

6.1 Sampling Implementation

The sampling algorithm as described in Section 4 is controlled by a number of parameters. The two parameters that have the highest impact on image quality are the number of initial samples

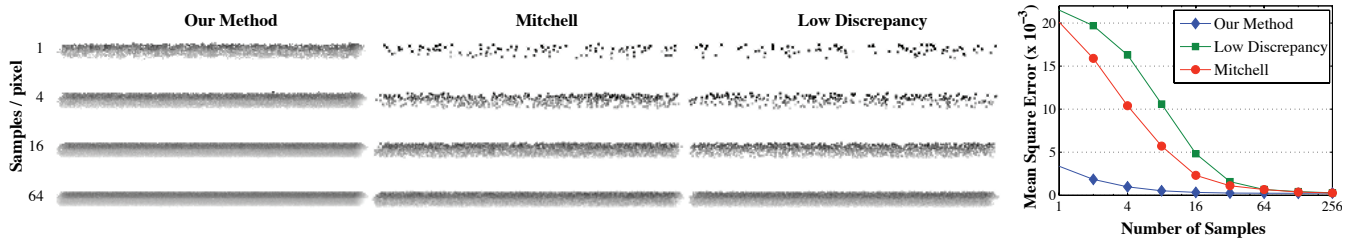

Figure 5: *A motion blurred sphere is moving horizontally across the image. Note how our sampler is able to achieve a significantly better sampling distribution and reconstruction as it is able to track the surface of the moving sphere accross both time and space. For this scene the Mitchell sampler requires roughly an order of magnitude more sampler to produce an image with the same MSE.*

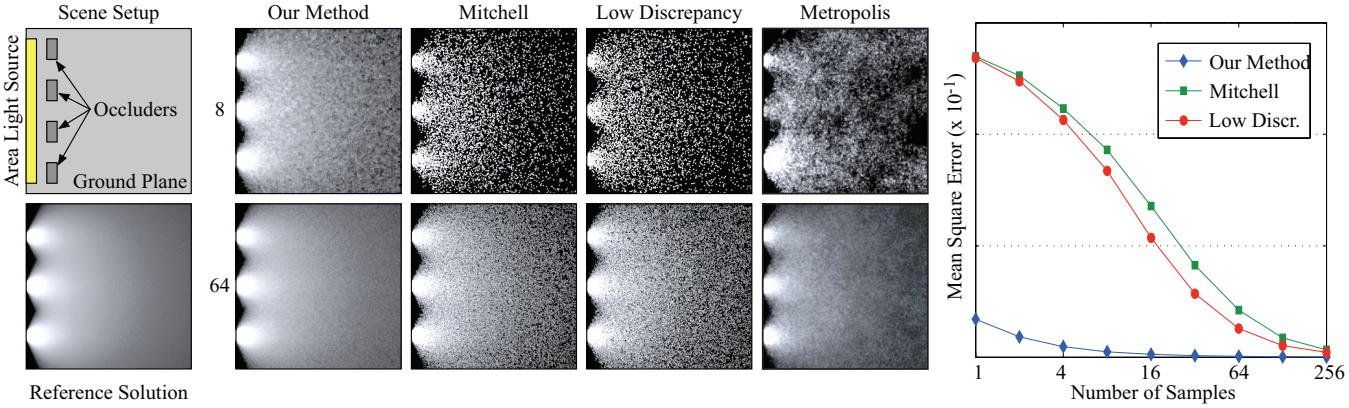

Figure 6: *An occluded area light source. Our technique adaptively samples the openings in the blocker where the light source is visible, and it produces images with the same MSE as Mitchell's sampler with nearly two orders of magnitude fewer samples. We only provide a visual comparison to Metropolis Light Transport because we render it with a different rendering system.*

and the desired average number of samples per pixel in the final distribution. Additionally, the number of samples per kD-tree leaf influences the accuracy of our error metric and the adaptivity of the final distribution: more samples per node results in smoother and more robust error estimates but also results in a more uniform sample distribution. Since the error metric requires each node split to result in at least 2 sample per node, the minimum value for this parameter is 4. We have used this minimum value in most of our test scenes as it gives a good overall performance, with the exception of Figure 6 which uses 8. When adding new samples using the best-candidate approach, we use 4 candidates for all our results. When nodes are subdivided in the kD-tree we also use an axis-scaling parameter. This parameter controls the relative scaling of the different dimensions in our sampling space. This parameter is shared with the reconstruction implementation and it is described in more detail the next section.

6.2 Reconstruction Implementation

The reconstruction pass relies on one main parameter k: the number of nearest neighbors to use for estimating the gradients, for computing the anisotropic distance metric, and for finding candidates for the nearest neighbor interpolation. We set $k = 12$ unless otherwise noted. Making k much smaller can introduce some artifacts in the reconstruction such as noise due to inaccurate statistics. We have found that for our scenes increasing k beyond 20 does not make a difference in the quality of the results. Making k larger increases render time without improving image quality.

Additionally, we use a parameter to define the relative scale of the image axes with respect to all other dimensions. The splitting of the kD-tree during adaptive sampling, the contrast error metric, the gradient and tensor computations, and the k-nearest neighbor *candidates* for interpolation are all computed using un-warped Euclidean distances. Because of this, the relative scale of axes in the multidimensional domain have an influence on the sampling distribution and reconstruction of our algorithm. Since our results here are focused on sampling domains of medium (3–5) dimensionality, we have found it sufficient to specify the relative scale of the different types of dimensions manually. For images containing only motion blur, depth of field, or soft shadows a single parameter is sufficient to specify the scale of the image axes relative to the other dimensions. The smallest dimension of the image is always mapped to an interval of $[0 \ldots 1]$ in the multidimensional hypervolume and the other image axis is scaled to maintain the aspect ratio. A scaling factor of 2 for the motion blur axis means that time samples are distributed within a range of $[0 \ldots 2]$ in the multidimensional domain. We report the scaling parameters individually for each of our results in the following section.

6.3 Memory Usage

Memory usage is an important consideration since we store and maintain each sample in a kD-tree. In our implementation, each sample requires about 400 bytes of storage. This includes the memory overhead of the kD-tree structure, the priority queue, gradient information, and structure tensors. This comes out to 400 MB for a 1K x 1K image with an average of 1 sample per pixel. However, we have not yet optimized the memory consumption in our implementation. To reduce memory requirements for images requiring many samples, we currently divide the image into disjoint tiles. We have found this method to work well, though it does introduce subtle tiling artifacts that are visible in the visualizations of sample density. These artifacts are, in practice, not noticeable in the final reconstructed image.

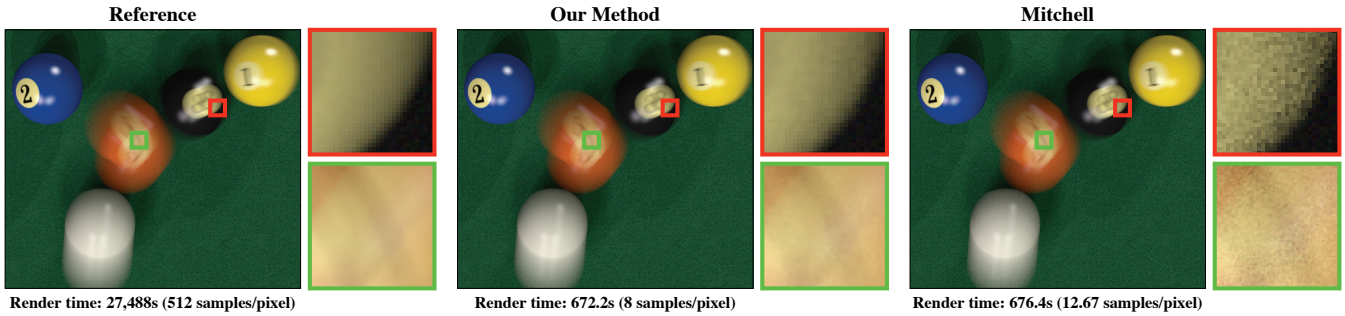

Figure 7: *Pool scene showing motion blur. Given equal time, our sampler generates an image with significantly less noise and an MSE that is 9 times lower than Mitchell's adaptive sampler. The sample density image shows how our sampler is able to find and sample the regions with strong motion. These regions are problematic for the Mitchell sampler as shown in the error images.*

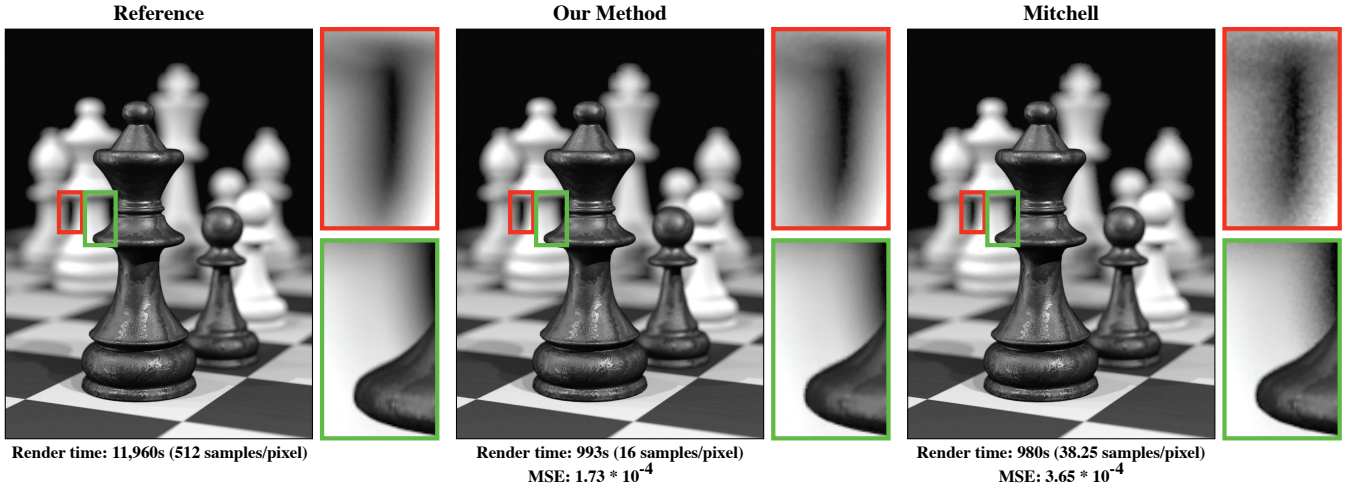

Figure 8: *Chess scene showing depth of field. Our technique is able to sample and reconstruct the regions that are out of focus while the Mitchell sampler is noisy in these regions. Even though the out of focus areas are a small part of the image our sampler is able to produce an image with an MSE that is two times lower than the equal time rendered image with Mitchell's adaptive sampler. Due to our anisotropic reconstruction technique we can successfully reconstruct both blurry, out-of-focus regions (red), and sharp, in-focus regions (green). Without anisotropic reconstruction, the MSE of our method more than doubles to* $3.85 * 10^{-4}$.

6.4 Results

We have applied our adaptive sampling scheme to a number of multidimensional sampling domains, including motion blur, soft shadows, and depth-of-field. All results were rendered on an 2.4GHz Intel Core 2 Q6600 using one core. We compare our technique to Mitchell's adaptive sampling technique [Mitchell 1991]. We chose Mitchell's sampler because it performed as well, or better than other image-space adaptive sampling techniques that we tried including Whitted adaptive sampling. For all results we tuned the Mitchell parameters to give the best quality for the given render time or total number of samples.

Figure 5 compares our method to both uniform sampling (using low discrepancy points) and Mitchell's image-space adaptive sampling technique in a scene of a fast moving sphere on a white background. Motion blur constitutes a 3-D sampling domain: 2 for the image dimensions, and 1 for time dimension. Using only 2 samples per pixel, our technique is able to produce images with a lower error than Mitchell sampling with 16 samples per pixel.

In Figure 6 we apply our technique to a 4-D domain by adaptively sampling soft shadows across the image plane. The scene contains four pillars blocking a large area light source which is illuminating a plane. Our algorithm is able to detect the discontinuities caused by the occluders and adaptively samples these high frequency regions in the multidimensional domain. For these two results, we used a value of $k = 15$ and set the time and lighting axes scaling factors to 1/2 and 1/3 respectively. We use 1024 initial samples for our method in both results. We also provide a comparison to Metropolis Light Transport (MLT). We only include a visual comparison since our MLT implementation uses a different rendering system. The result using 8 samples with our method has similar quality as MLT with 64 samples.

Both of these examples demonstrate the challenges of image-space adaptive sampling in the presence of even a simple multidimensional domain. Mitchell's adaptive sampling provides little benefit in these scenes since each individual image-space sample is a poor estimate of the integral of scene radiance along all non-image dimensions, making it difficult to detect important features. Furthermore, due to the extent of the motion blur and size of the area light, nearby image samples span a large multidimensional domain leading to gaps in the motion blurred sphere and scattered dots in the penumbra of shadows. Our approach, on the other hand, is able to distribute sample contribution over large pixel regions, resulting in much less noise and more efficient sample placement.

Figure 7, another motion blur example, shows an updated version of the classic billiard ball scene from Cook et al. [1984]. The time axis scaling factor is set to 1/16 and we use 10K initial samples. A visualization of sample density using our approach is shown in

Figure 9: *Car showing a 5-D sampling domain including both motion blur and depth-of-field. We use 10K initial samples and 32 samples per pixel in the final distribution. This scene renders in 2884 seconds and reconstruction takes 22% of total render time.*

Figure 10. Notice that in regions where the balls are stationary, our technique does an excellent job of placing samples around image discontinuities (e.g., texture edges), while simultaneously allocating samples to areas with considerable motion, resulting in considerably reduced noise and mean square error (Figure 11) with no increase in rendering time. For the pool image the reconstruction stage took 25% of total render time.

Figure 8 shows a chess scene with substantial focus effects due to a shallow depth of field; we perform adaptive sampling across both the image plane and the lens aperture and visualize the projected sample density in Figure 10. The dimensions corresponding to the aperture are scaled by a factor of 1/48 and we use 10K initial samples. In this 4-D sampling space, we again correctly reconstruct both discontinuities in the image plane (e.g., the edges of the in-focus foreground object), as well as discontinuities in the non-image dimensions, resulting in smoothly blurred out-of-focus regions. For this scene reconstruction takes 34% of total render time. Omitting the anisotropic distance metric calculation results in a reconstruction with more than twice the mean square error (MSE) of our approach.

Finally, Figure 9 shows a car speeding down a highway. This scene features both motion blur and depth of field, resulting in a combined sampling domain of 5 dimensions. In this scene we set the scale of all non-image axes to 1/64 and use 10K initial samples.

We explore the effects of different parameter settings for the pool scene and provide numerical results in Figure 12. In these experiments we kept the total number of samples constant and varied one sampling or reconstruction parameter at a time. The number of nearest neighbors for reconstruction involves a trade-off between error and reconstruction time. Increasing the number of nearest neighbors results in smaller errors as shown in Figure 12a. However reconstruction time also increases linearly (Figure 12b). Figure 12c shows that the number of candidate samples has little influence on the error. Since it linearly increases the sampling time (Figure 12d), we prefer to use few candidates for performance reasons. In Figure 12e we see that the optimal scaling factor for the image axis is about 16, but our method is not very sensitive to this parameter. The number of samples per cell also has rather little influence on the error (Figure 12f). For this particular scene, increasing the number of samples per cell to more than 4 increases the reconstruction error. This is because samples within each cell do not adapt to the underlying function within the cell. Increasing the number of samples per cell is probably beneficial for higher dimensional functions to obtain a better error estimate during adaptive sampling.

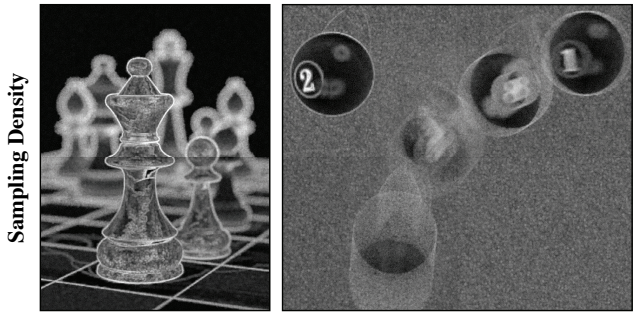

Figure 10: *Visualizations of projected sample distributions using our method for the chess scene from Figure 8 and the pool scene from Figure 7. Our adaptive sampler places samples both around high frequency image discontinuities (in focus chess piece and stationary pool ball) as well as in regions which exhibit significant motion blur or depth of field effects.*

7 Discussion and Future Work

In this section we discuss limitations of our method. We relate our approach mathematically to traditional randomized sampling, and present ideas that we feel are fruitful for future exploration.

7.1 Limitations

Similar as other adaptive techniques our method may miss small features of a scene. For example, our method may miss features not captured by the initial sampling. Furthermore, the error metric during the adaptive sampling may become zero if all samples in a cell happen to have the same radiance, even if the underlying function varies within the cell. In these cases our method will not place a new sample to capture a region where the function exhibits variation unless a neighboring cell generates this sample. Figure 13 shows an example of the same scene as in Figure 6, but with a reduced size of the gaps between occluders. Depending on the initial samples and parameter settings, our method may not detect illumination coming through a gap. Similar effects can occur if the scene geometry has lots of small features such as hair, foliage, or grass.

It should be noted that our method is more robust than most 2D sampling methods. Suppose that all initial samples in Figure 1 fell in the same region (e.g., the grey region). In this case, Mitchell's adaptive sampling method does not place new samples at all because all regions satisfy its adaptive sampling criterion. On contrast, our method places new samples even in this case because it always choses a cell with the largest error (in this case, any of the cells) to place a new sample. In addition, our method is able to exploit co-

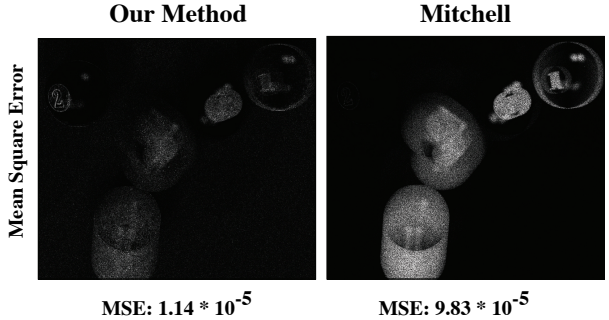

Figure 11: *Mean square error visualizations for the pool scene. Given equal time, our sampler generates an image with an MSE that is 9 times lower than Mitchell's adaptive sampler. Isotropic reconstruction results in an MSE of* $2.1 * 10^{-5}$.

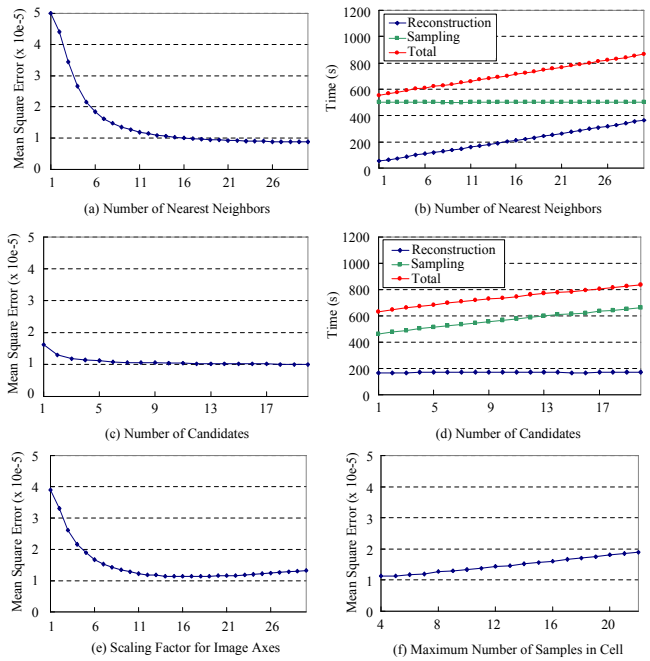

Figure 12: *Effect of parameters. We rendered the billiard ball scene in Figure 7 by varying: (a) the number of nearest neighbors for reconstruction, (c) the number of candidates during sampling (i.e., the number of candidates for new sample placement), (e) the scaling factor for image axes (i.e., the reciprocal of the scaling factor for the other axes), (f) the maximum number of samples per cell. The graphs (b) and (d) show the corresponding rendering time.*

herency beyond the image space which improves robustness further. For example, consider the moving sphere scene in Figure 5. In this case, the 3D function of the sphere using pixel position and time becomes a cylinder. Our method captures this coherency in the three dimensional function and focuses samples around the surface of the cylinder. A two dimensional adaptive sampling method cannot exploit this coherency and may miss part of the cylinder if this is not captured by its adaptive sampling metric on image space.

7.2 Comparison to Monte Carlo Integration

Although our method is a Monte Carlo method as it uses random numbers to place samples, our approach is not a standard Monte Carlo *integration* method. Instead, we use a Riemann sum for integration, similar to numerical integration methods based on quadrature rules. Furthermore, we are able to determine the relative weighting of the samples *after* they have been computed. In standard Monte Carlo integration methods, relative weighting of the samples are usually fixed by the probability density function. It essentially means that standard Monte Carlo integration methods do not use information obtained by samples that have been observed so far. Markov Chain Monte Carlo methods including Metropolis method use information from just one previous sample, whereas our method uses all previous samples to generate a new sample.

7.3 Generalizing to Higher Dimensional Domains

We have shown results for sampling domains of three (Figure 7), four (Figures 8 and 6), and five (Figure 9) dimensions, although the theory behind the technique extends to any number of dimensions. Of course, a medium-dimensional (e.g., 5-D) version of our

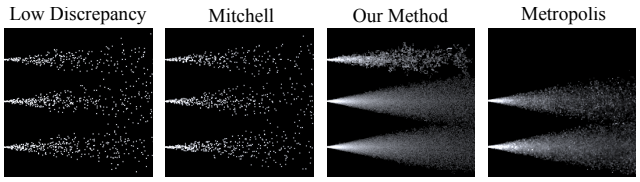

Figure 13: *A failure case. All images are rendered using 64 samples per pixel. The only difference from Figure 6 is that the size of the gaps between occluders is significantly smaller. Our method may miss illumination coming through one of the gaps depending on the parameters. Note that this scene is equally challenging for Metropolis Light Transport.*

approach could always be used within a higher dimensional space by adaptively sampling along only a 5-D subset of the full domain (much in the same way that image-space adaptive sampling performs adaptive decisions on a 2-D subset). However, as the number of dimensions increases, as it will for effects such as glossy reflection/transmission, global illumination, and participating media, new issues will arise that provide exciting areas for future exploration.

Neighbor Searching and Anisotropy The most significant challenge that arises when generalizing our technique to higher dimensions is the cost of computing nearest neighbors, which is exponentially related to the dimensionality of the search. We intend to explore the use of nearest-neighbor approximation algorithms to alleviate the curse of dimensionality for neighbor searches [Indyk and Motwani 1998]. In addition, we do not know how many neighbors will be required to generate robust local function statistics in very high dimensional space.

As the number of dimensions increases, accurately detecting the anisotropic behavior becomes more challenging because there are more degrees of freedom. One possible solution would be to store or compute additional per-sample information. For instance, we could compute gradient information directly for a subset of the dimensions, using techniques explored by previous researchers [Ward and Heckbert 1992; Ramamoorthi et al. 2007], or we could store local frequency content [Durand et al. 2005] with each sample. Sample points could also be tagged with auxiliary data such as motion vectors to analytically compute appropriate anisotropic kernels.

Nonuniform Dimension Scaling Because the non-image dimensions can all have different units, determining the relative scale of each axis is important. Moreover, scaling factors directly affect resulting image quality as in Figure 12. As a general principle, the scale of each axis should be set such that the maximum frequency in each direction is the same. For motion blur, the scale should be set such that the maximum motion in one unit of time is equal to one pixel. This cannot be predicted exactly because of highlights, shadows, and other lighting effects, but using the motion of objects in the scene may provide a good first-order approximation. For depth of field, the expected frequencies can be computed based on depth using light field analysis [Chai et al. 2000]. Such an approach is theoretically justified only for diffuse surfaces but could provide a good starting point. Frequency analysis of general light transport is an area of active research that could be directly applied to our algorithm.

8 Conclusions

We have developed a new adaptive sampling and reconstruction technique for sampling domains containing more than two dimensions, such as those that arise in scenes with motion blur, depth of field, or soft shadows. Sampling directly in the high dimensional

space has several advantages: the samples returned by the ray tracer are deterministic, our sampler can track discontinuities in the high dimensional function, and our reconstruction can take advantage of smooth regions within the function by using each sample in these regions to reconstruct the final radiance value for several neighboring pixels. We have demonstrated several scenes with complex geometry, lighting, and camera effects for which our technique outperforms traditional adaptive sampling in both visual quality and quantitative error.

9 Acknowledgements

This work was supported in part by NSF grant CPA 0701992 and the UCSD FWGrid Project, NSF Research Infrastructure Grant Number EIA-0303622. We would also like to thank UCSD graphics lab members for fruitful discussion.

References

AMIDROR, I. 2002. Scattered data interpolation methods for electronic imaging systems: a survey. *Journal of Electronic Imaging 11*, 2 (April), 157–176.

BERNTSEN, J., ESPELID, T. O., AND GENZ, A. 1991. An adaptive algorithm for the approximate calculation of multiple integrals. *ACM Transactions on Mathematical Software 17*, 4, 437–451.

BOLIN, M. R., AND MEYER, G. W. 1998. A perceptually based adaptive sampling algorithm. *Computer Graphics 32*, Annual Conference Series, 299–309.

CHAI, J.-X., CHAN, S.-C., SHUM, H.-Y., AND TONG, X. 2000. Plenoptic sampling. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 307–318.

CLARBERG, P., JAROSZ, W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Wavelet importance sampling: Efficiently evaluating products of complex functions. *ACM Trans. Graph. 24*, 3, 1166–1175.

COOK, R. L., PORTER, T., AND CARPENTER, L. 1984. Distributed ray tracing. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, vol. 18, 137–45.

DURAND, F., HOLZSCHUCH, N., SOLER, C., CHAN, E., AND SILLION, F. X. 2005. A frequency analysis of light transport. 1115–1126.

DUTRÉ, P., BALA, K., BEKAERT, P., AND SHIRLEY, P. 2006. *Advanced Global Illumination*. AK Peters Ltd.

GLASSNER, A. 1995. *Principles of Digital Image Synthesis*. Morgan Kaufmann.

INDYK, P., AND MOTWANI, R. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Symposium on Theory of Computation (STOC)*, 604–613.

JENSEN, H. W. 2001. *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters, Ltd., Natick, MA.

KAJIYA, J. T. 1986. The rendering equation. In *Computer Graphics (SIGGRAPH '86 Proceedings)*, D. C. Evans and R. J. Athay, Eds., vol. 20, 143–150.

KELEMEN, C., AND SZIRMAY-KALOS, L. 2001. Simple and robust mutation strategy for metropolis light transport algorithm. Tech. Rep. TR-186-2-01-18, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, July. human contact: technical-report@cg.tuwien.ac.at.

KIRK, D., AND ARVO, J. 1991. Unbiased sampling techniques for image synthesis. In *Proceedings of SIGGRAPH '91*, ACM Press, New York, NY, USA, 153–156.

LANDAU, H., AND POLLAK, H. 1962. Prolate spheroidal functions, Fourier analysis and uncertainty, III. the dimension of the space of essentially time- and band-limited signals. *Systems Technical Journal 41*, 4 (July), 1295–1336.

LAWRENCE, J., RUSINKIEWICZ, S., AND RAMAMOORTHI, R. 2004. Efficient brdf importance sampling using a factored representation. *ACM Trans. Graph. 23*, 3, 496–505.

LEESON, W. 2003. Rendering with adaptive integration. In *Graphics programming methods*, Charles River Media, Inc., Rockland, MA, USA, 271–278.

MITCHELL, D. P. 1987. Generating antialiased images at low sampling densities. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, vol. 21, 65–72.

MITCHELL, D., 1990. The antialiasing problem in ray tracing, Aug. SIGGRAPH 1990 Course Notes.

MITCHELL, D. P. 1991. Spectrally optimal sampling for distributed ray tracing. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, vol. 25, 157–164.

PHARR, M., AND HUMPHREYS, G. 2004. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann.

PRESS, W. H., AND FARRAR, G. R. 1990. Recursive stratified sampling for multidimensional monte carlo integration. *Comput. Phys. 4*, 2, 190–195.

RAMAMOORTHI, R., MAHAJAN, D., AND BELHUMEUR, P. 2007. A first-order analysis of lighting, shading, and shadows. *ACM Trans. Graph. 26*, 1, 2.

RIGAU, J., FEIXAS, M., AND SBERT, M. 2003. Refinement criteria based on f-divergences. In *Rendering Techniques*, 260–269.

SAN JOSE ESTEPAR, R. 2005. *Local Structure Tensor for Multidimensional Signal Processing. Applications to Medical Image Analysis.* PhD thesis, University of Valladolid, Spain.

VEACH, E., AND GUIBAS, L. J. 1997. Metropolis light transport. In *Computer Graphics (SIGGRAPH Proceedings)*, 65–76.

WALTER, B., ARBREE, A., BALA, K., AND GREENBERG, D. P. 2006. Multidimensional lightcuts. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, ACM, New York, NY, USA, 1081–1088.

WARD, G. J., AND HECKBERT, P. 1992. Irradiance Gradients. In *Third Eurographics Workshop on Rendering*, 85–98.

WARD, G. J., RUBINSTEIN, F. M., AND CLEAR, R. D. 1988. A ray tracing solution for diffuse interreflection. In *Computer Graphics (SIGGRAPH '88 Proceedings)*, J. Dill, Ed., vol. 22, 85–92.

WHITTED, T. 1980. An improved illumination model for shaded display. *Communications of the ACM 23*, 6, 343–349.