MIT Libraries | DSpace@MIT

MIT Open Access Articles

Practical filtering for efficient ray-traced directional occlusion
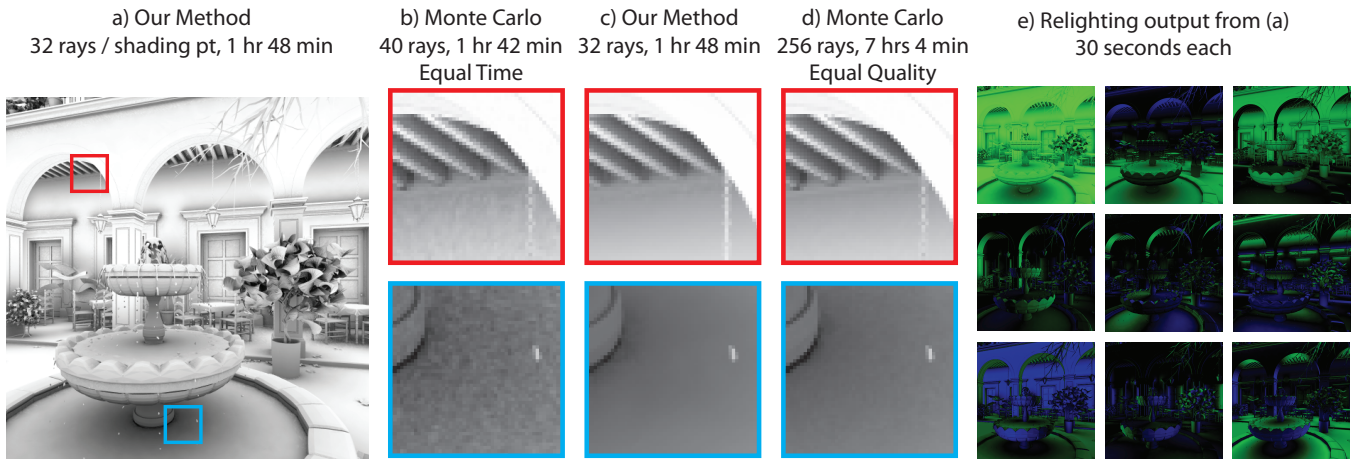
Massachusetts Institute of Technology

Practical Filtering for Efficient Ray-Traced Directional Occlusion

Kevin Egan [*]
Columbia University

Frédo Durand
MIT CSAIL

Ravi Ramamoorthi
University of California, Berkeley

| a) Our Method 32 rays / shading pt, 1 hr 48 min | b) Monte Carlo 40 rays, 1 hr 42 min Equal Time | c) Our Method 32 rays, 1 hr 48 min | d) Monte Carlo 256 rays, 7 hrs 4 min Equal Quality | e) Relighting output from (a) 30 seconds each |

Figure 1: *(a) A visualization of ambient occlusion produced by our method. This scene used 32 samples per shading point, 13 rays in the sparse sampling pass (41 min) and 19 rays in the second pass in areas with contact shadows (1 hr 7 min). Total running time for both passes was 1 hr 48 min. (b) Closeups of Monte Carlo using equal time (40 samples, 1 hr 42 min), noise can be seen. (c) Closeups of our method. (d) Closeups of Monte Carlo with equal quality (256 samples, 7 hrs 4 min). (e) At little extra cost our method can also compute spherical harmonic occlusion for low frequency lighting. While computing (a) our method also outputs directional occlusion information for 9 spherical harmonic coefficients (green is positive, blue is negative).*

Abstract

Ambient occlusion and directional (spherical harmonic) occlusion have become a staple of production rendering because they capture many visually important qualities of global illumination while being reusable across multiple artistic lighting iterations. However, ray-traced solutions for hemispherical occlusion require many rays per shading point (typically 256-1024) due to the full hemispherical angular domain. Moreover, each ray can be expensive in scenes with moderate to high geometric complexity. However, many nearby rays sample similar areas, and the final occlusion result is often low frequency. We give a frequency analysis of shadow light fields using distant illumination with a general BRDF and normal mapping, allowing us to share ray information even among complex receivers. We also present a new rotationally-invariant filter that easily handles samples spread over a large angular domain. Our method can deliver 4x speed up for scenes that are computationally bound by ray tracing costs.

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

Keywords: ambient occlusion, relighting, sampling, frequency analysis

[*] e-mail:ktegan@cs.columbia.edu

Links: DL PDF

1 Introduction

Modern production rendering algorithms often compute low frequency hemispherical occlusion, where the surrounding environment is approximated to either be a solid white dome (ambient occlusion), or a series of low frequency spherical harmonics. Two different bodies of work related to ambient occlusion were given scientific Academy Awards in 2010 [AcademyAwards 2010], and the movie Avatar used ray-traced ambient and spherical harmonic occlusion for lighting and final rendering [Pantaleoni et al. 2010]. While fully sampling the surrounding illumination at each receiver is the completely accurate way to compute global illumination, these approximations of distant lighting work well in practice. Another advantage is that the ambient occlusion and spherical harmonic calculations are independent of the final lighting environment and can be reused throughout the lighting process.

Ray-traced occlusion is often very expensive to compute due to the large number of incoherent ray casts. The authors of the PantaRay system state that they typically shoot 512 or 1024 rays per shading
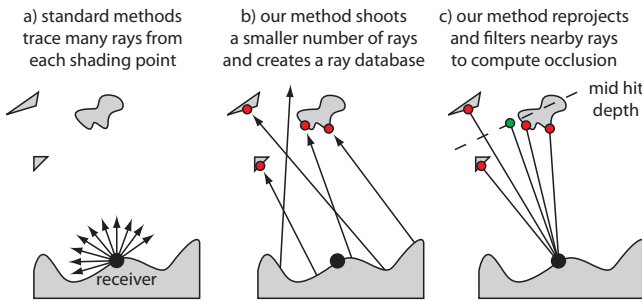
a) standard methods trace many rays from each shading point

b) our method shoots a smaller number of rays and creates a ray database

c) our method reprojects and filters nearby rays to compute occlusion

Figure 2: *(a) We show a simple scene in flatland. Standard methods for computing ray-traced ambient occlusion shoot many rays. (b) Our method shoots a sparse set of rays in a first pass and saves them to a point cloud. Red dots are ray hits. (c) In a second pass we use our theory to reproject and weight nearby rays to compute directional or ambient occlusion. The green dot represents the target point of the unoccluded ray (the intersection between the orginal ray and the mid depth plane).*

point to compute occlusion [Pantaleoni et al. 2010]. One of their test frames from the movie Avatar took over 16 hours and 520 billion rays to compute an occlusion pass. While our scenes do not have the complexity of production environments, our method shows substantial performance benefits with scenes of moderate complexity. As scenes become more computationally bound by ray tracing, the benefits of our method increase.

We propose a method that speeds up occlusion calculations by shooting fewer rays and sharing data across shading points (shown in Figure 2). We present a new frequency analysis of occlusion from an omni-directional distant light source that also includes normal mapping and a general BRDF at the receiver point. Using this analysis, we develop a method to share rays across the full hemisphere of directions, vastly cutting down on the number of expensive incoherent ray casts. Our method makes a number of important contributions:

Frequency Analysis of Distant Lighting: We present a frequency analysis of distant lighting from all possible incoming directions over the hemisphere in Section 3. Our work builds on the recent analysis of Egan et al. [2011] for shadows from a compact planar area light souce. We first derive new equations to handle distant lighting by splitting up the spherical domain into linear subdomains (such as cube map faces). We then derive the appropriate bandlimits and filter sizes for each linear sub-domain. This theory is used as the basis for our new rotationally-invariant filter.

General BRDFs and Normal Mapping: We also show how the occlusion signal interacts with general BRDFs. We show that the sum of the lighting and BRDF bandlimit determines the cutoff frequencies for occlusion. Furthermore, as long as the surface BRDF is bandlimited, high frequency changes in the normal do not affect our occlusion calculations. Our method takes advantage of this by sparsely sampling occlusion, which greatly reduces the the number of expensive ray casts. We show results with sparse occlusion sampling, glossy BRDFs and low frequency environment lighting (Figure 8).

Rotationally-Invariant Filter: We present a filter that uses the above theory, modified such that it is rotationally-invariant (Section 4). This property of our filter allows us to handle large angular domains without needing to stitch together linear sub-domains. The

filter is intuitive, easy to implement, and constructed using the frequency analysis in Section 3. Results with our filter are shown in Figures 1, 7, 8, and 9.

2 Previous Work

Shadows and Ambient Occlusion: Ambient occlusion is an approximation of global illumination that is simply the aggregate visibility for a solid white dome [Zhukov et al. 1998; Landis 2008]. Spherical harmonic occlusion improves on this, computing the aggregate visibility for a number of low order spherical harmonics. We focus on the methods most closely related to our paper, and we refer the reader to a survey of recent ambient occlusion techniques [Méndez-Feliu and Sbert 2009].

The PantaRay system uses GPU ray tracing, various spatial hierarchies, and geometric LOD to compute spherical harmonic occlusion [Pantaleoni et al. 2010]. Our method is complementary because we focus on reducing the number of rays cast, whereas PantaRay focuses on reducing the cost per ray. Another recent method reduces cost per ray by using similar sample patterns across many receivers so that they can process multiple receivers in parallel on the GPU [Laine and Karras 2010]. Pixar's RenderMan represents distant occluding geometry using an octree that contains point sampled data as well as spherical harmonics at parent nodes, and rasterizes this data onto a coarse grid at each receiver [Christensen 2008]. We compare our results to point based occlusion, and discuss the relevant tradeoffs with both methods in Section 6.

Interactive techniques for approximating ambient occlusion are also used. Ambient Occlusion Volumes compute analytic occlusion per polygon [McGuire 2010]. Because some polygons may be double counted, the method approximates the aggregate occlusion using a compensation map, whereas our method samples occlusion using ray tracing and does not suffer from double counting. Screen space methods can be very efficient, but may miss geometry that is not directly visible to the camera [Bavoil and Sainz 2009]. Since we focus on ray tracing our method accounts for all relevant occluders.

Frequency Analysis and Reconstruction: Our method builds upon recent sheared filtering techniques for reconstructing light fields, motion blur, and shadows from planar light sources [Chai et al. 2000; Egan et al. 2009; Egan et al. 2011]. Other methods have also examined occlusion in the Fourier domain [Soler and Sillion 1998; Durand et al. 2005; Ramamoorthi et al. 2005; Lanman et al. 2008]. We extend the theory to include distant lighting, a general BRDF, and high frequency normal maps. We also introduce a rotationally invariant filter that uses the theory for sheared filtering but is able to orient itself in any direction across a large angular domain.

A number of other techniques have also shared data between neighboring receiver points to cut down on computation. Irradiance caching is used to filter sparse samples of low frequency indirect lighting [Ward et al. 1988]. Irradiance caching aggregates angular information, and then shares it spatially, whereas our method is able to share samples across both space and angle. Radiance caching [Křivánek et al. 2005] shares recorded radiance values and gradients between surface points while assuming that the visibility of shared radiance samples does not change between receivers. Our method determines which samples are appropriate to share by setting the filter radius based on the minimum and maximum occluder depths. Irradiance decomposition uses low frequency radiance caching for far-field components and switches to a heuristic for occluders closer than a fixed depth chosen by the user [Arikan et al. 2005]. Recently Lehtinen et al. [2011] developed a method that locally reconstructs multiple points within a
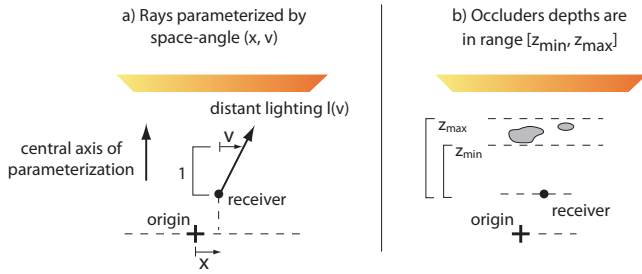
Figure 3: *For our theoretical derivation we split up the environment into distinct faces and use a linearized angle. We parameterize the receiver based on the spatial offset x from the origin. The linearized angle v is the offset from the central direction of the cube map at a plane one unit away from the receiver. The occluders are bounded by a range of distances [z_{min}, z_{max}] from the receiver.*

pixel using the GPU. Unlike these methods, the signal processing framework used by our method smoothly scales our filter and tells us how much information to share.

Sparse Transport Computation: Precomputed Radiance Transport is the foundation for current relighting methods [Sloan et al. 2002; Ng et al. 2003]. However, the precomputation time required for these methods is often prohibitive when using standard Monte Carlo sampling. A number of methods have used various techniques to sparsely sample light transport. Row-column sampling uses shadow maps to sparsely compute light transport for a single surface point to all lights (one row), or a single light to all surface points (one column) [Hašan et al. 2007]. The big advantage of row-column sampling is the batch visibility computation achieved by using shadow maps, whereas our method uses ray tracing to avoid shadow map artifacts. Huang et al. sparsely precompute the light transport matrix by only densely sampling the angular domain at selected "dense" vertices [Huang and Ramamoorthi 2010]. Our method extends this by more intelligently sharing rays across space and angle. Our method also computes filter widths based on the range of depths of the occluders.

3 Theory

We first discuss the basic reflection equation and derive equations for occlusion from distant lighting in the primal domain. We then look at preliminary Fourier derivations taken from previous work, before deriving new equations that give insight into occlusion and reflection under distant lighting. Our final reconstruction uses a filter that is based on our theory but uses geometric measures that can rotate to any given sample in the hemispherical domain.

3.1 Occlusion from Distant Lighting

3.1.1 Preliminaries

The reflected radiance $h(x, \omega_o)$ at a surface point x in direction ω_o can be written as follows:

$$h(x, \omega_o) = \int r(x, \omega_i, \omega_o) l(\omega_i) f(x, \omega_i) \, d\omega_i \ .$$

The reflected radiance is computed by integrating over all incoming light directions ω_i. Inside the integral is a product of the spatially

varying BRDF $r(x, \omega_i, \omega_o)$ (which includes the damped cosine term for compactness), the distant lighting $l(\omega_i)$, and occlusion $f(x, \omega_i)$.

Our method focuses on direct lighting with a fixed camera such that the viewing angle ω_o is fixed for a given spatial location x,

$$h(x) = \int r(x, \omega_i) l(\omega_i) f(x, \omega_i) \, d\omega_i \ . \quad (1)$$

To compute ambient occlusion, we simply set $l(\omega_i)$ to a constant value of 1, and $r(x, \omega_i)$ to a clamped cosine $\max(n(x) \cdot \omega_i, 0)$ in Equation 1 (where $n(x)$ is the surface normal). For more accurate low frequency relighting, we replace the lighting $l(\omega_i)$ term with a set of low order spherical harmonic basis functions $[s_0(\omega_i), ..., s_n(\omega_i)]$, and a corresponding intermediate response function $[h'_0(x), ..., h'_n(x)]$ (visualized in Figure 1e). Using these intermediate values, we can then easily relight the scene by projecting any distant lighting $l(\omega_i)$ into the orthonormal spherical harmonic basis, taking the resulting coefficients $[l_0, ..., l_n]$, and performing a dot product:

$$h(x) = \sum_{i=0}^{n} h'_i(x) l_i \ . \quad (2)$$

Directional and ambient occlusion are primarily used to aid in the realistic approximation of slightly glossy or matte components of a BRDF. If a BRDF has sharper specular reflections this is usually calculated more directly using a different reflection algorithm.

3.1.2 Distant Lighting in Primal Domain

The above angular parameterization using ω is not easy to analyze in a Fourier context. We could expand the lighting into spherical harmonics, but proceeding to analyze the spatial-angular occlusion of $f(x, \omega)$ and subsequent transport becomes intractable. Thus, like Durand et al. [2005], we use the Fourier basis with a linearized measure of angle.
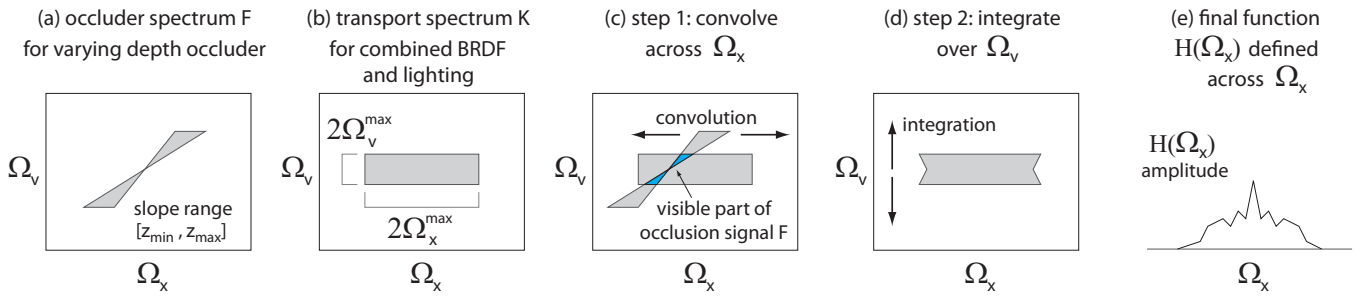
We first reparameterize the circle of angles in flatland to a square map (the analog of a cube map in 3D). For each map face we define a linearized angle v measured against that map's central axis of projection. The v measures a direction vector's offset from the axis of parameterization at a plane 1 unit away (shown in Figure 3), similar to previous methods [Soler and Sillion 1998; Durand et al. 2005; Egan et al. 2011]. This parameteriztion allows us to analyze how the relevant signals (distant lighting, occlusion and BRDF) interact, as opposed to using spherical harmonics. We analyze the contribution of each lighting "face" separately, and the final answer is the sum of all face contributions. Our final filter uses the following derivations, but with a simpler geometric measure that does not require reparameterization to cube maps.

We define visibility along a ray as $f(x, v)$ where x is a spatial measure perpendicular to the central axis of projection (Figure 3). We first look at a single planar occluder defined by a binary visibility function $g(x)$ at constant distance z from the receiver. We will extend this to occluders with a range of depths later.

$$f(x, v) = g(x + zv) \ . \quad (3)$$

We now have a re-parameterized BRDF $r(x, v)$ and distant lighting function $l(v)$. With this change of variables we have to adjust for

| (a) occluder spectrum F for varying depth occluder | (b) transport spectrum K for combined BRDF and lighting | (c) step 1: convolve across Ω_x | (d) step 2: integrate over Ω_v | (e) final function $H(\Omega_x)$ defined across Ω_x |

Figure 4: *(a) The occluder spectrum F in the Fourier domain. (b) The combined lighting and BRDF response has small angular extent for low frequency transport (Ω_v^{max}), but may have large spatial extent if the normal varies rapidly (Ω_x^{max}). (c) The inner integral of Equation 7 is a convolution across Ω_x. Note that F effectively becomes bandlimited by the Ω_v^{max} bandlimit of K. (d) The outer integral of Equation 7 integrates over Ω_v. (e) The final result is $H(\Omega_x)$, the Fourier spectrum of the spatial occlusion function h(x).*

the Jacobian $|\partial\omega_i/\partial v|$, which we incorporate into $l(v)$ (note that Jacobian calculations will not be necessary for the final rotationally invariant version of our filter). Now the reflection equation for $h(x)$ is:

$$h(x) = \int r(x,v)l(v)f(x,v)\,dv$$
$$= \int r(x,v)l(v)g(x+zv)\,dv \quad . \tag{4}$$

We now combine the BRDF and lighting $r(x,v)l(v)$ into a new combined response function $k(x,v)$,

$$h(x) = \int k(x,v)g(x+zv)\,dv. \tag{5}$$

3.2 Fourier Analysis

3.2.1 Preliminaries

When we can express a two-dimensional function f in terms of a one-dimensional function g in the form given in Equation 3, previous work has shown that the Fourier transform of $f()$, which we name $F()$, lies along a line segment in the Fourier domain [Shinya 1993; Chai et al. 2000; Durand et al. 2005; Egan et al. 2009]:

$$F(\Omega_x, \Omega_y) = G(\Omega_x)\delta(\Omega_y - z\Omega_x) \quad . \tag{6}$$

where $G()$ is the 1D Fourier transform of $g()$, and δ is the delta function. Intuitively, if you have a 1D function embedded in a 2D domain, it makes sense that the frequency spectrum is also 1D.

If all occluders are planar and lie along a single depth z then the occlusion spectrum corresponds exactly to Equation 6. In practical scenes with a range of depths $[z_{min}, z_{max}]$ the occlusion spectrum F is a double wedge determined by the distance between the occluder and the receiver [Chai et al. 2000; Egan et al. 2011]. The double wedge can be thought of as a spectrum that is swept out by many line segments that correspond to z values between $[z_{min}, z_{max}]$. This is shown in Figure 4a.

3.2.2 Fourier Spectrum for Distant Lighting

We now derive the occlusion spectrum for distant lighting, as well as the interaction between the spectra considering complex occlud-

ers and a surface with a general BRDF and normal maps. Both of these are novel contributions of our paper.

Response Function in the Fourier Domain: The response function spectrum $K(\Omega_x, \Omega_v)$ is shown in Figure 4b. Because r and l are multiplied the spectrum K is the convolution $R \otimes L$. Because L has no spatial dimension we can conclude that the spatial bandlimit of K, Ω_x^{max}, is simply the spatial bandlimit of R. The angular bandlimit of K, Ω_v^{max}, is the sum of the angular bandlimits for R and L.

Normal Mapping in the Fourier Domain: Rapid variation in the normal rotates the BRDF and causes the spectrum K's spatial bandlimit, Ω_x^{max}, to be large. However, rapid rotation in the normal does not affect the angular bandlimit. To see that this is true we can split up the 2D Fourier transform into two 1D transforms, and first take the Fourier transform in v for a given fixed value of x. After transforming in v each slice of the angular transform along $\Omega_v = v_0$ is zero where $v_0 > \Omega_v^{max}$, and therefore the spatial Fourier transform of this slice is zero. Therefore if the BRDF at every x location is bandlimited by Ω_v^{max}, then the final spectrum K will be bandlimited by Ω_v^{max} as well. We call the the portion of the F spectrum that is non-zero after after bandlimiting by Ω_v^{max} the "visible frequencies" of F, as seen in Figure 5.

Lighting and Surface Reflection in the Fourier Domain: Taking the Fourier transform of $h(x)$ in Equation 5 is not trivial because the response function $k(x,v)$ has two dimensions but one of the dimensions is integrated out (see Appendix A for the derivation). In the end we get the following:

$$H(\Omega_x) = \int\left(\int F(\Omega_x - s, -t)K(s,t)\,ds\right)dt, \tag{7}$$

where s is a temporary variable used to compute the inner 1D convolution across Ω_x (shown in Figure 4c). The t variable is used to compute the outer integral across the Ω_v dimension of the resultant spectrum (as seen in Figure 4d). Finally we are left with $H(\Omega_x)$, the spectrum of occlusion across the spatial axis (Figure 4e).

Discussion: The above analysis includes a number of important results. First, we have derived bandlimits for the occlusion spectrum from distant lighting. We split the angular domain of directions into sub-domains, and then reparameterized each sub-domain separately using a linearized angle formulation.

Second, our frequency analysis of occlusion can handle surfaces with a general BRDF and high frequency normal maps. We have shown that the transport spectrum K may have high frequencies in the *spatial domain* due to high frequency changes in the normal. However, an important result is that the visible portion of the occlusion spectrum F is still low frequency in the *angular domain* and is bandlimited by $\Omega_v^{\max}$ (Figure 4c).

Our method directly reconstructs visibility $f(x, v)$ using sparse ray casting. We can use sparse sampling because the compact shape of the sheared filter in the Fourier domain lets us pack Fourier replicas closer together. Our method densely samples the combined lighting-BRDF term $k(x, v)$ which is much cheaper to sample and may include high frequency normal maps. Attempting to share the integrated product $h(x)$ directly, as in irradiance caching, has less benefit because of the possible high spatial frequencies in $H(\Omega_x)$.

3.3 Sheared Filtering Over Linear Sub-Domains

Now that we have defined the sparse shape of the visible occluder spectrum, we can design a sheared filter that compactly captures the frequency content of the signal we care about. We can then transform the filter back to the primal domain where it is used to reconstruct our final answer, while allowing for sparse sampling and the sharing of information across pixels [Chai et al. 2000; Egan et al. 2009; Egan et al. 2011]. The shape of the sheared filter will guide our design of our rotationally-invariant filter in Section 4.

The visible parts of the occlusion spectrum F that we need to capture are shown in Figure 5. The Fourier footprint for a standard reconstruction filter is shown in Figure 5a, and a sheared filter that tightly bounds the visible parts of F is shown in Figure 5b. By applying a scale and shear we can transform the filter shown in Figure 5a to the one shown in Figure 5b.

We can see from the measurements in Figure 5 that, in the Fourier domain, our filter is scaled along the Ω_x axis and sheared in Ω_x per unit Ω_v. In the primal domain we need to scale along the x-axis by the inverse amount, and shear in v per unit x:

$$\text{primalScale} = \frac{2\Omega_{\text{pix}}^{\max}}{\Omega_v^{\max}} \left[\left(\frac{1}{z_{\min}} \right) - \left(\frac{1}{z_{\max}} \right) \right]^{-1}, \qquad (8)$$

$$\text{primalShear} = -\frac{1}{2} \left[\left(\frac{1}{z_{\min}} \right) + \left(\frac{1}{z_{\max}} \right) \right]. \qquad (9)$$

where $\Omega_{\text{pix}}^{\max}$ represents the smallest wavelength that can be displayed. We set $\Omega_{\text{pix}}^{\max}$ to be $(0.5/shadingDiameter)$, meaning that the highest frequency that can be captured is half a wavelength per output diameter. We set $\Omega_v^{\max}$ to be 2.0, which approximates that the BRDF * lighting function can be captured with two wavelengths per unit of linearized angle (45 degrees).

In areas of sharp contact shadows the visible portion of F may extend beyond the $\Omega_{\text{pix}}^{\max}$ bandlimit of the standard filter. In this case we must make sure that our filter does not capture any portion of F outside the $\Omega_{\text{pix}}^{\max}$ bandlimit. We check this by testing if $\Omega_v^{\max}/z_{\min} > \Omega_{\text{pix}}^{\max}$, and if so we revert to brute force Monte Carlo. Our implementation stores separate $z_{\min}$ values for different portions of the hemisphere, see Section 5 for details.

We can now use Equations 8 and 9 to construct a filter that operates over a linearized sub-domain of the sphere (as shown in Figures 5c). Intuitively this takes a nearby sample, and warps it to create an effective ray that originates at the current receiver point, pictured in Figure 5d. However, splitting the sphere into multiple sub-domains requires stitching together the results from each filtering operation.
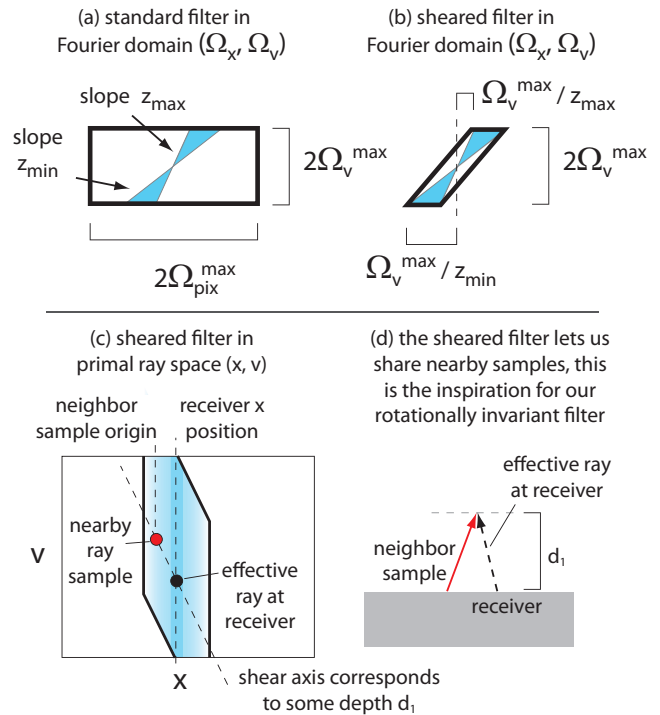
Figure 5: *(a) The visible spectrum of F is shown, as well as the Fourier transform of a standard filter that is axis-aligned in x and v. (b) A sheared filter that tightly bounds the visible parts of F. (c) The theoretical sheared filter graphed in (x, v) space. The sheared filter effectively reprojects rays along the shear axis which is based on harmonic average of depth bounds [$z_{\min}, z_{\max}$]. However, the sheared filter uses a linearized angle v that depends on a fixed axis of projection. (d) A visualization of how a sheared filter shares samples in the primal domain. We use this as inspiration for the rotationally invariant filter presented in Section 4 that does not require a fixed axis of projection.*
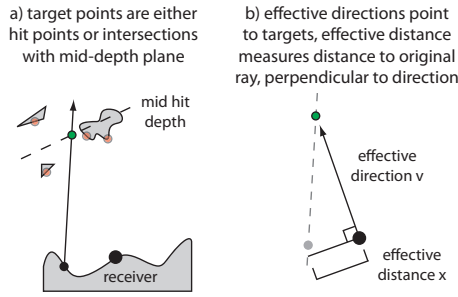
We solve this problem by introducing our rotationally-invariant filter.

4 Rotationally-Invariant Filter

We now develop our rotationally-invariant filter that allows us to easily filter over large angular domains. The sheared filter from the previous section essentially has two outputs: what filter weight to apply to a given sample, and also what is the effective ray direction when we warp a nearby sample to the receiver point. The directional information is useful when we bucket effective ray directions to account for non-uniform sampling densities across the receiver.

Implementing a set of finite linear subdomains has a number of drawbacks. There may be discontinuities where the linear subdomains meet, and we have to account for the Jacobian of the linearization in the transfer function $k(x, v)$. One possible strategy would be to subdivide the sphere into even more subdomains. Of course this may reduce possible artifacts but we would still have to worry about discontinuities and Jacobians. Our approach is to take the limit of subdivision where each sample that we consider is defined to be in its own infinitesimal subdomain. This leads to a rotationally-invariant filter that uses the theory from Section 3 and is easy to implement.

Figure 5d and Equation 9 together show that warping a sample ray

a) target points are either hit points or intersections with mid-depth plane

b) effective directions point to targets, effective distance measures distance to original ray, perpendicular to direction

Figure 6: *(a) To use our rotationally invariant filter filter we first compute target positions for the current receiver. Unoccluded rays use an intersection with a plane that goes through the harmonic average of $[z_{min}, z_{max}]$ for the current receiver. (b) The effective directions for the samples simply connect the current receiver to the target positions. To calculate effective distance we compute the intersection of the original ray and a plane that goes through the current receiver and is perpendicular to the effective direction.*

to an effective ray originating from the current receiver is based on the shear, and the shear in turn is based on the harmonic mean of z_{min} and z_{max}. If we consider each sample to be in its own infinitesimally small subdomain, then for occluded rays we can say that the occluder hit point is the only occlusion distance z that we care about. In this case the effective ray direction is simply the vector from the receiver to the sample hit point. If the ray is unoccluded, we make a ray "target" point where the ray intersects a mid depth plane based on the harmonic mean of occluders in a surrounding region. The calculation of the depth plane will be explained in more detail in Section 5.

To compute a filter weight we must compute the x coordinate for the sample. Using an infinitesimal subdomain our axis of projection is defined to be the same as our effective ray direction. The x measure needs to be tangent to our axis of projection, and we compute the intersection point between the sample ray and plane that is perpendicular to the effective ray direction and goes through the receiver point. The x measure is simply the offset vector from the receiver to this intersection point. Our rotationally-invariant filter with the computed target point for a given sample is shown in Figure 6a, and the filter effective direction and spatial x value are shown in Figure 6b.

Given a ray target point we compute an effective direction that originates from the origin, and then compute the tangent x coordinate for the ray sample. These computations are tightly bound with the derivations in Section 3. We used Equation 9 and the intuitive notion of the filter shear (Figure 5d) to guide setting the effective direction for a sample. The x coordinate and *primalS cale* from Equation 8 are used as the input and scale respectively for computing filter weights.

Discussion: Our rotationally-invariant filter operates on an infinite number of of small subdomains, and it is fair to ask how this affects the earlier Fourier derivations. In Equations 8 and 9 the only Fourier bandlimit that is affected is the transfer function Ω_v^{max} angular bandlimit. Instead of computing Ω_v^{max} for a fixed set of linear subdomains, we can instead precompute the largest Ω_v^{max} over a range of different localized areas and orientations both for each BRDF and for the distant lighting. As stated before, the Ω_v^{max} value for the transfer spectrum K is the sum of angular bandlimits for the BRDF and lighting.

```
// Filtering algorithm for one shading point
// 1.  Calculate hemisphere cell info
```
1 **foreach** s in *S ampleImageCache* **do**
2   **if** *s.isOccluded* **then**
3   *targetPoint* = *s.hitPoint*;
4   **if** *InCorrectHemi(targetPoint)* **then**
5   (x, v) = ComputeFilterCoords(s, *targetPoint*);
6   *cell* = GetCell(v);
7   UpdateCellMinMaxDepth(*cell*, s);
8   **end**
9   **end**
10 **end**
11 **foreach** *cell* in *HemisphereCellArray* **do**
12   *cell.spatialRadius* = ComputeSpatialRadius(*cell*);
13   *cell.midDepth* = ComputeMidDepth(*cell*);
14 **end**

```
// 2.  Filter over neighboring samples
```
15 cellResults = InitializeArray(*numCells*);
16 **foreach** s in *S ampleImageCache* **do**
17   **if** *s.isOccluded* **then**
18   *focusPoint* = *s.hitPoint*;
19   **else**
20   *initCell* = GetInitialCell(s);
21   *targetPoint* = ComputeTarget(s, *initCell*);
22   **end**
23   **if** *InCorrectHemi(targetPoint)* == FALSE **then** continue;
24   (x, v) = ComputeFilterCoords(s, *targetPoint*);
25   *cell* = GetCell(v);
26   **if** *IsMarkedForBruteForce(cell)* **then** continue;
27   *weight* = ComputerFilterWeight(x / *cell.spatialRadius*);
28   (*BRDF*,*light*) = ComputeBRDFandLighting(v);
29   AddWeightedSample(*cellResults*[*cell.index*],
30   *weight*, *BRDF*, *light*, *s.visibility*);
31 **end**
33 NormalizeWeights(cellResults);
35 **return** *cellResults*

Algorithm 1: *Our algorithm for filtering results at each shading point (this is implemented in our C++ RenderMan plugin). For each shading point we return an array of values, one per hemisphere cell. Each value either represents a filtered BRDF * lighting value, or a flag that tells the calling RenderMan shader to brute force the corresponding hemispherical cell.*
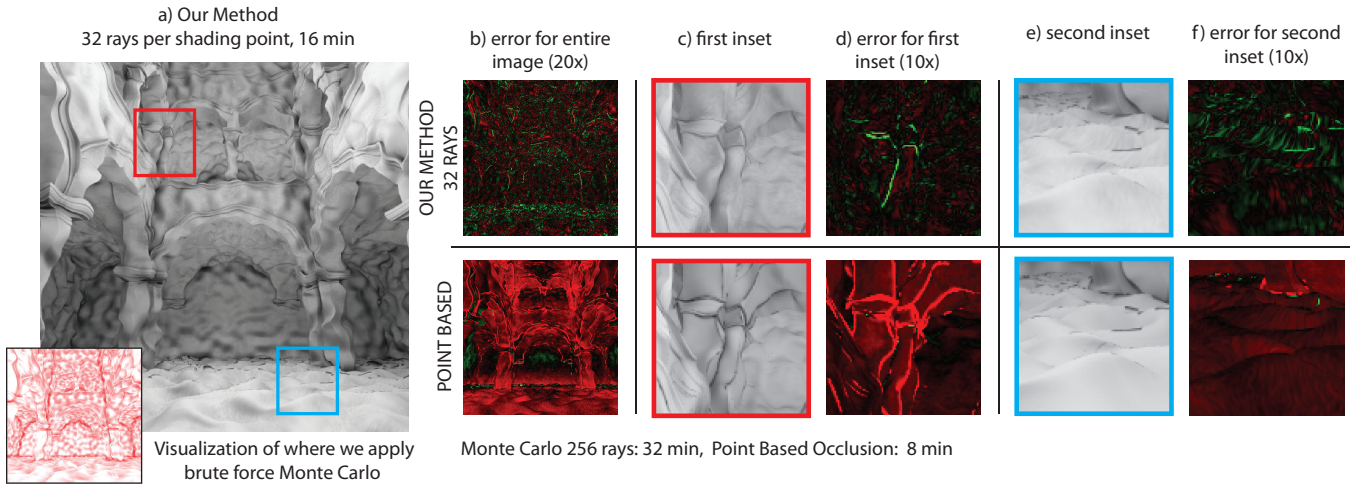
5 Implementation

Our implementation takes a two pass approach: sampling followed by filtering. Our first pass shoots a sparse set of rays (4 - 32 rays per shading point) and writes the rays to a point cloud. This is similar to common Monte Carlo implementation, except for the low number of rays used (Figure 2b). The second pass reads in the point cloud and filters over a large number of nearby ray samples at each shading point to compute a smooth and accurate result (Figure 2c).

We implemented our algorithm by writing shaders and a C++ plugin for a RenderMan compliant renderer [Pixar 2005]. The core filtering algorithm runs inside our plugin for each shading point and can be seen in Algorithm 1. At a high level we compute depth bounds for different sections of the hemisphere (lines 1 to 14 in Algorithm 1), then use these depth bounds to filter neighboring samples (lines 15 to 35).

Computing Bounds for the Hemisphere The first thing we do is compute the $[z_{min}, z_{max}]$ depth bounds for the current receiver

a) Our Method
32 rays per shading point, 16 min

b) error for entire image (20x)

c) first inset

d) error for first inset (10x)

e) second inset

f) error for second inset (10x)

OUR METHOD 32 RAYS

POINT BASED

Visualization of where we apply brute force Monte Carlo

Monte Carlo 256 rays: 32 min, Point Based Occlusion: 8 min

Figure 7: *This scene shows complex occluders and receivers with displacement. (a) Our method with 32 rays per shading point (9 rays in the first pass, 23 rays in the second pass). As an inset we also show a visualization of where our method reverted back to brute force Monte Carlo. We decide whether to filter over neighboring samples or use Monte Carlo at each hemisphere cell, so many shading points use a mix of both methods. In (c) and (d) the error in our method comes from overblurring and missing occlusion within some creases. Point based occlusion has larger error and over darkens the creases. Green error values are areas that are too bright, red error values are areas that are too dark. In (e) and (f) our method can be seen to have some noise (due to some cells of the hemisphere requiring Monte Carlo integration). The point based result is consistently too dark, but smoother.*

(lines 1 to 14). To compute tighter depth bounds we divide the hemisphere of visible directions into cells of equal projected area and compute depth bounds per cell (our implementation subdivides the disk into 8x8 cells). We also filter results per cell, which helps to compensate for possible non-uniform sample densities.

In our implementation, a user-specified screen space radius determines the set of potential neighbor samples (for our results we used a radius of 8-16 pixels). For each neighbor sample that is occluded and whose hit point is in the correct hemisphere, we compute the effective sample distance x and effective direction v (line 5 as described in section 4). We then use the direction v to lookup the appropriate cell (line 6) and update the cell's depth bounds (line 7).

After all samples are processed, we compute the spatial radius and mid depth for each cell (lines 12 and 13). We compute the spatial radius by multiplying the current micro polygon diameter by `primalScale` (from Equation 8). In Equation 9 we can see that the shear value is simply the harmonic average of the minimum and maximum depths. Therefore we store the harmonic average of z_{min} and z_{max} for the cell's mid depth. If a cell has zero samples, or if $\Omega_v^{max}/z_{min} > \Omega_{pix}^{max}$ (see Section 3), the `ComputeSpatialRadius()` function marks the cell as requiring brute force computation.

Filtering Samples Now that we have $[z_{min}, z_{max}]$ depth bounds at each cell we can filter over neighboring samples (lines 15 to 35 in Algorithm 1). We first compute the target point of each sample (lines 17 to 22). For occluded samples, the target point is simply the hit point of the occluded ray (line 18). For non-occluded samples, we use a two step process to compute the target point. We first lookup an initial cell based purely on the ray direction (line 20). We construct a plane that is perpendicular to the central direction of the cell and whose distance to the current receiver is the same as the mid depth of the cell. We set the target point to be the intersection of the sample ray with this plane (line 21).

Once we have the target point we can compute the final sample distance x and effective direction v (line 24). Using the direction v we can lookup the final cell for the sample (line 25). Using the sam-

ple distance x and the cell's spatial radius we can compute the filter weight (line 27). Using the effective incoming light direction v we can also compute the BRDF and lighting response (line 28). Many implementations let the user fade out occlusion beyond a certain distance, so that distant occluders are not counted [McGuire 2010]. This is especially necessary for indoor scenes. We have incorporated this in our method by testing the distance from the receiver to the sample target point and reducing visibility accordingly.

After we have processed all of the samples, we normalize the total contribution per cell based on the weight in each cell. Any cells with too little weight (we use a weight that corresponds to approximately 2 samples) are marked as requiring brute force computation. This is the end of Algorithm 1, and where our C++ plugin hands control back to the RenderMan shader. The shader then goes through each cell, and uses Monte Carlo ray tracing to compute answers for any cells that needed to be brute forced (for our results we shot a single ray to estimate these cells).
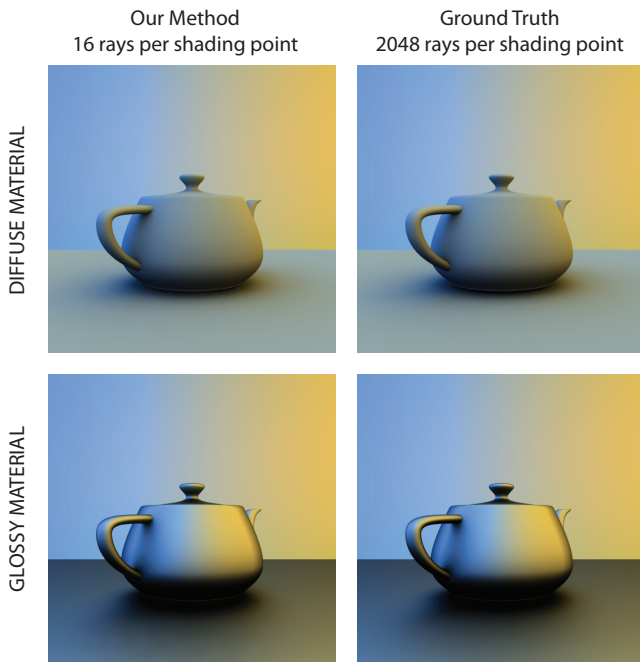
6 Results

6.1 Setup

All results are 512x512 and were generated on a dual quad-core Xeon 2.33 GHz processor with 4 GB of memory using Pixar's RenderMan Pro Server 15.2. Our plug-in is thread safe and is designed to run in parallel (we used 8 threads for our results).

We used the `trace()` call in RenderMan which finds the nearest hit point regardless of occluder surface. Other functions such as the `occlusion()` function have options to cap the maximum distance of a ray, which could reduce the working set of ray-traced geometry and reduce paging to disk. Of course, with more dense geometry, the problem of paging to disk will easily reoccur. The San Miguel scene (Figure 1) is the only scene that was computationally bound by paging to disk, and the Sponza scene (Figure 7) was limited by computation (ray tracing and displacement).

Our Method
16 rays per shading point

Ground Truth
2048 rays per shading point

DIFFUSE MATERIAL

GLOSSY MATERIAL

Figure 8: *Here we show an example of our method with both matte and glossy BRDFs with environment lighting.*

6.2 San Miguel

In the San Miguel scene (Figure 1a) we show a complex scene that contains both large smooth areas, and areas of very high complexity. The smooth areas are challenging because the human visual system is drawn to any small errors or oscillations. Accurately calculating occlusion for areas of high geometric detail and/or contact shadows is difficult because these areas can easily be missed or undersampled.

The scene is 5.2 million triangle faces. Because of the high geometric complexity of the scene RenderMan generated and shaded an average of 5 micro polygons per pixel. Because of the complexity of the scene and the incoherent nature of the rays, geometry was constantly paged in and out of memory. While it is always possible to increase memory on a machine, artists will in turn keep producing larger models. RenderMan reported using 5.5GB of virtual memory (with 2GB devoted to ray-traced geometry) and 3.5GB of physical memory. This scene is well suited for our method because the cost per ray is very high relative to the cost of filtering over nearby ray samples.

In Figure 1, we show equal time and equal quality comparisons with stratified Monte Carlo sampling. Our method took 1 hr and 48 min, spending about 40% of the time in the first pass casting sparse ray samples (13 rays per shading point), and the remaining time in the second pass filtering and casting rays for areas with very close occluders (19 rays per shading point). Our method gave a 4x speed up over Monte Carlo with 256 samples (7 hrs 4 min) as well as an order of magnitude reduction in the number of ray casts. We also show significantly less noise versus Monte Carlo with 40 samples using equal time. Our method is smoother than Monte Carlo with 256 samples in many areas, although our method does have some areas with noise (inside the lip of the fountain) and overblurring (contact shadows with the leaves on the ground). See section 6.6 for more discussion on limitations and artifacts.

The filtering operation in our method is more expensive than simple

Monte Carlo, which accounts for the discrepancy between our reduction in rays and speed up. One reason for the increased filtering cost is the increased algorithmic complexity of our filtering (shown in Algorithm 1). Another reason is that our filter often needs more samples to achieve a smooth result due to samples being unstratified when they are warped onto the hemisphere of the current receiver.

In Figure 1e we also show that our method can output spherical harmonic occlusion. Our RenderMan shader takes the cell reflection values returned from Algorithm 1, calculated per hemispherical cell, and uses environment maps that represent low order spherical harmonic basis functions $[s_0(\omega_i), ..., s_n(\omega_i)]$ as the distant lighting $l(\omega_i)$ for each cell (Equation 2). As of version 15.2 RenderMan does not support an efficient method for producing spherical harmonic occlusion using its point based algorithm.
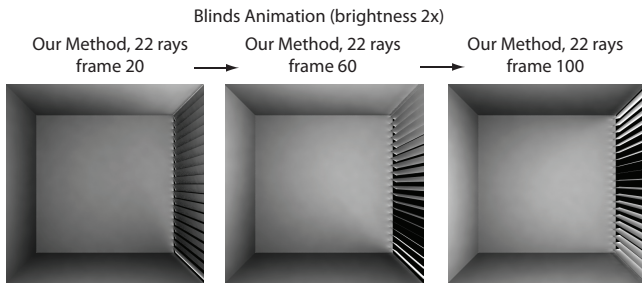
6.3 Bumpy Sponza

In the bumpy Sponza scene we apply a displacement shader to show that our method can handle complex occluders as well as high frequency changes in receiver surface and normal orientation. The scene only has 300,000 triangles (before displacement) and fit inside memory during our renders. In Figure 7 we compare the quality of our method with point based occlusion. We use the RenderMan point based occlusion implementation with high quality settings (6x increase in micro polygon density for initial geometry pass, clamping turned on, rasterresolution set to 32, max solid angle set to 0.01). For this scene our method used 32 rays per shading point and took 16 minutes, while Monte Carlo used 256 rays and took 32 minutes. Our method reduces the number of ray casts by an order of magnitude, and is still 2x faster than Monte Carlo with 256 samples even when the scene fits in memory and the cost per ray is low.

Point based occlusion is a popular solution because it is fast and the results are generally smooth (in this example point based occlusion took 8 minutes). However, it can also have a number of disadvantages. In Figure 7d, we visualize the image error for our method and point based occlusion (red is used for areas that are too dark, and green for areas that are too bright). We can see that in some of the crease areas point based occlusion can produce results that are too dark. Our method is slightly too bright due to overblurring and missing some details, but the amplitude of our errors is much less.

In Figures 7e and 7f our method has noise in some areas. Because of the high frequency displacement many surfaces have nearby occluders which can trigger our method to fallback to brute force Monte Carlo. A visualization of where and to what degree our method used Monte Carlo sampling can be seen in Figure 7a. In this scene 74% of shading points used Monte Carlo for less than half of their hemispherical cells. The higher the Ω_v^{max} is set, the more often our method falls back to Monte Carlo. Point based occlusion is consistently slightly too dark in this area, but it does produce smooth results.

6.4 Glossy

In Figure 8, we show a glossy teapot scene demonstrating our method's ability to handle diffuse and glossy surfaces with spherical harmonic lighting. The different colored shadows to the left and right of the teapot show that we are capturing directional occlusion. The shape of the shadow on the ground plane also changes as the material goes from diffuse to glossy. Our method very closely matches ground truth with either a diffuse or glossy BRDF and environment lighting.

Blinds Animation (brightness 2x)

Our Method, 22 rays
frame 20

Our Method, 22 rays
frame 60

Our Method, 22 rays
frame 100

Figure 9: *Frames from our supplementary video (brightness multiplied by 2). Our method accurately captures the angular content of the small slats.*

6.5 Blinds Animation

We also provide a supplemental animation (frames shown in Figure 9) that shows a set of blinds rotating together. The angular content of the occluders is very important in this example, and we show that our method still handles this well. Our method has a low amplitude of error overall (Figures 10a and 10b). At object boundaries the non-uniform distribution of samples can lead to some small errors (in this case the edges of slats are too bright). The point based solution is smooth, but the results are consistently too dark (Figures 10d and 10e).

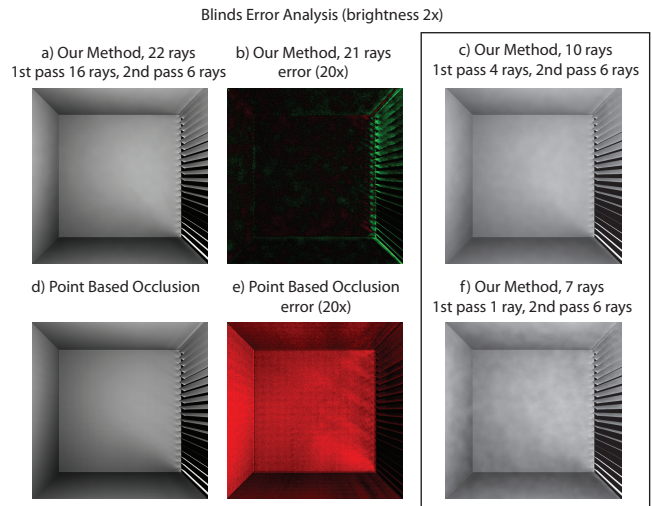### 6.6 Limitations and Artifacts

We now examine the limitations and possible artifacts that can occur in our method. In Figures 10c and 10f we show the artifacts that can occur when we don't use enough samples in the first pass of our algorithm (we use 4 and 1 rays per shading point for the first pass in Figures 10c and 10f respectively). Because of our wide filtering, any undersampling in the first pass shows up as low amplitude medium frequency error instead of high frequency noise. While the amplitude of these errors is often low, spurious changes in the derivative can be visually noticeable, especially in areas of constant or linearly changing occlusion. Raising the sample count in the first pass of our algorithm reduces, and at a certain point, eliminates this problem.

In areas where we revert to brute force computation, our method can show noise, such as in Figure 1a near the lip of the fountain. We can again reduce noise in these areas by increasing sampling density, but this will in turn reduce performance. Our method can also smooth out some areas of detail. In Figure 1a some of the contact shadows under the leaves and around the detailed geometry of the door are slightly washed out as compared to ground truth.

7 Conclusion and Future Work

We have presented a new frequency analysis for occlusion that incorporates distant lighting, general BRDFs, and high frequency normal maps for complex receivers and occluders. In addition, we have also provided a new rotationally-invariant filter that is parameterized according to our analysis, and that is capable of sharing samples across a large angular domain. Our results show that our method can substantially reduce the number of rays cast, and can lead to large speed up in scenes that are computationally bound by ray tracing costs.

For future work we want to investigate methods for stratifying samples in such a way that the results will be stratified across multiple receivers. We also want to investigate alternative ways to integrate

Blinds Error Analysis (brightness 2x)

a) Our Method, 22 rays
1st pass 16 rays, 2nd pass 6 rays

b) Our Method, 21 rays
error (20x)

c) Our Method, 10 rays
1st pass 4 rays, 2nd pass 6 rays

d) Point Based Occlusion

e) Point Based Occlusion
error (20x)

f) Our Method, 7 rays
1st pass 1 ray, 2nd pass 6 rays

Figure 10: *We show the errors with point based occlusion and our method for the blinds scene frame 60 (ambient occlusion brightness is multiplied by 2x, green error values are areas that are too bright, red error values are areas that are too dark). (a) (b) Our method produces a scene with overall low amplitude error. (d) (e) Point based occlusion produces an image that is consistently too dark. (c) (f) Output from our method using a reduced ray count. The medium frequency noise is due to too few rays stored in the first pass (4 and 1 rays in (c) and (f) respectively).*

over sample point sets. If we could improve our accuracy when dealing with smaller point sets of non-uniform density, we could reduce our filter radius and speed up the filtering process substantially.

In summary, directional occlusion is of increasing importance in Monte Carlo rendering. We have taken an important step towards fully exploiting the space-angle coherence. We expect many further developments in this direction, based on a deeper analysis of the characteristics of the occlusion function.

Acknowledgements

References

ACADEMYAWARDS, 2010. Scientific and Technical Achievements to be Honored with Academy Awards. http://www.oscars.org/press/pressreleases/2010/20100107.html.

ARIKAN, O., FORSYTH, D. A., AND O'BRIEN, J. F. 2005. Fast and Detailed Approximate Global Illumination by Irradiance Decomposition. *ACM Trans. on Graph. (SIGGRAPH) 24*, 1108–1114.

BAVOIL, L., AND SAINZ, M. 2009. *ShaderX7 - Advanced Rendering Techniques*. ch. Image-Space Horizon-Based Ambient Occlusion.

CHAI, J., TONG, X., CHAN, S., AND SHUM, H. 2000. Plenoptic Sampling. In *Proceedings of SIGGRAPH 2000*, 307–318.

CHRISTENSEN, P. H. 2008. Point-Based Approximate Color Bleeding. Tech. Rep. 08–01, Pixar Animation Studios.

DURAND, F., HOLZSCHUCH, N., SOLER, C., CHAN, E., AND SILLION, F. X. 2005. A Frequency Analysis of Light Transport. *ACM Transactions on Graphics (SIGGRAPH 05) 24*, 3, 1115–1126.

EGAN, K., TSENG, Y.-T., HOLZSCHUCH, N., DURAND, F., AND RAMAMOORTHI, R. 2009. Frequency Analysis and Sheared Reconstruction for Rendering Motion Blur. *ACM Trans. on Grap. (SIGGRAPH) 28*, 3, 93:1–93:13.

EGAN, K., HECHT, F., DURAND, F., AND RAMAMOORTHI, R. 2011. Frequency Analysis and Sheared Filtering for Shadow Light Fields of Complex Occluders. *ACM Trans. on Graph. 30*, 2, 9:1–9:13.

HAŠAN, M., PELLACINI, F., AND BALA, K. 2007. Matrix row-column sampling for the many-light problem. *ACM Trans. Graph. 26*, 3, 26:1–26:10.

HUANG, F.-C., AND RAMAMOORTHI, R. 2010. Sparsely Precomputing the Light Transport Matrix for Real-Time Rendering. *Computer Graphics Forum (EGSR 10) 29*, 4.

KŘIVÁNEK, J., GAUTRON, P., PATTANAIK, S., AND BOUATOUCH, K. 2005. Radiance caching for efficient global illumination computation. *IEEE Trans. on Visualization and Computer Graphics 11*, 5.

LAINE, S., AND KARRAS, T. 2010. Two methods for fast ray-cast ambient occlusion. *Computer Graphics Forum (EGSR 10) 29*, 4.

LANDIS, H. 2008. Production ready global illumination. In *ACM SIGGRAPH Course Notes: RenderMan in Production*, 87–102.

LANMAN, D., RASKAR, R., AGRAWAL, A., AND TAUBIN, G. 2008. Shield Fields: Modeling and Capturing 3D Occluders. *ACM Transactions on Graphics (SIGGRAPH Asia) 27*, 5.

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.

MCGUIRE, M. 2010. Ambient Occlusion Volumes. In *Proceedings of High Performance Graphics 2010*, 47–56.

MÉNDEZ-FELIU, A., AND SBERT, M. 2009. From Obscurances to Ambient Occlusion: A Survey. *Vis. Comput. 25*, 2, 181–196.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2003. All-Frequency Shadows Using Non-Linear Wavelet Lighting Approximation. *ACM Trans. on Graph. (SIGGRAPH 03) 22*, 3, 376–381.

PANTALEONI, J., FASCIONE, L., HILL, M., AND AILA, T. 2010. PantaRay: Fast Ray-Traced Occlusion Caching of Massive Scenes. *ACM Trans. on Graph. (SIGGRAPH 10) 29*, 4, 37:1–37:10.

PIXAR, 2005. The RenderMan Interface, version 3.2.1. https://renderman.pixar.com/products/rispec/rispec_pdf/RISpec3_2.pdf.

RAMAMOORTHI, R., KOUDELKA, M., AND BELHUMEUR, P. 2005. A fourier theory for cast shadows. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 2, 288–295.

SHINYA, M. 1993. Spatial anti-aliasing for animation sequences with spatio-temporal filtering. In *SIGGRAPH '93*, 289–296.

SLOAN, P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM TOG (SIGGRAPH 02) 21*, 3.

SOLER, C., AND SILLION, F. 1998. Fast Calculation of Soft Shadow Textures Using Convolution. In *Proceedings of SIGGRAPH 98*, ACM Press / ACM SIGGRAPH, 321–332.

WARD, G., RUBINSTEIN, F., AND CLEAR, R. 1988. A Ray Tracing Solution for Diffuse Interreflection. In *SIGGRAPH 88*, 85–92.

ZHUKOV, S., IONES, A., AND KRONIN, G. 1998. An ambient light illumination model. In *Rendering Techniques (Eurographics 98)*, 45–56.

Appendix A: Fourier Derivations

We temporarily define h and H to be 2D to make the derivation more concise (this allows us to use a 2D convolution operator). The second angular dimension for both of these functions will not be important, and our final step will be to reduce H to 1D. We define $h(x, v)$ to be constant across v, such that $\forall v \in \mathbb{R}, h(x, v) = h(x, 0)$.

$$m(x, v) = \delta(x) \tag{10}$$

$$h(x, v) = \int f(x, t)k(x, t)dt \tag{11}$$

$$h(x, v) = \int \int (f(s, t)k(s, t)) \, m(x - s, v - t)ds\,dt \tag{12}$$

$$h(x, v) = (fk) \otimes m \tag{13}$$

Because $h(x)$ is constant across v it is not surprising that spectrum H will only have frequencies along the $\Omega_v = 0$ line.

$$M(\Omega_x, \Omega_v) = \delta(\Omega_v) \tag{14}$$

$$H(\Omega_x, \Omega_v) = (F \otimes K)M \tag{15}$$

$$H(\Omega_x, \Omega_v) = \int \int F(\Omega_x - s, \Omega_v - t)K(t, s)\delta(\Omega_v) \, ds \, dt \tag{16}$$

To reduce a 2D spectrum to 1D we must integrate across the dimension we want to remove. We integrate out the Ω_v dimension to measure the spatial frequencies of H along the Ω_x axis:

$$H(\Omega_x) = \int \int F(\Omega_x - s, -t)K(s, t) \, ds \, dt \tag{17}$$