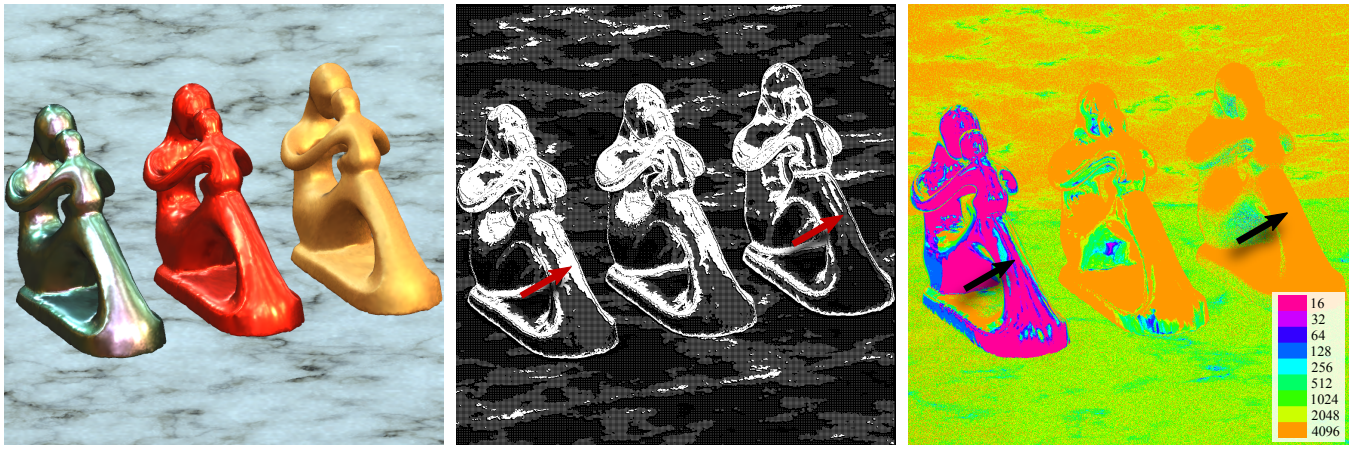
Fast multi-resolution shading of acquired reflectance using bandwidth prediction

Mahdi Mohammadbagher, Cyril Soler, Kartic Subr, Laurent Belcour, Nicolas Holzschuch

Fast multi-resolution shading of acquired reflectance using bandwidth prediction

Mahdi M.Bagher Cyril Soler Kartic Subrr Laurent Belcour Nicolas Holzschuch
INRIA Grenoble Rhône-Alpes



(a) Rendered image (b) Pixels actually computed (c) Light samples per pixel

Figure 1: *We accelerate the shading of acquired materials by selectively shading image pixels (shown in white in the middle image) and adapting the number of samples used in the integration across pixels (right). The figure shows three identical objects rendered under environment lighting with different acquired materials: a color-changing paint, red car paint and gold paint [Matusik et al. 2003]. More pixels per area are shaded for the specular materials (red arrows) but a larger number of samples are used to compute shading on the diffuse material (black arrows).*

Abstract

Shading complex materials such as acquired reflectances in multi-light environments is computationally expensive. Estimating the shading integral involves stochastic sampling of the incident illumination independently at several pixels. The number of samples required for this integration varies across the image, depending on an intricate combination of several factors. Ignoring visibility, adaptively distributing computational budget across the pixels for shading is already a challenging problem.

In this paper we present a systematic approach to accelerate shading, by rapidly predicting the approximate spatial and angular variation in the local light field arriving at each pixel. Our estimation of variation is in the form of *local bandwidth*, and accounts for combinations of a variety of factors: the reflectance at the pixel, the nature of the illumination, the local geometry and the camera position relative to the geometry and lighting. The speed-up, using our method, is from a combination of two factors. First, rather than shade every pixel, we use this predicted variation to direct computational effort towards regions of the image with high local variation. Second, we use the predicted variance of the shading integrals, to cleverly distribute a fixed total budget of shading samples across the pixels. For example, reflection off specular objects is estimated using fewer samples than off diffuse objects.

1 Introduction

Complex materials used in the real world can exhibit subtle and rich shading effects, such as colors changing depending on the position of the viewer (see Figure 1). Designers and artists would benefit from editing the geometry and reflectance properties of an object, under varying lighting conditions, with interactive results. While recent work has addressed this problem, the combination of varying geometry and arbitrary BRDFS in real-time is still an open research problem. Existing algorithms can simulate dynamic geometry with simple material models (such as Phong), or they can precompute radiance transfer in a static scene, but editing the geometry is difficult.

Simulating material-appearance under any illumination requires, at each pixel, the computation of an integral of the product of the material's reflectance with the incoming illumination. The numerical estimation of these integrals involves sampling the integrand and the extent of sampling required depends on the properties of the material: diffuse materials require many samples over incoming directions, but exhibit low variation between neighboring pixels, specular materials require fewer samples over incoming directions but dense sampling in pixel-space.

In this paper, we present an algorithm to systematically predict the number of samples required for reconstruction in pixel-space and for integration over incident illumination directions at every pixel. Our prediction of sampling rates takes into account the reflectance, illumination, local geometry and camera position. We use them in an interactive multi-resolution rendering algorithm of objects with acquired material reflectances.

Our algorithm consists of two steps. First, for each pixel of the picture, we compute the spatial and angular variations. This information is stored, hierarchically, in a buffer having the same size as the picture generated. In a second step, this information is used for optimal sampling of the image. Areas with low image-space frequency will have a small number of pixels sampled, while pixels with high expected variance will be estimated with a large number of samples for their integrals. The spatial samples are combined together in an upsampling pass.

Our contributions are the following:

- *Frequency propagation*: We present a fast algorithm that pre-

dicts the local screen space frequency information of reflected illumination, with very small overhead.

- *Reflectance bandwidth-estimation*: We describe an algorithm for estimating the local bandwidth of arbitrary reflectance functions; this is a key element of the frequency propagation algorithm.

- *Multiresolution shading*: We describe a fast multiresolution deferred shading algorithm that uses this frequency information for optimal sampling. We reduce both spatial sampling (shading only selected pixels) and angular sampling (adapting the number of light samples for each shaded pixel)

Our algorithm results in optimal sampling for arbitrary geometry, reflectance and incoming light, resulting in interactive rendering framerates. It does not rely on pre-computations, and thus works with dynamic geometry and materials. It does not, however, take into account effects such as visibility for shadows or secondary light bounces. Including these effects would require independent attention and they are kept as avenues for future work.

1.1 Related Work

Deferred shading Common rendering methods on the GPU apply shading to fragments as they are rasterized, before testing for occlusion. This strategy is inefficient for computationally expensive shaders. *Deferred shading*, on the other hand, postpones shading until occlusions are resolved in image space (see, e.g. [Deering et al. 1988]). Both approaches allow dynamic geometry, but are expensive when combining complex material data with many lights. Deferred shading is also incompatible with multi-sampling anti-aliasing methods [Fatahalian et al. 2010]. Our algorithm allows adaptive sampling with complex materials and incoming light, and efficient multi-sample antialiasing.

GPU rendering of complex materials Heidrich and Seidel [1999] interactively rendered simple BRDFs with environment maps by pre-filtering the environment map. Interactively rendering arbitrary BRDFs has been done using separable approximations [Kautz and McCool 1999], homeomorphic factorization [Latta and Kolb 2002], spherical harmonics compactness properties [Kautz et al. 2002] and spectral properties [Ramamoorthi and Hanrahan 2002]. Wavelet encoding of BRDFs [Claustres et al. 2007] also provides real time rendering of acquired data by taking advantage of the sparsity of the wavelet representation and low rank of the reflection operator. Wavelet importance sampling [Clarberg et al. 2005] provides an elegant way of integrating products of complex functions for rendering measured materials. Compared to all these works, our main contribution is that we only compute shading for a subset of the pixels, spreading computations to neighboring pixels.

Fourier analysis of light transport Durand *et al.* [2005] study the properties of the Fourier spectrum of the local light field around a central ray. They derive transformations that propagate these spectra. Subsequent work has provided interesting applications of this theory, including motion blur [Egan et al. 2009] and depth of field [Soler et al. 2009]. They however do not provide any mechanism by which frequency content can be efficiently propagated to image space in real time. Propagating sampled spectra, as done by Soler et al. [2009], is costly. On the other hand, Egan et al. [2009] derive formulae for transformations to the 3D lightfield (space and time) without considering the angular component. Our approach builds upon these works, but we propose a rapid method to only estimate maximum variation along space and angle, rather than estimate entire spectra.

Multiresolution screen-space algorithms To avoid shading at every pixel, researchers have developped multi-resolution rasterization techniques, combined with upsampling. Multiresolution splatting for indirect illumination [Nichols and Wyman 2009] and hierarchical image space radiosity [Shopf et al. 2009] use a fast virtual point light source approach for indirect illumination. They hierarchically combine rasterized images using bilateral upsampling [Kopf et al. 2007; Durand et al. 2005]. Other applications of similar techniques include computing interactive lighting from dynamic area light sources [Nichols et al. 2010], and indirect illumination in glossy scenes [Soler et al. 2010; Nichols and Wyman 2010]. Light gathering methods can efficiently be improved using GPU-friendly interleaved sampling [Segovia et al. 2006]. Ritschel *et al.* [2009] achieve global illumination on GPU. They do not fully take advantage of the bandwidth of the reflectance or illumination [Ki and Oh 2008; Shopf et al. 2009].

Precomputed transport Precomputed Transport Algorithms address arbitrary geometry and incoming light by compressing the transport operator in an alternative basis [Sloan et al. 2002; Ramamoorthi 2009]. Although they can depict rich materials [Sun et al. 2007] their primary goal is to precompute effects such as soft-shadows and global illumination as a function of the illumination. Consequently they require that the geometry remains static. In comparison to these algorithms, we are restricted to first bounce radiance without visibility for shadows, but impose no restrictions on the geometry.

1.2 Overview

In this paper we compute direct reflected radiance, ignoring visibility. We refer to this computation as shading. The radiance arriving at each pixel p after one-bounce direct reflection at a point x, that projects to p, is

$$L_p = \int_{\Omega_x} L_i(\omega)\ \rho(x, \omega, \omega_{x\to p})\ \omega.\mathbf{n}(x)\ \mathrm{d}\omega. \qquad (1)$$

Here $\omega_{x\to p}$ denotes the direction from x to the eye through pixel p, $\mathbf{n}(x)$ is the normal at x, L_i is radiance from distant illumination and Ω_x is the set of incident directions on the hemisphere above the local tangent plane, and rho is the reflectance function. This integral is typically estimated using Monte Carlo estimators as an average of N_p illumination samples:

$$L_p \approx \frac{G}{N_p} \sum_{i=1}^{N_p} \frac{L_i(\omega_i)}{g(\omega_i)}\ \rho(x, \omega_i, \omega_o)\ \omega_i.\mathbf{n}(x) \qquad (2)$$

where the ω_i are random variates drawn from the importance function $g(\omega_i)$ over the hemisphere of incident directions and $G = \int_{\Omega_x} g(y)\mathrm{d}y$.

We accelerate rendering by changing two aspects of typical approaches. First, *we avoid shading every pixel*, thus computing the integral only at pixels with large local variation. At other pixels, we upsample the estimate from neighboring shaded pixels (sec. 3.2). Second, for each pixel p where the integral is estimated, *we adaptively choose* N_p according to the predicted variance of the shading integrand (sec. 2.3). Our method is independent of the importance function used, and therefore complementary to approaches that propose importance functions [Clarberg et al. 2005; Subr and Arvo 2007] for shading.

We present a multiresolution shading algorithm (Section 3) that exploits the calculated bandwidths. For each frame, we first compute the image space bandwidth and sampling density using the input material, screen-space normals, depth and curvature (G-buffers). Next, starting from the coarsest resolution, we shade only those pixels whose screen-space bandwidth corresponds to the current resolution, and upsample previously shaded pixels with an anisotropic kernel.

2 Real-time bandwidth estimation

Rather than estimating and using complete spectra, we only calculate the 2D *bandwidth* of the local light field about a ray to indicate local spatial and angular variation. We demonstrate that computing the 2D bandwidth is sufficient to adaptively sample the image and the various shading integrals.

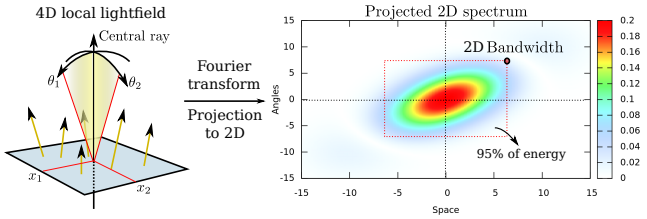


Figure 2: Left: *Parametrization adopted by Durand [2005] for the 4D light field along a ray.* Right: *The spectrum projected to 2D as used by Soler [2009]. We define the* bandwidth *(dashed red box) of the local light field as the frequencies that enclose* 95% *of the energy in the power spectrum of the lightfield.*

2D bandwidth We analyze the local lightfield using the parametrization proposed by Durand et al [2005]. While their parametrization is in 4D, we define the bandwidth of the local lightfield as a 2D vector with the maximum non-zero Fourier frequencies in the spatial and angular components. For robustness, in practice, we use a quantile (the 95^{th} percentile) of the power spectra rather than the absolute maximum. For example, according to our definition, the bandwidth of a signal with a unit normal power spectrum is $\sqrt{2}\ \mathrm{erf}^{-1}(2 * 0.95 - 1)$. For non-bandlimited signals, we store an arbitrarily large value until the final calculation in image-space, where we clamp to the maximum representable frequency, which depends on the extent of anti-aliasing chosen. We denote the bandwidth using

$$\nu = \begin{bmatrix} \nu_s \\ \nu_a \end{bmatrix}$$

so that the rectangle with opposite corners $(-\nu_s, -\nu_a)$ and (ν_s, ν_a) contains the 2D spatio-angular spectrum of the local light field around a central ray (fig. 2).

Using the results of Durand et al [2005], we derive the transformations undergone by ν at each step of the transport process (see fig. 3 for a review of these transformations). Then we describe how to derive sampling rates using the bandwidth information. Finally, we explain how to estimate the variance of the shading integral, for adaptive sampling.

	4D ray space	*Fourier domain*
Transport (free space)	spatial shear	angular shear
Occlusion	product	convolution (spatial)
Curvature	angular shear	spatial shear
BRDF	convolution (angular)	product
Texture	product	convolution (spatial)

Figure 3: *Review: The table summarizes the effect of transport operators on 4D local light fields and their spectra..*

2.1 Computing one-bounce 2D bandwidth

At the light: The bandwidth of the local light field leaving light sources depends on the geometry and emission of the light sources. For distant illumination, ν_s is zero and ν_a is directly computed from the environment map (see Section 2.4 for details).

Transport through free space: Since transport through free space results in an angular shear of the local light field's spectrum [Durand et al. 2005], the transported bandwidths can be written as $T_d\nu$ for transport by a distance of d (See Figure 5), where T_d is defined in fig 4

Reflection: In the frequency analysis framework, reflection is decomposed into four basic steps [Durand et al. 2005]:

1. The incident lightfield is re-parameterized into the local coordinate system of the reflecting surface, which induces a spatial scale of $\cos\theta_i$ of the spectrum, followed by a spatial curvature shear of length c in the frequency domain (c is the gaussian curvature of the surface expressed in m^{-1});
2. The product with the incident cosine, is an angular convolution in Fourier space with a Bessel function;
3. The angular convolution of the light field with the BRDF is a band-limiting product in the frequency domain, while the spatial product by the texture is a convolution by the spectrum of the texture in the Fourier domain.
4. The lightfield is re-parameterized along the outgoing direction, which successively induces a mirror reflexion in the spatial domain, followed by a spatial curvature shear of length $-c$, and a spatial scale of $1/\cos\theta_x$.

These operations can be translated into matrix operations onto the bandwidth vector ν of the incident local light field. The spatial scales by $\cos\theta_i$ and $1/\cos\theta_x$ of the incident-to-plane and outgoing reparametrization correspond to multiplying ν by scaling matrices P_i and P_x (see fig 4).

$$T_d = \begin{bmatrix} 1 & 0 \\ -d & 1 \end{bmatrix} \quad P_x = \begin{bmatrix} \frac{1}{\cos\theta_x} & 0 \\ 0 & 1 \end{bmatrix} \quad P_i = \begin{bmatrix} \cos\theta_i & 0 \\ 0 & 1 \end{bmatrix}$$
$$S = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad C_c = \begin{bmatrix} 1 & c \\ 0 & 1 \end{bmatrix} \quad \mathcal{B}_{t,\rho}\ \nu \equiv \begin{bmatrix} \nu_s + t \\ \min(\nu_\rho, \nu_a) \end{bmatrix}$$

Figure 4: *Matrix operators on 2D bandwidth.*

We neglect the effect of the product by the incident cosine, which only adds a small constant to the angular frequency. The mirror reparametrization flips the spectrum and its bounding box about the spatial axis, which multiplies ν by matrix S. The two curvature shears act on the bandwidth vector ν as products by shear matrices C_c and C_{-c} .

The reflectance function bandlimits the angular frequencies by its own angular bandwidth ν_ρ while the convolution with local texture adds the texture bandwidth ν_t to the spatial bandwidth. We denote this combined operator on $\mathcal{B}_{t,\rho}$.

The local angular and spatial bandwidths of the reflectance distribution and texture are precomputed quickly (Section 2.4), and allow support for a wide variety of materials including analytical BRDFs, acquired BRDFs and artistic shaders. The overall transformation undergone by incident bandwidths during reflection can thus be represented by a reflection operator $\mathcal{R}$ over the bandwidth vector:

$$\mathcal{R} = P_x\ C_{-c}\ S\ \mathcal{B}_{t,m}\ C_c\ P_i$$

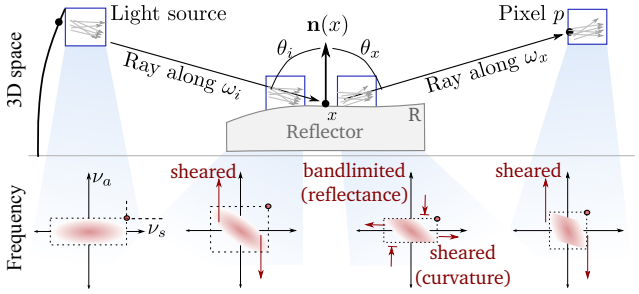


Figure 5: *Flatland illustration of local bandwidth propagation from lights to pixels. Rather than propagate the entire spectra, we only compute local bandwidth (dotted rectangles). Using the local bandwidth along each incident direction at R , we estimate the local bandwidth at p and hence determine image-space sampling rates (see Eq. 5). The spectra are illustrational. For distant illumination, incident spectra are purely angular.*

The bandwidth around a light path arriving at pixel p after one-bounce of a single ray from the light is

$$\nu_p^i = T_{d'}\ \mathcal{R}\ T_d\ \nu_l^i. \tag{3}$$

Here d is the distance from the light source to the bouncing point on the surface, d' is the distance from the surface to the image plane and ν_l^i is the bandwidth while originating at the light source along direction ω_i.

2.2 Image-space bandwidth and sampling rate

The bandwidth at pixel p, depends on the choice of ω_i sampled at x. That is, the 2D bandwidth ν_p at pixel p is a combination of the individual bandwidths ν_p^i along the sampled directions.

We calculate the 2D bandwidth at each pixel, ν_p as a weighted average of the sampled incident illumination $L_i(\omega_i)$ at x , reflectance and the 2D bandwidths of the associated one-bounce paths ν_p^i:

$$\nu_p = \frac{1}{\sum_{j=1}^{n_b} L_i(\omega_j)} \sum_{i=1}^{n_b} \nu_p^i L_i(\omega_i)\ \rho(\omega_i, \omega_{x\to p})\ \omega_i.\mathbf{n_x}. \tag{4}$$

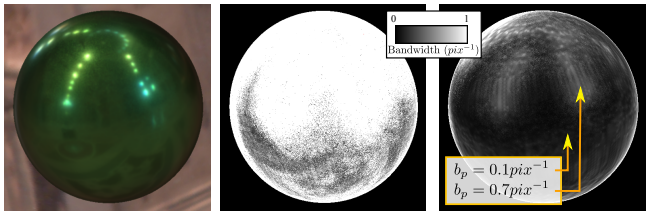
Although the bandwidth at each pixel is estimated using multiple samples, we show that a small choice of n_b yields results of sufficient quality and increasing n_b produces only minor improvements in quality (see fig 12).

To determine the local sampling rate, we convert the angular bandwidth estimates at the image plane into image space bandwidth b_p (in pixel^{-1}) where

$$b_p \quad = \quad \nu_{pa}\ \max\left[\frac{f_x}{W}, \frac{f_y}{H}\right], \tag{5}$$

where f_x and f_y are the horizontal and vertical fields of view, and the rendered image is $W \times H$ pixels. A minimum of $2b_p$ samples per pixel are required to represent the local variations, to satisfy the Nyquist criterion.

One simple alternative to equation 4 is to set $\nu_p = \max \nu_p^i,\ 1 \le i \le n_b$ where n_b is the number of samples used for bandwidth estimation. However, in practice, this turns out to be highly conservative (see fig 6). For illustration, consider a specular sphere. The visible frequencies on the sphere correspond to the bandwidth along the direction of mirror reflection and is therefore



Rendered image (Our technique) | Bandwidth estimate based on max | Our estimate (Equation 4)

Figure 6: *Estimating the reflected bandwidth using several one-bounce paths by sampling incident illumination.* Middle: *The maximum over the incident directions results in a very conservative result (1 pixel^{-1} almost everywhere on the sphere).* Right: *Our estimate (eq. 4) predicts high variation when the incident illumination is high-frequency along the reflected direction.*

view-dependent. However, if the maximum of all incident bandwidths is chosen to be the reflected bandwidth, then all points on the sphere will possess similar values of reflected bandwidth. For practically useful bandwidth estimates, the reflected bandwidth needs to take into account the material, relative orientation of illumination and view, and the local geometry.

2.3 Adaptive sampling for shading

The variance of the shading integrand $\sigma_p^2(\omega_i)$ about a single illumination direction ω_i, at a point x that projects to pixel p, is given as

$$\begin{aligned} \sigma_p^2(\omega_i) \quad &= \quad \mathrm{E}(\chi_i^2 \mu_i^2) - \mathrm{E}(\chi_i \mu_i)^2 \\ &\le \quad \mathrm{E}(\chi_i^2) = \mathrm{E}(\hat{L}_i^2 \otimes \hat{\rho}^2) \end{aligned} \tag{6}$$

where $\chi_i = L_i(\omega_i)\rho(\omega_i, \omega_{x\to p})$, $\mu_i = \omega_i.\mathbf{n_x}$, $\mathrm{E}(X)$ denotes the expected value of X, $\hat{f}$ denotes the Fourier transform of f and $\otimes$ denotes convolution. The second equality is a consequence of Parseval's theorem. The convolution is in the angular domain. Further,

$$\mathrm{E}(\hat{L}_i^2 \otimes \hat{\rho}^2) \le (\nu_{xa}^i + \nu_{\rho a}^i)\ \chi_i^2 \tag{7}$$

where ν_{xa}^i is the angular bandwidth at x along incident direction ω_i and $\nu_{\rho a}^i$ is the angular bandwidth of the reflectance function at $(\omega_i, \omega_{x\to p})$.

We adapt the number of shading samples at each pixel N_p to be proportional to the sum of the bandwidths, weighted by the illumination and reflectance values, along sampled incident directions ω_i:

$$N_p \propto \sum_{i=1}^{n_b} (\nu_{xa}^i + \nu_{\rho a}^i)\ \chi_i^2. \tag{8}$$

The sum is a conservative approximation of the variance of the integrand. In practice, $n_b = 16$ already provides acceptable quality (see fig 12).

The summations over multiple incident directions (equations 4 and 8) indicate that we implicitly account for the relative alignment (phase) of the illumination and reflectance. Previous approaches that neglect phase cannot predict variation due to view-dependent effects.

2.4 Illumination and BRDF bandwidth computation

All the bandwidth components that we use are computed on the fly, except the angular bandwidth of the BRDF, $\nu_{\rho a}$, which we precompute and store. In this paper, we only demonstrate separable reflectance composed of a spatially homogeneous BRDF along with a

texture. However, all the derivations for bandwidth can seamlessly handle spatially varying BRDFs.

Distant illumination: The local 2D bandwidth of distant illumination along ω: $\nu(\omega) = [0, \nu_{a\omega}]^T$. The local angular bandwidth, $\nu_{a\omega}$, can be computed either using a windowed Fourier transform centered at $L(\omega)$ or using wavelets. We use wavelets and compute bandwidth by measuring the first level in the wavelet pyramid at which coefficients involved in the computation of $L(\omega)$ get larger than a chosen threshold (See appendix for details). In practice, L is mapped onto an image that we process using the discrete wavelet transform (See Figure 7 for a typical example). The wavelet hierarchy level h is converted into angular bandwidth using $\nu_{a\omega} = \frac{2\pi}{2^h \lambda_{max}}$, where λ_{max} is the maximum eigenvalue of the Jacobian for the mapping of spherical coordinates onto the image plane. This approach allows to compute instant angular bandwidth in real time on GPU for environment maps, and is not prone to windowing artifacts. In all our experiments, we used 2D Daubechies wavelets of order 4.

Figure 7: Right: *Local angular bandwidth estimate for the environment map at left, computed from a 2D wavelet decomposition.*

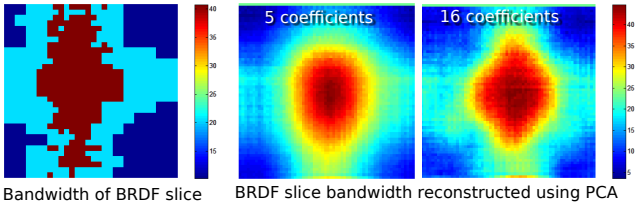


Figure 8: *A 64×64 ($n_b = 4096$) slice of the BRDF bandwidth and its reconstructions. Using PCA, the storage requirements of the BRDF bandwidth may be significantly reduced.*

Texture (spatial): The spatial component of the reflectance function is treated similarly. The spatial bandwidth is extracted using the same approach, this time accounting for the Jacobian of the mapping onto the surface so that the bandwidth is correctly expressed in inverse meters.

Reflectance (angular): For each incident direction in the local tangent frame, we compute the angular bandwidth map of the outgoing BRDF lobe. For this, we again use the same technique as for distant illumination and apply 2D wavelet transforms on the slices. We store the result for each lobe of the BRDF in a large texture. For the general case of 4D reflectance data we use 16×16 input directions and a 16×16 image for each reflectance lobe, packed into a 1024^2 texture. Since the maximum expressible bandwidth depends on resolution, we compute the bandwidth for higher-resolution angular slices and reduce it to 16×16. Alternatively, since the BRDF slices are smooth, their bandwidths can be effectively compressed using principal component analysis (PCA). Only the mean and a few coefficients are sufficient to reasonably reconstruct the BRDF bandwidth along arbitrary slices (see fig. 8).

3 Hierarchical shading algorithm

Our rendering algorithm consists of three steps: (1) a geometry pass that renders G-buffers; (2) a bandwidth buffer is filled with image-space bandwidth and the number of integration samples to use per pixel; (3) a one-pass multiresolution shading step, interleaved with upsampling. Fig. 9 presents an overview of the algorithm.

The rendering of G-Buffers is a classical geometry pass where normals, depths and material ID are computed and stored per fragment into a set of screen-space buffers. G-Buffers do not need to be hierarchically built (mip-mapped) in our method, which avoids typical numerical problems faced during averaging of normals and depth. A multi-resolution pyramid is built only for the bandwidth buffer.

3.1 Bandwidth buffer initialization

The bandwidth buffer contains two different values: the local image-space bandwidth and the number of samples to be used for shading each pixel. These are computed using the G-Buffers (equations 5 and 8). Although these estimations involve numerical integration, they are several orders of magnitude faster than the actual shading, since a coarse sampling is sufficient (see fig 12). In practice, rather than storing b_p (see Eq. 5) in the bandwidth buffer, we store

$$\min(\ \lfloor \log_2 \frac{1}{b_p} \rfloor,\ \ \min(\log_2(W), \log_2(H))\ ) \qquad (9)$$

which is the pyramid resolution at which pixel p needs to be shaded, accounting for the local variation at p. The floor operation ensures that the Nyquist sampling rate is respected. Note that storing b_p directly in the bandwidth buffer leads to identical results, and this optimization simplifies tests for deciding the pyramid resolution while shading each pixel.

In a second step, the bandwidth buffer itself is mip-mapped using a min filter, so that at a given level in the hierarchy, the value for a pixel conservatively decides whether subpixels pixel can be computed at this level. The same is done for the variance estimate using a max filter.

3.2 Shading and up-sampling

Our algorithm renders the image hierarchically, progressively from coarse to fine. At a given resolution (say $2^k \times 2^k$), we examine the bandwidth buffer and shade those pixels for which the bandwidth buffer pyramid contains the current coarseness resolution k. For pixels whose bandwidth buffer entries are less than the current resolution (i.e.$< k$), we bilaterally upsample from neighbors at the preceding level of coarseness ($2^{k-1} \times 2^{k-1}$), only accounting for pixels that are already computed. The parents' values are averaged with coefficients

$$w_i = g_z(z - z_i) g_a(p - p_i) \alpha_i$$

where z (resp. z_i) are the depths of the shaded (resp. parent) pixels, and g_z is a Gaussian that cancels out pixels of irrelevant depth, and α_i are bilinear weights (See Figure 10). The last term g_a is an anisotropic 2D Gaussian defined as

$$g_a(\mathbf{v}) = e^{-\mathbf{v}^T M \mathbf{v}} \quad \text{with} \quad M = R_\phi^T \begin{bmatrix} 1 & 0 \\ 0 & \cos\theta \end{bmatrix} R_\phi$$

where ϕ is the angle of the screen-projected normal at the surface, and θ_x the angle between the normal and the view direction. This enables efficient anisotropic filtering aligned with the highest and lowest screen-space frequencies, since $\frac{b_p}{\cos\theta}$ and b_p estimate the minimum and maximum directional screen-space bandwidth around current pixel.

We continue this process over successive levels, until we reach the finest resolution where we shade all remaining pixels. Pseudo-code for the algorithm is presented in Figure 10.

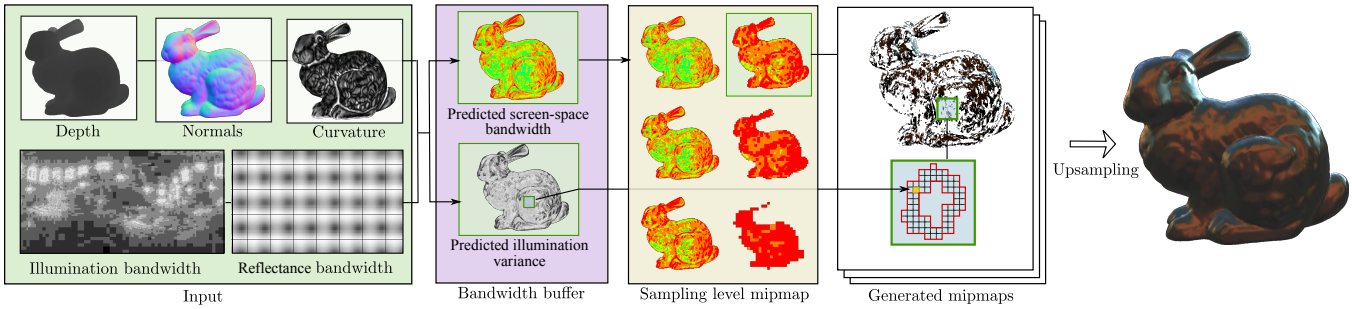


Figure 9: *Our rendering pipeline: At each frame, we first render G-Buffers. From these, we compute the* bandwidth buffer *that consists of image space bandwidth and shader integrand variance maps. The image-space bandwidth is then stored in a multiresolution pyramid. The final image is sampled starting from coarse levels. Depending on our bandwidth and variance predictions, pixels are either explicitly computed by numerical integration using an adapted number of samples or upsampled from parent pixels.*

for all points p at level L **do**
  if $b_w(p) < L$ **then**
    compute $w_0, ..., w_3$
    $c(p) \leftarrow \sum_k w_k c(p_k)$
  else if $b_w(p) == L$ **then**
    shade(p)

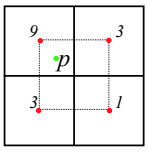
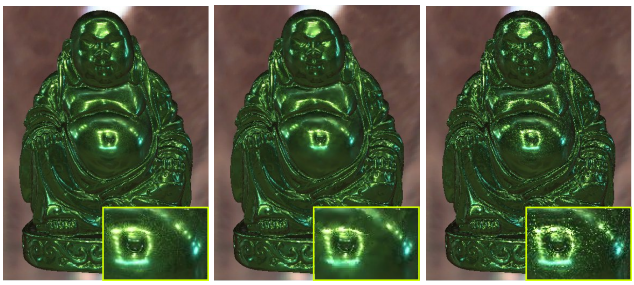


Figure 10: *Upsampling interpolation scheme.* Left: *Pseudocode for the computation of one level.* Right: *relative weights α_i for parent pixels of pixel p at the next level.*



(a) Equal-quality (10474 ms) (b) Our algorithm (1049 ms) (c) Equal-time (890 ms)

Figure 11: *Comparison between our algorithm (b) and reference for either equal quality (a) or equal-time (c). Our algorithm is 10 times faster than the reference for equal quality, and greatly lowers the variance for equal-time rendering. Forward-shaded references were computed using BRDF-importance sampling and a constant number of shading samples per pixel. Material:* `green-metallic-paint`

3.3 Shading computation

For each pixel that is shaded, the number of samples N_p to be used is read from the bandwidth buffer. We estimate reflected radiance (Equation 2) by importance sampling the BRDF lobe for the current view direction. In our implementation, we read N_p samples randomly, from multiple precomputed vectors of importance samples that is stored in a texture.

Shading computations are always performed with depth and normal values at the finest level, since G-Buffers are not mip-mapped. This is possible because the bandwidth buffer predicts whether, for any sub-pixel of the current level, the computation will yield similar estimates despite potential variation in the depths, normals and illumination.

4 Results and discussion

All the timings reported in this paper were obtained using an NVIDIA GeForce GTX 560 Ti graphics card with 1GB memory. All the pictures and videos were generated using the acquired materials from the MERL BRDF database [Matusik et al. 2003].

Performance comparisons: Figure 11 shows a comparison of our algorithm with reference pictures computed using brute-force BRDF importance sampling with the same quality (left) or with a similar computation time (right). For this scene, our algorithm is roughly $10\times$ faster for the same rendering quality and results in much less variance than the reference algorithm for a similar computation time.

Figure 12 shows a Stanford bunny with an acquired material, `color-changing-paint3`. This material is highly glossy and is therefore a challenging case for integration in a multi-light settings. Using our algorithm, we compute illumination for only 56 % of the pixels. Overall, our algorithm only computes 25 % of the shading samples, compared to a reference with similar quality, resulting in a 4 times speedup (see Figure 13).

Figure 13 presents the computation times for three of our test scenes, Fertility (Figure 1), Bunny (Figure 12) and Buddha (Figure 11), both for our algorithm and for a reference solution computed using forward-shading, BRDF-importance sampling and a constant number of shading samples per pixel. Depending on the material, geometry of the scene and percentage of screen-space covered by the object, our algorithm performs four to ten times faster than the reference. More importantly, bandwidth computation represents only a small fraction of the total computation time (less than 10 ms), meaning that the overhead is quite small.

Influence of parameters: the main parameter for our algorithm is the number of samples we use for the bandwidth estimation, n_b (see Equation 4). Figure 12 shows the influence of this parameter. We render the Stanford Bunny with `color-changing-paint3`, for different values of n_b. Qualitatively, the results are almost impossible to distinguish from each other, even in the zoomed-in insets. Quantitatively, the Peak Signal-Noise Ratio stays almost constant for all values of n_b. The rendering time has a small dependency on n_b. In short, the influence of n_b seems to be small, and our algorithm performs well with a small value of n_b, such as 16. This makes sense as n_b is only used to estimate the bandwidth, but not for the actual illumination computations.

Validation: Figure 14 shows a comparison between the image-space bandwidth and variance predicted by our algorithm and the actual image-space bandwidth and variance, computed on a refer-

(a) Our variance estimate | (b) Measured variance | (c) Our bandwidth estimate (64 samples) | (d) Measured FFT bandwidth | (e) Computed result | (f) False color difference

Figure 14: *Comparison between the variance and bandwidth estimated by our algorithm and reference values, for the Stanford Buddha and* `color-changing-paint3`*. (a) our estimate for the variance of the shading integrand (Equation 8), (b) reference variance computed using extensive sampling, (c) our estimate of the screen-space bandwidth, (d) reference screen-space bandwidth computed using windowed FFT, (e) the final result of our method, and (f) false color difference between our result and the reference picture in proportion to average image intensity.*

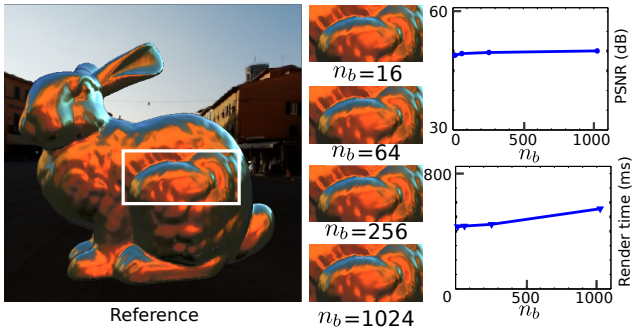


Figure 12: *Stanford Bunny rendered using* `color-changing-paint3`*, with different values of* n_b $(16, 64, 256, 1024)$*, at a screen resolution of* 512×512*, along with the PSNR (top left) and the rendering time (bottom left). The zoomed-in insest are virtually indistinguishable for all values of* n_b*. The PSNR plots show that our algorithm gives correct results for low values of* n_b*.*

Model	Fertility	Bunny	Buddha
Reference	1064	1708	10474
Ours: Total	247	425	1049
Bandwidth calculation	8	9	10
Shading integration	193	416	944

Figure 13: *Total computation times for three of the models shown in the paper: Fertility (fig. 1), Bunny (fig. 12 and Buddha (fig. 14), for both our algorithm and a reference picture with same quality. Depending on the material and percentage of screen covered, our algorithm provides a 4 to 10 times speedup. Computing the bandwidth takes less than 10 ms for all scenes.*

ence image.

The reference bandwidth was computed using a windowed Fourier transform over the image, with a window size of 32×32 pixels, resulting in a blurry picture. A large window for the FFT results in uncertainties in space (blur), while a small window results in uncertainties in frequency (bandwidth). Comparing Figures 14(c) and 14(d), it appears that our predicted bandwidth has the same order of magnitude than the reference, and — more importantly — a similar spatial distribution.

The reference variance was computed using extensive sampling. The variance estimated by our algorithm has a similar spatial distribution, but is conservative (as predicted in Section 2.3). Since we distribute a fixed budget of shading samples for rendering, the spatial distribution of variance and bandwidth predicted by our algorithm are more important than their actual values.

Anti-aliasing: Deferred shading approaches are not adapted to multi-sample anti-aliasing (MSAA). Since deferred shading result in a shader evaluation for every sample, the computation time increases linearly with the number of samples per pixel: with 4 samples per pixe, the framerate is divided by 4, and with 16 samples per pixel, it is divided by 16. As a consequence, anti-aliasing is impractical with deferred shading except for very simple shaders.

A consequence of our bandwidth-aware rendering algorithm is that we reduce the cost of anti-aliasing. We render the G-buffers at the higher resolution (4 times or 16 times the original number of pixels), but we only compute shading at the required level in the pyramid, depending on the predicted bandwidth. As a consequence, many samples are computed at a higher level. In practice, we only compute shading calculations at the finest levels introduced for anti-aliasing for a small fraction of the picture (see Figure 15). The overall cost of anti-aliasing is therefore low: computing 4 samples per pixels only doubles the rendering time (404 ms instead of 190 ms), and 16 samples per pixel only multiply it by 5 (1020 ms).

For comparison, our algorithm renders the picture with 16 samples per pixel in 1020 ms, while rendering a reference picture at 16 samples per pixels requires 32,000 ms.

Dynamic geometry, normal and displacement mapping: The bandwidth and variance computed are based on curvatures, normals and geometry from the G-buffers, inside deferred shading. Our algorithm handles dynamic geometry, displacement mapping and normal maps seamlessly. Figure 16 shows an example of our bandwidth estimation algorithm on a dynamic geometry, running at 15 fps (see also the accompanying video).

4.1 Discussion and limitations

Visibility and indirect lighting: Our algorithm only handles direct illumination, without any visibility effects. This is a strong limitation of our method, and an important avenue for future work.

Spatially-varying BRDFs: In this paper, we have only used homogeneous (non spatially varying) materials, multiplied by a tex-

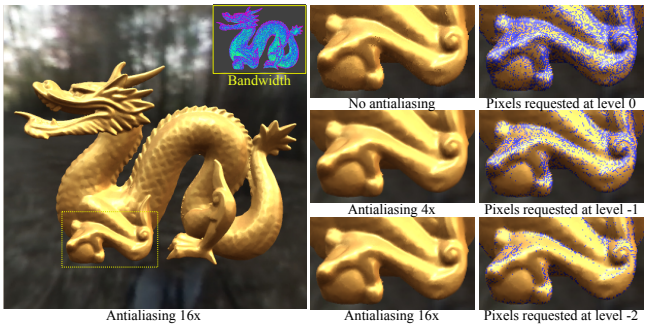


Figure 15: *The Stanford dragon, rendered with* `gold-paint`*. Without anti-aliasing (190 ms, top row), pixel artefacts appear. These artefacts are removed using anti-aliasing. Our algorithm performs anti-aliasing simply by adding extra levels to the pyramid: one level for* $4\times$ *anti-aliasing (level* -1*), two levels for* $16\times$ *anti-aliasing (levels* -1 *and* -2*). Extra shading computations are done only at pixels where the predicted bandwidth demands it (the blue pixels in the rightmost column), keeping the extra cost of anti-aliasing under control (404 ms for* $4\times$*, 1020 ms for* $16\times$*).*

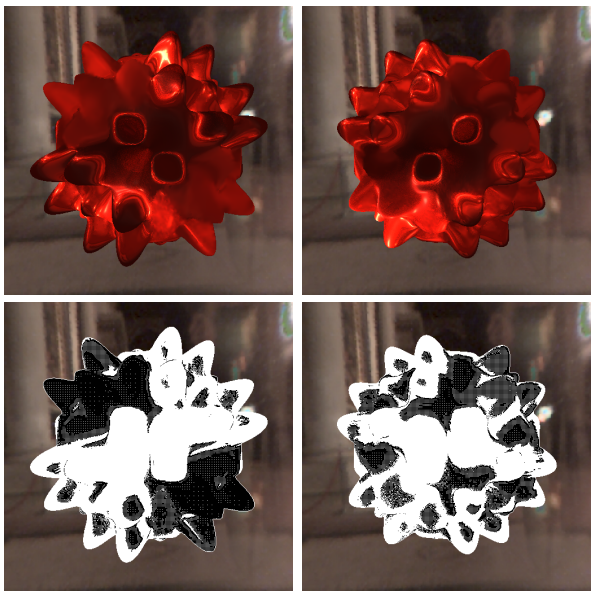


Figure 16: *Application of our algorithm to dynamic geometry, with* `red-metallic-paint`*. Since our algorithm depends only on data from the G-buffers, it can handle dynamic geometry, displacement mapping and normap maps.* top row: *rendered images,* bottom row: *pixels for which we performed shading computations. The screen-space bandwidth adapts to the curvature (see also the companion video. Rendering time: 66 ms.*

ture. This way, we could precompute local bandwidths independently for reflectance ($4D$) and texture ($2D$), in a few seconds. Extending our algorithm to fully spatially varying BRDFs ($6D$) is possible. It would involve a local bandwidth precomputation over the $6D$ data, followed by compression using PCA.

Local light sources: In this paper, we have placed ourselves in a multi-light settings because it is the most challenging case for efficiently estimating the shading integrals. It is possible to use our algorithm with local light sources instead of acquired environment maps. The only change would be the frequencies emitted by the light source. While distant illumination only contains angular frequencies, illumination from local sources would also contain spatial frequencies, that would depend on their geometries.

Anisotropic estimations : Our definition of 2D bandwidth does not account for anisotropy in space-angle. Therefore we cannot accurately predict the most relevant orientation of anisotropic reconstruction filters. This would require a 4D bandwidth calculation which would be more expensive to compute.

Conservative bandwidth estimate: We conservatively predict bandwidth, choosing suboptimal (excessive) sampling over artifacts from insufficient sampling. In particular, the $\min$ operator of Figure 4 is always larger than the real bandwidth of the product of the BRDF and illumination spectra.

5 Conclusion

In this paper, we have described a fast algorithm for shading acquired materials. Our algorithm is based on freauency bandwidth information, computed in real-time using information from the G-buffers. This bandwidth information is used for optimal sample placement in interactive shading.

Our acceleration of shading computation results from: (1) shading only a small fraction of pixels where the local bandwidth is expected to be high and (2) adaptive sampling of the shading integrals across pixels in the image. We have demonstrated the efficacy of using our screen space bandwidth information for efficient antialiasing in the context of deferred shading.

In future works, we would like to extend this work to indirect lighting and visibility effects.

A Instantaneous frequency using wavelets

Wavelets are local in both the spatial and frequency domains and reliable estimates of local bandwidths in a signal can be obtained by examining the set of wavelets that contribute to each point of the signal. If x be a point in the domain of a signal s, ψ and ϕ denote the scale function and the mother wavelet respectively,

$$s(x) = \sum_i \beta_i \, \phi_i(x) \quad + \quad \sum_i \sum_j \lambda_{i,j} \, \frac{1}{2^i} \, \psi\left(\frac{x - 2^i j}{2^i}\right).$$

Since wavelets of the same scale have identical bandwidths, we compute the maximum wavelet coeficient $\lambda_i^{max} = \max_j |\lambda_{ij}|$ per frequency band, and estimate the signal bandwith by:

$$b_w = 2^{I_w} \;\text{ with }\; I_w = \text{argmin}_i \sum_{k=i}^{n} \lambda_k^{max} < \epsilon \max_k \lambda_k^{max}$$

Because wavelets are localized in the frequency domain, the result barely depends on ϵ as long as it is a small value. We use $\epsilon = 0.01$ in all our experiments.

References

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.

CLAUSTRES, L., BARTHE, L., AND PAULIN, M. 2007. Wavelet Encoding of BRDFs for Real-Time Rendering. In *Graphics Interface (GI)*, 169–176.

DEERING, M., WINNER, S., SCHEDIWY, B., DUFFY, C., AND HUNT, N. 1988. The triangle processor and normal vector shader: a VLSI system for high performance graphics. *SIGGRAPH Comput. Graph. 22*, 4, 21–30.

DURAND, F., HOLZSCHUCH, N., SOLER, C., CHAN, E., AND SILLION, F. X. 2005. A frequency analysis of light transport. *ACM Trans. Graph. 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. Graph. 28*, 3, 93:1–93:13.

FATAHALIAN, K., BOULOS, S., HEGARTY, J., AKELEY, K., MARK, W. R., MORETON, H., AND HANRAHAN, P. 2010. Reducing shading on GPUs using quad-fragment merging. *ACM Trans. Graph. 29*, 4, 67:1–67:8.

HEIDRICH, W., AND SEIDEL, H.-P. 1999. Realistic, hardware-accelerated shading and lighting. In *SIGGRAPH '99*, 171–178.

KAUTZ, J., AND MCCOOL, M. D. 1999. Interactive rendering with arbitrary brdfs using separable approximations. In *SIGGRAPH '99 abstracts and applications*, 253.

KAUTZ, J., SNYDER, J., AND SLOAN, P.-P. J. 2002. Fast arbitrary BRDF shading for low-frequency lighting using spherical harmonics. In *Eurographics Symposium on Rendering*, 291–296.

KI, H., AND OH, K. 2008. A GPU-based light hierarchy for real-time approximate illumination. *Vis. Comput. 24*, 7, 649–658.

KOPF, J., COHEN, M. F., LISCHINSKI, D., AND UYTTENDAELE, M. 2007. Joint bilateral upsampling. *ACM Trans. Graph. 26*, 3.

LATTA, L., AND KOLB, A. 2002. Homomorphic factorization of BRDF-based lighting computation. *ACM Trans. Graph. 21*, 3.

MATUSIK, W., PFISTER, H., BRAND, M., AND MCMILLAN, L. 2003. A data-driven reflectance model. *ACM Transactions on Graphics 22*, 3 (July), 759–769.

NICHOLS, G., AND WYMAN, C. 2009. Multiresolution splatting for indirect illumination. In *Symposium on Interactive 3D graphics and games (I3D)*, 83–90.

NICHOLS, G., AND WYMAN, C. 2010. Interactive indirect illumination using adaptive multiresolution splatting. *IEEE Transactions on Visualization and Computer Graphics 16*, 5, 729–741.

NICHOLS, G., PENMATSA, R., AND WYMAN, C. 2010. Interactive, multiresolution image-space rendering for dynamic area lighting. *Computer Graphics Forum 29*, 4, 1279 – 1288.

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

RAMAMOORTHI, R. 2009. Precomputation-based rendering. *Found. Trends. Comput. Graph. Vis. 3* (April), 281–369.

RITSCHEL, T., ENGELHARDT, T., GROSCH, T., SEIDEL, H.-P., KAUTZ, J., AND DACHSBACHER, C. 2009. Micro-rendering for scalable, parallel final gathering. *ACM Trans. Graph. 28*, 5, 132:1–132:8.

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

SHOPF, J., NICHOLS, G., AND WYMAN, C. 2009. Hierarchical image-space radiosity for interactive global illumination. *Computer Graphics Forum 28*, 4.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21* (July), 527–536.

SOLER, C., SUBR, K., DURAND, F., HOLZSCHUCH, N., AND SILLION, F. X. 2009. Fourier depth of field. *ACM Trans. Graph. 28*, 2.

SOLER, C., HOEL, O., AND ROCHET, F. 2010. A Deferred Shading Algorithm for Real-Time Indirect Illumination. In *ACM SIGGRAPH Talks*, 18:1–18:1.

SUBR, K., AND ARVO, J. 2007. Steerable importance sampling. In *Proceedings of the 2007 IEEE Symposium on Interactive Ray Tracing*, IEEE Computer Society, Washington, DC, USA, 133–140.

SUN, X., ZHOU, K., CHEN, Y., LIN, S., SHI, J., AND GUO, B. 2007. Interactive relighting with dynamic brdfs. *ACM Trans. Graph. 26* (July).