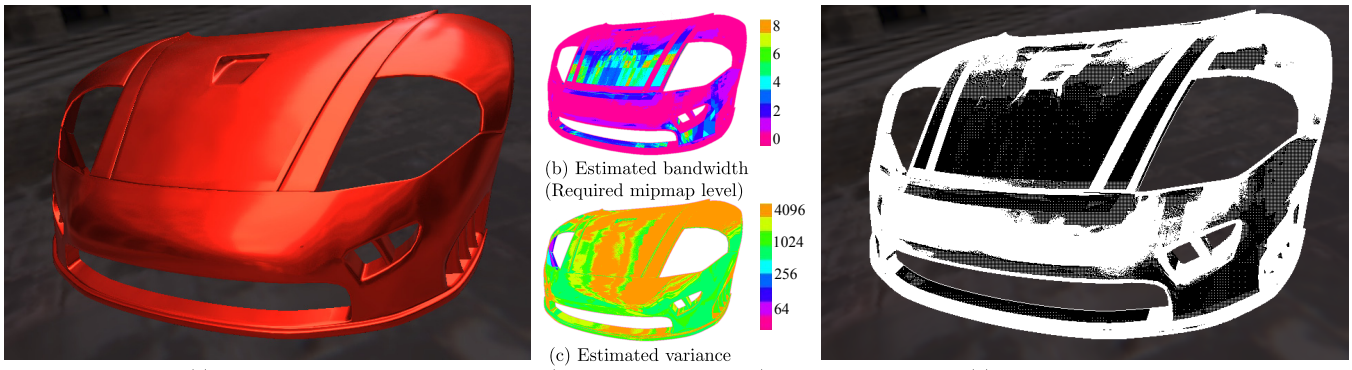
Interactive rendering of acquired materials on dynamic geometry using bandwidth prediction

Mahdi M. Bagher, Cyril Soler, Kartic Subr, Laurent Belcour, Nicolas Holzschuch

Interactive rendering of acquired materials on dynamic geometry using bandwidth prediction

Mahdi M. Bagher* Cyril Soler* Kartic Subr† Laurent Belcour* Nicolas Holzschuch*



(a) Rendered image (b) Estimated bandwidth (Required mipmap level) (c) Estimated variance (Required # light samples) (d) Pixels actually computed

Figure 1: *We accelerate the shading of acquired materials by selectively shading pixels (white pixels in (d)) using bandwidth prediction (b), and adapting the number of samples to the variance of the illuminant (c). We interactively render acquired materials while editing the geometry (material:* `red-metallic-paint` *from the MERL database).*

Abstract

Shading complex materials such as acquired reflectances in multi-light environments is computationally expensive. Estimating the shading integral requires multiple samples of the incident illumination. The number of samples required varies across the image, depending on a combination of several factors. Adaptively distributing computational budget across the pixels for shading is a challenging problem. In this paper we depict complex materials such as acquired reflectances, interactively, without any precomputation based on geometry. We first estimate the approximate spatial and angular variation in the local light field arriving at each pixel. This *local bandwidth* accounts for combinations of a variety of factors: the reflectance of the object projecting to the pixel, the nature of the illumination, the local geometry and the camera position relative to the geometry and lighting. We then exploit this bandwidth information to adaptively sample for reconstruction and integration. For example, fewer pixels per area are shaded for pixels projecting onto diffuse objects, and fewer samples are used for integrating illumination incident on specular objects.

1 Introduction

Real materials can exhibit subtle and rich shading effects, such as colors changing depending on the viewing direction. While acquired real-world reflectance data are publicly available, their realistic depiction under environmental lighting conditions is slow. Photorealistic shading involves costly numerical integration per pixel over multiple incident directions.

*Maverick, INRIA Grenoble Rhône-Alpes and Laboratoire Jean Kuntzmann, Université de Grenoble and CNRS.

†University College London.

For some applications, such as shape design (Figure 1), interactive feedback of material appearance is desirable. The need to allow dynamically changing geometry poses the first challenge to interactive shading. In addition, it is important that the shading is realistic, that is consistent with its final appearance after post-design offline rendering, which is often physically-based.

A gamut of approaches partially address this problem. At one end, fast algorithms focus on editable geometry with simple material models. Other algorithms strive to depict a variety of effects such as global illumination. The latter approach typically requires the precomputation of radiance transfer (including visibility) and prevents geometry editing [Sloan et al. 2002]. The combination of editable geometry and realisticly portraying complex materials such as acquired bidirectional reflectance distributions (BRDFs) is still an open research problem and is the focus of this paper. We achieve this combination at the cost of global illumination and visibility.

Simulating material-appearance under environmental illumination requires the estimation of an integral of the incident illumination at each pixel modulated by the material's reflectance function. The integrands are typically sampled, and the sampling rate depends on the material: diffuse materials require many samples over incident directions, but exhibit low variation between neighboring pixels; specular materials require fewer samples over incident directions but cause large variation across nearby pixels.

We leverage theories in frequency domain light transport [Durand et al. 2005] to systematically exploit the relation between sampling in image-space (reconstruction) and sampling for shading (integration). For reconstruction, we propose a new multiresolution algorithm. For integration, we predict the required *number of samples*. Our prediction may be used in conjunction with any sampling strategy for numerical integration.

In this paper, we introduce the concept of computing and storing the maximum local frequencies of the radiance field. We propose a practical representation of local variation—*local bandwidth*—along with a fast algorithm to compute it. We use this information to adapt sampling rates for reconstruction and integration during rendering. Our rendering algorithm consists of two steps. First, for each pixel, we estimate the spatial and angular bandwidth. This in-

formation is stored, hierarchically, in a buffer having the same size as the picture generated. Next, we use this information to sample the image. We shade fewer pixels in smoothly varying areas, and adapt the number of samples according to the predicted variance. We render the final image using the scattered shaded pixels and upsampling.

Our contributions are the following:

1. *Rapid bandwidth computation*: we quickly (about 8ms) predict local variation in the image due to reflected illumination.
2. *Multiresolution shading*: our multiresolution deferred-shading algorithm uses the local frequency information for efficient sampling. We adaptively sample for reconstruction (shading only some pixels) and for integration (number of light samples for each shaded pixel)
3. *Adaptive multisampling anti-aliasing*: we only compute sub-pixel shading for those pixels where the predicted image-space frequency is greater than 1 pixel^{-1}.
4. *Reflectance bandwidth-estimation*: we estimate local bandwidth of arbitrary reflectance functions.

1.1 Related Work

Deferred shading: unlike common rendering methods on the GPU, *Deferred shading* postpones shading until occlusions are resolved in image space (see, e.g. [Deering et al. 1988]). It is more efficient for computationally expensive shaders, but incompatible with multi-sampling anti-aliasing methods [Fatahalian et al. 2010]. Our algorithm allows adaptive sampling with complex materials and incoming light, and sub-linear multi-sample anti-aliasing.

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 also 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]. [Claustres et al. 2007] provides real time rendering of acquired data using the sparsity of a wavelet representation and low rank of the reflection operator. Wang *et al.* [2009] proposed a real-time rendering technique based on a spherical Gaussian approximation of the BRDF. Our main contribution is accurate shading of acquired materials while only shading a subset of the pixels.

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 simulation of motion blur [Egan et al. 2009] and depth of field [Soler et al. 2009]. [Ramamoorthi et al. 2007] extended the Fourier analysis to gradients. These works provide key insights and understanding of the variation in light transport. They do not provide practical mechanisms to propagate the frequency content to image space in real time. Propagating sampled spectra [Soler et al. 2009] is costly. Egan et al. [2009] derive formulae for transformations to the 3D lightfield (space and time) assuming only diffuse objects. Our approach builds upon these works, but we propose a rapid method to only estimate maximum variation along space and angle, rather than the entire spectra.

Multiresolution screen-space algorithms: techniques render by heuristically shading pixels at varying levels of coarseness, then 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]. 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 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.

Precomputed transport: Precomputed approaches compress or represent 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. They require that the geometry remains static. In comparison to these algorithms, we are restricted to first bounce radiance without global illumination or visibility for shadows, but allow interactive editing of the geometry.

None of the approaches described above achieves realistic material depiction on dynamically editable geometry.

1.2 Overview

The radiance arriving at each pixel p after one-bounce direct reflection at a point x (ignoring visibility) is

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

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, Ω_x is the set of incident directions on the hemisphere above the local tangent plane, and ρ is the reflectance function. This integral is typically estimated using Monte Carlo estimators as an average of N_p illumination samples:

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

where the $\omega_i \in \mathcal{S}^2$ are random incidence directions distributed according to the importance function $g(\omega_i)$ and G is the importance function integrated over Ω_x.

We accelerate rendering by, first, *avoiding shading all pixels*, computing the integral only at pixels with large local variation and upsampling from neighboring pixels for the others (Section 3.2), and second, for each pixel p where we estimate the integral, *adaptively choosing* N_p according to the predicted variance of the shading integrand (Section 2.3). In Section 3, we present a multiresolution shading algorithm that implements this strategy.

2 Real-time bandwidth estimation

We only propagate maximum local variation (bandwidth) about light paths. See Figure 2 for an example of bandwidth propagation in Flatland.

2D bandwidth We analyze the local lightfield using the parametrization of Durand et al. [2005]. Their parametrization is in 4D and we define the bandwidth of the local lightfield as a 2D vector with the maximum non-zero Fourier frequencies in space and angle. For robustness we use a quantile (the 95^{th} percentile) of the power spectra rather than the absolute maximum. 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

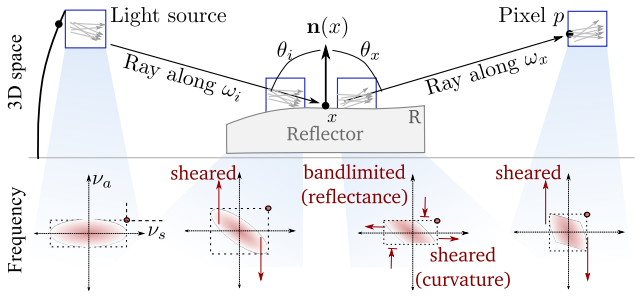


Figure 2: *Flatland illustration of local bandwidth propagation. Our idea is to only propagate local bandwidth information (dotted rectangles). Then, using local bandwidth along a few (typically 16) incident directions at R, we estimate the local image variation at p and use it to determine image-space sampling rates (see Eq. 6).*

chosen. We denote the bandwidth using $\nu \equiv [\nu_s \ \nu_a]^T$ 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 (Figure 3).

From [Durand et al. 2005], we derive simple linear transformations undergone by ν for each step of the transport process (see Figure 4). We describe how to derive sampling rates using the bandwidth information. Finally, we explain how to estimate the variance of the shading integrand for adaptive sampling.

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 2), where T_d is defined in Figure 5.

Reflection: in the frequency analysis framework, reflection is realized in four steps [Durand et al. 2005]:

1. Re-parametrization of the incident lightfield into the frame of the reflecting surface. This is a spatial scale by $\cos\theta_i$ 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, which 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. Re-parametrization along the outgoing direction. This is a mirror reflection in the spatial domain, followed by a spatial curvature shear of length $-c$ and a spatial scale of $1/\cos\theta_x$.

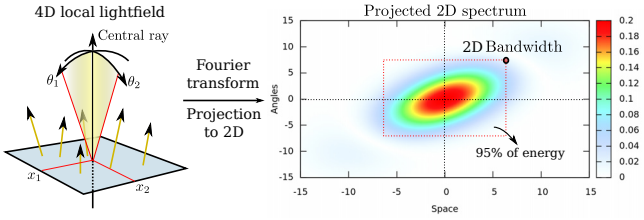


Figure 3: Left: *4D local lightfield parametrization adopted by Durand et al. [2005].* Right: *2D parametrization introduced by Soler et al. [2009]. We define local bandwidth $\nu \equiv [\nu_s \ \nu_a]^T$ (black dot) so that 95% of the spectral energy lies in the dotted rectangle.*

	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 4: *Review of spectral operations from [Durand et al. 2005].*

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

Figure 5: *Matrix operators on 2D bandwidth.*

We translate these operations into matrix operations onto the bandwidth vector ν of the incident local light field (see Figure 5). The reparametrization (first and last steps) are simply scaling (P_i and P_x) and shearing (C_c and C_{-c}) matrices applied to ν. Mirror reparametrization is multiplication by matrix S. The reflectance function bandlimits angular frequencies based on its own angular bandwidth ρ while the convolution with local texture augments the spatial bandwidth by the bandwidth t of the texture. We denote this using the operator $\mathcal{B}_{t,\rho}$. We neglect the product by the incident cosine, which only adds a small constant to the angular frequency.

We quickly precompute angular and spatial bandwidths of the reflectance distribution (and texture) (Section 2.4). This computation is applicable to any type of reflectance function (analytical BRDFs, acquired BRDFs or 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:

$$\mathcal{R} = P_x\ C_{-c}\ S\ \mathcal{B}_{t,\rho}\ C_c\ P_i \tag{3}$$

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

$$\nu^i = T_{d'}\ \mathcal{R}\ T_d\ \nu_l^i \tag{4}$$

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.

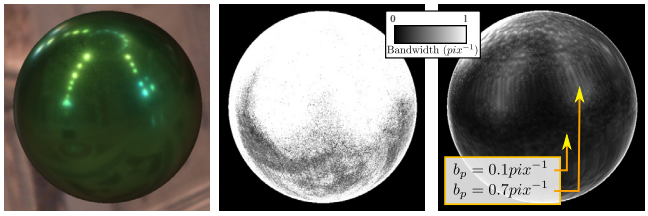
2.2 Image-space bandwidth and sampling rate

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

We compute the 2D bandwidth at each pixel ν 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^i:

$$\nu = \begin{bmatrix} \nu_s \\ \nu_a \end{bmatrix} = \frac{1}{\sum_{j=1}^{n_b} L_i(\omega_j)} \sum_{i=1}^{n_b} \nu^i L_i(\omega_i)\ \rho(\omega_i, \omega_{x\to p})\ \omega_i.\mathbf{n_x}. \tag{5}$$

Although the bandwidth at each pixel is estimated using multiple samples, a small choice of n_b is sufficient (see Figure 13). Using



Rendered image (Our technique) | Bandwidth estimate based on max | Our estimate (Equation 3)

Figure 6: *Combining bandwidth estimates from sampled incident directions.* Middle: *Applying a* max *overestimates sampling rates (1 pixel^{-1} almost everywhere on the sphere).* Right: *Our approach (eq. 5) predicts view-dependent sampling rates.* Left: *Final result.*

a max operation in place of the sum in equation 5 does not capture view-dependent effects. For example, bandwidth after reflection from a specular sphere would be equally high regardless of viewing or light direction (see Figure 6). We account for the material, relative orientation of illumination and view, and local geometry.

The required sampling rates at the image plane are twice the local image-space bandwidth (Nyquist criterion) b_p (in pixel^{-1}):

$$b_p \quad = \quad \nu_a \ \max\left[\frac{f_x}{W}, \frac{f_y}{H}\right], \tag{6}$$

where f_x and f_y are the horizontal and vertical fields of view, and the rendered image is $W \times H$ pixels.

2.3 Adaptive sampling for shading

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

$$\sigma_p^2(\omega_i) \quad = \quad \mathrm{E}(\chi_i^2\mu_i^2) - \mathrm{E}(\chi_i\mu_i)^2 \leq \mathrm{E}(\chi_i^2) = \mathrm{E}(\hat{L}_i^2 \otimes \hat{\rho}^2)$$

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 (see, *e.g.* [Oppenheim and Schafer 1999]). The convolution is in the angular domain. Further, $\mathrm{E}(\hat{L}_i^2 \otimes \hat{\rho}^2) \leq (\nu_a^i + \nu_{\rho a}^i)\ \chi_i^2$, where ν_a^i is the angular component of ν^i, and $\nu_{\rho a}^i$ is the local angular bandwidth of the reflectance function.

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

$$N_p \propto \sum_{i=1}^{n_b} (\nu_a^i + \nu_{\rho a}^i)\ \chi_i^2. \tag{7}$$

The sum is a conservative approximation of the variance of the integrand. $n_b = 16$ provides acceptable quality (see Figure 13).

The summations over incident directions (Equations 5 and 7) 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

We perform bandwidth computations on the fly, except for the angular bandwidth of the reflectance functions, $\nu_{\rho a}$, which we precompute and store. In this paper, we only demonstrate separable reflectance: q spatially-homogeneous angular reflectance distribution along with a texture. However, all the derivations for bandwidth hold for spatially-varying BRDFs.

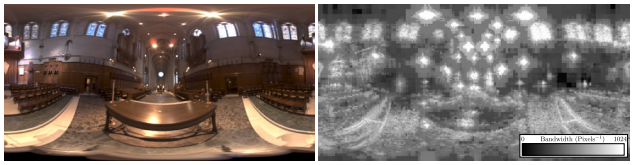


Figure 7: *Input environment map and its local angular bandwidth computed using a 2D wavelet decomposition.*

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 chose to 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 (Figure 7). 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.

Texture (spatial): 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. We 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. We tried compression of the BRDF bandwidth with principal component analysis but did not observe any improvement.

2.5 Implementation roadmap

Practical implementation is simple (although the theory is not): for each pixel, compute the image bandwidth using Equation 5 and 6, and the number of samples for the shading integrand using Equation 7. Both are summations over $n_b = 16$ incoming directions ω_i. For each ω_i, the local 2D bandwidth ν^i is given by Equations 3 and 4, using the matrices listed in Figure 5. These matrices require the curvature c, normal n, the incident and outgoing angles (θ_i and θ_o), the precomputed spatial and angular bandwidth of the material (resp. t and ρ) for the current pixel and direction ω_i, and the precomputed bandwidth of the light source ν_l^i in direction ω_i.

3 Hierarchical shading algorithm

Our rendering algorithm consists of three steps (see Figure 8): (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.

Rendering G-Buffers is a classical geometry pass where we store normals, depths and material ID into a set of screen-space buffers. G-Buffers do not need to be hierarchically built (mip-mapped) in

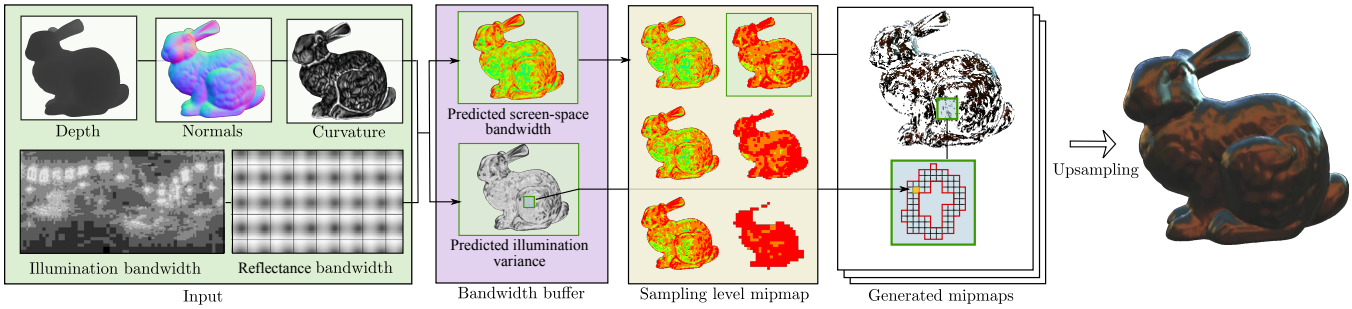


Figure 8: *Our rendering pipeline: at each frame, we first render G-Buffers. From these, we compute an additional* bandwidth buffer *that stores image space bandwidth and shader integrand variance maps. The former is stored in a multiresolution pyramid. During rendering of the final image from coarse to fine scale, depending on our bandwidth and variance predictions, pixels are either explicitly shaded (numerical integration) or upsampled from parent pixels.*

our method; we build a multi-resolution pyramid 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 6 and 7). Although these estimations involve numerical integration, they are several orders of magnitude faster than the actual shading, since a coarse sampling is sufficient (Figure 13). Rather than storing b_p (see Equation 6) in the bandwidth buffer, we store

$$\min(\ \lfloor \log_2 \frac{1}{b_p} \rfloor,\ \min(\ \log_2(W),\ \log_2(H)\ )\ ) \qquad (8)$$

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. Storing b_p directly in the bandwidth buffer leads to identical results; our optimization simplifies tests for deciding the pyramid resolution while shading each pixel.

The bandwidth buffer is mip-mapped using a min filter, so that at a given level in the hierarchy, the value for a pixel conservatively tells us whether sub-pixels should be computed at this level. We do the same for the variance estimate using a max filter.

3.2 Shading and up-sampling

We render 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 the 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 (Figure 9). 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.

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)

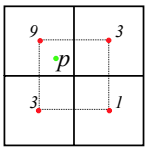


Figure 9: *Upsampling interpolation scheme.* Left: *Pseudocode for the computation of one level.* Right: *relative weights α_i for parent pixels of pixel p at the next level.*

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 9.

3.3 Shading computation

For each shaded pixel we read the number of samples N_p 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 are stored in a texture. Our algorithm is compatible with any importance sampling strategy. In practice, we importance sample the reflectance by numerical inversion of its cumulative distribution.

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

4 Results and discussion

All timings reported in this paper were measured on an NVIDIA GeForce GTX 560 Ti graphics card with 1GB memory.

4.1 Behavior of our algorithm

Computation time: the computation time for our algorithm scales linearly with the total number of shading samples (Figure 10, left). The total number of shading samples required depend on the desired image quality, the material and the environment map. Figure 11 tabulates the computation times for our algorithm to obtain similar quality as ground truth, for several scenes. It details the cost

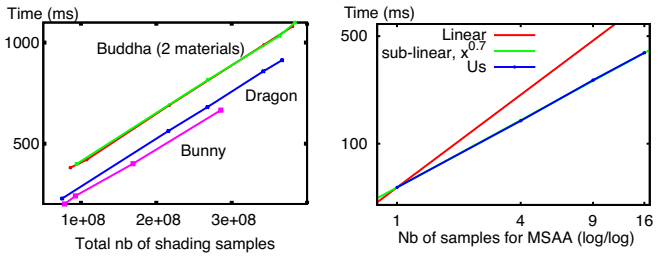


Figure 10: Left*: Rendering times are linear in the number of shading samples for various models.* Right*: Rendering time against number of samples for antialiasing (blue) in log scale. Our algorithm scales sub-linearly (with an exponent of 0.7) for antialiasing.*

Model	Car body	Bunny	Buddha	Dragon
Reference	2056	1628	2634	1739
Ours: Total	229	419	390	205
Bandwidth calculation	7	8	7	8
Shading integration	216	390	200	182

Figure 11: *Computation times (in ms) at* 512×512 *screen resolution: Car body (Fig. 1), Bunny (Fig. 13), Buddha (Fig. 12) and Dragon (Fig. 16). Our algorithm provides a* 4 *to* $10\times$ *speedup compared to a forward-shaded reference of similar quality, depending on the material and screen occupancy. Bandwidth computation is fast (*< 10 *ms) for all scenes.*

of individual steps: the cost of bandwidth computation is independent of the scene and the material, and negligible compared to the overall cost (8 ms, or $< 0.33\%$). Shading estimation consumes most of the total time (up to 90 %).

Memory cost: The memory footprint of our algorithm on the GPU is approximately 432 MB at a resolution of 512×512 pixels: 17 MB for the G-buffers (4 RGBA buffers for position, normal, tangent, material ID and the depth buffer); 2 mip-mapped buffers (5.5 MB each) for the bandwidth and variance and for the shading computations with upsampling; 2 RGB buffers of 6 MB for the environment map and its bandwidth; and 2 buffers of 196 MB for the raw BRDF data and its importance samples. Increasing the picture resolution only increases the cost of the G-buffers and mip-mapped buffers: 112 MB (instead of 22.5) for 1024×1024 and 448 MB at 2048×2048, which is currently the maximum for our algorithm.

Validation: Figure 12 compares our predictions with reference variance and image-space bandwidth. Our predictions are similar in spatial distribution, and of the same order of magnitude. Our variance estimate is conservative, as explained in Section 2.3. We computed the reference bandwidth[1] using a windowed Fourier transform over the image, with a window size of 32×32 pixels. We computed the reference variance using extensive sampling.

Influence of parameters: the main parameter for our algorithm is the number of samples we use for the bandwidth estimation, n_b (see Equation 5). Figure 13 shows the influence of varying this parameter. The results are indistinguishable even in the zoomed-in insets and the Peak Signal-Noise Ratio stays almost constant for all values of n_b. The rendering time has a small dependency on n_b. We used a small value, $n_b = 16$, for all results in this paper. This makes sense as n_b is only used to estimate the bandwidth and not for the actual illumination computations.

[1]Reference local frequencies cannot be computed exactly (uncertainty principle).

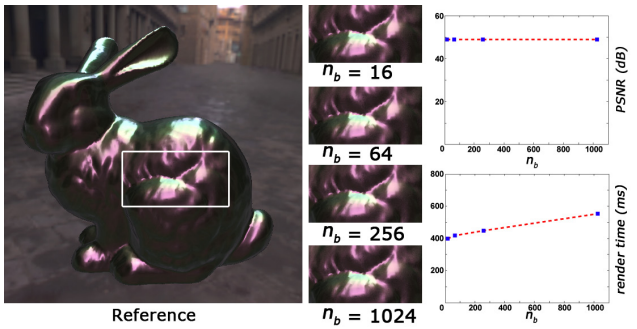
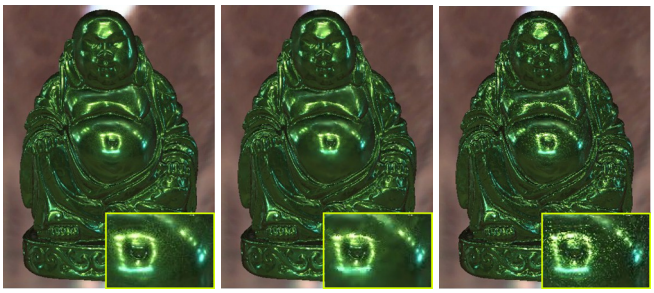


Figure 13: *Effect of the number of sampled directions* n_b *for bandwidth computation on time (ms) and quality (PSNR). The data points in the plots are at* $n_b = 16, 64, 256, 1024$*. The enlarged insets are virtually indistinguishable for the different* n_b*. The plots depict that fast bandwidth computation (*$n_b = 16$*) is sufficient. Resolution:* 512×512*. Material:* `color-changing-paint3`*.*



(a) similar quality (2639 ms) (b) Our algorithm (1015 ms) (c) similar time (906 ms)

Figure 14: *Comparison of our algorithm (b) with a reference for equal quality (a) and equal time (c). For this scene, we achieve a* $2.5\times$ *speedup (without antialiasing). Forward-shaded references use BRDF-importance sampling and a fixed number of shading samples per pixel. Material:* `green-metallic-paint`*.*

4.2 Comparison with related work

Brute-force rendering: Figure 14 compares our result with a forward shading reference computed using importance sampling and a fixed number of shading samples per pixel. For this scene, we achieve a $2.5\times$ speedup due to our adaptive sampling. The extent of our speed-up depends on the material, the environment map and the area occupied by the object on screen. Figure 11 tabulates rendering times for our algorithm and brute-force rendering for several scenes. We observe a speed-up of $4\times$ to $10\times$.

Spherical Gaussian approximation: Figure 15 compares the results of Wang *et al.* [2009] with ours and ground truth computed using path-tracing. The authors were kind enough to provide us with their best images for the materials. Our algorithm accurately shades acquired materials. The fast algorithm [Wang et al. 2009] is visibly different from the ground truth.

4.3 Discussion

Adaptive multisample anti-aliasing: Our bandwidth prediction reduces the cost of multi-sample anti-aliasing by adaptive sampling. Standard deferred shading evaluates shaders at every sample: 16 samples per pixel costs 16 times more. Our algorithm scales sub-linearly in the number of samples (see Figure 10, right). We render the G-buffers at the higher resolution (4 or 16 times the number of pixels), but compute shading at the appropriate level in the pyramid, depending on the predicted bandwidth. Antialiasing only requires

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

Figure 12: *Validation of predicted variance and bandwidth. (a) our estimate for the variance of the shading integrand (Eq. 7), (b) reference variance computed using brute-force sampling, (c) our estimate of the image bandwidth, (d) reference image bandwidth (windowed FFT of reference image), (e) our result, and (f) relative error of (e) with respect to a path-traced reference. Material:* `color-changing-paint3`

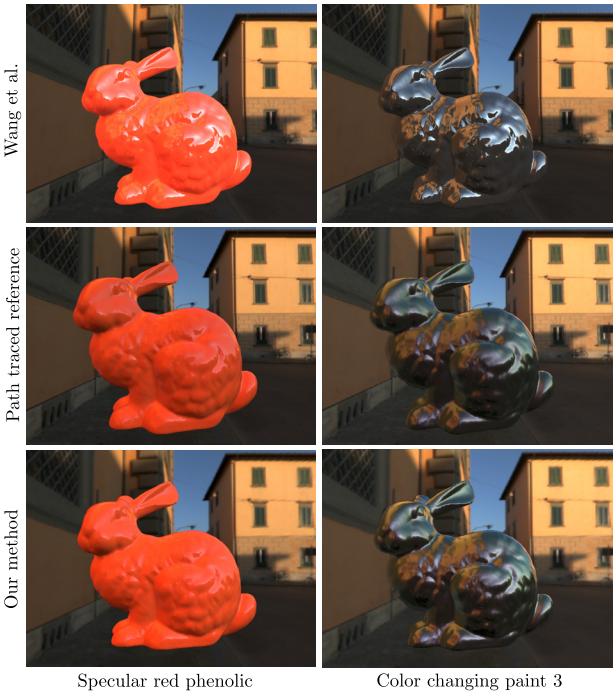


Figure 15: *Comparison with related work [Wang et al. 2009]. Our algorithm results in pictures that are identical to ground truth, while [Wang et al. 2009] result in clear differences.*

a few extra shader evaluations at the finest levels (blue pixels in Figure 16).

Dynamic geometry, normal and displacement mapping: We compute bandwidth and variance estimates using only the information from the G-buffers (normals, geometry and curvature). Our algorithm handles dynamic geometry, displacement mapping and normal maps seamlessly. The accompanying video show our algorithm running on dynamically changing shapes, at 15 fps.

Best and worst case: The rendering cost for our algorithm depends on the total number of shading samples that we predict. Our best cases are when the predicted variance is low (a specular material) or when the spatial variation is low (a smooth diffuse surface). We do not save time when both angular variance and spatial frequencies are high (a bumpy surface with high frequency illumination).

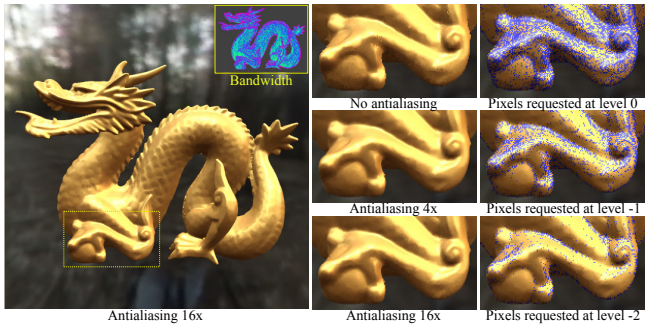


Figure 16: *Adaptive multisample anti-aliasing using bandwidth information. Aliasing artifacts are visible in the top row of enlarged insets (55 ms). Our multiresolution algorithm handles antialiasing seamlessly by adding extra levels to the pyramid (one level for* $4\times$ *and two levels for* $16\times$*). We only compute shading at the finest level for the blue pixels in the rightmost column, where the predicted bandwidth is high. The cost of anti-aliasing is thus reduced: 141 ms for* $4\times$*, 390 ms for* $16\times$*. Material:* `gold-paint`.

Conservative bandwidth estimate: We conservatively predict bandwidth, choosing suboptimal (excessive) sampling over artifacts from insufficient sampling. The min operator of Figure 5 is larger than the real bandwidth of the product of the BRDF and illumination; we also over-estimate variance.

4.4 Limitations and future avenues

Visibility and global illumination: we focus on accurate depiction of materials rather than scenes. We ignore effects such as visibility for shadows and global illumination.

Spatially-varying BRDFs: we have only used homogeneous (non spatially-varying) materials, modulated by a texture. We precomputed local bandwidths separately for reflectance ($4D$) and texture ($2D$). Extending our algorithm to fully spatially varying BRDFs ($6D$) is possible at the extra cost of 6D bandwidth precomputation.

Local light sources: we focused on multi-light settings but our algorithm can be extended to local light sources, by taking into account the spatial frequencies they introduce in the signal.

5 Conclusion

We have introduced an algorithm for interactively and accurately shading dynamic geometry with acquired materials. Our contri-

bution is to: (1) only shade a small fraction of pixels where the local bandwidth is predicted to be large; and (2) adaptively sample shading integrals based on the predicted variances. We achieve these predictions by quickly computing local bandwidth information from standard G-buffers. We have introduced the concept of a bandwidth buffer, to store this information, which is exploited to sub-linearly scale multisample antialiasing with deferred shading.

Acknowledgments

All pictures and the accompanying video were generated using the acquired materials from the MERL BRDF database [Matusik et al. 2003]. The Bunny and Dragon model are provided by the Stanford Computer Graphics Laboratory. The top row of Figure 15 was kindly provided by the authors of [Wang et al. 2009].

A Frequency localization using wavelets

Wavelets are localized both in space and frequency. We estimate the maximum frequency by examining the set of wavelets contributing to the signal s at each point$x$:

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

(where ϕ is the scale function and ψ is the mother wavelet). Wavelets of the same scale have identical bandwidths, so we compute the maximum wavelet coefficient $\lambda_i^{max} = \max_j |\lambda_{ij}|$ per frequency band, and estimate the signal bandwidth as:

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

The result is independent of ϵ as long as it is small enough. We used $\epsilon = 0.01$ 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*.

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.

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 Trans. Graph. 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 Trans. Vis. Comput. Graphics 16*, 5, 729–741.

NICHOLS, G., PENMATSA, R., AND WYMAN, C. 2010. Interactive, multiresolution image-space rendering for dynamic area lighting. *Comput. Graph. Forum 29*, 4.

OPPENHEIM, A. V., AND SCHAFER, R. W. 1999. *Discrete-Time Signal Processing*, 2nd ed. Prentice-Hall.

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

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

RAMAMOORTHI, R. 2009. Precomputation-based rendering. *Found. Trends. Comput. Graph. Vis. 3*, 4 (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. *Comput. Graph. 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*, 3 (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.

SUN, X., ZHOU, K., CHEN, Y., LIN, S., SHI, J., AND GUO, B. 2007. Interactive relighting with dynamic BRDFs. *ACM Trans. Graph. 26*, 3 (July), 27:1–27:10.

WANG, J., REN, P., GONG, M., SNYDER, J., AND GUO, B. 2009. All-frequency rendering of dynamic, spatially-varying reflectance. *ACM Trans. Graph. 28*, 5.