All-Frequency Rendering of Dynamic, Spatially-Varying Reflectance

Jiaping Wang[*] Peiran Ren[† *] Minmin Gong[*] John Snyder[‡] Baining Guo[* †]

[*] Microsoft Research Asia [†] Tsinghua University [‡] Microsoft Research

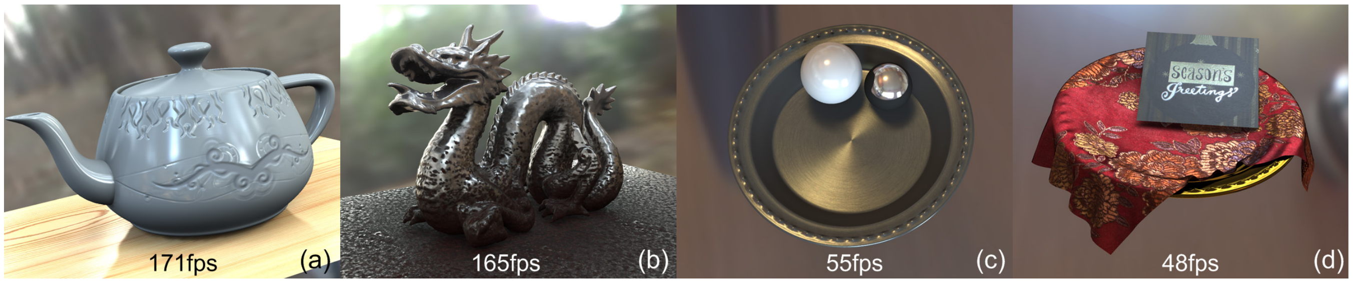

Figure 1: Real-time rendering results of our system. (a) Plastic teapot with spatially-varying Cook-Torrance model. (b) Iron dragon with spatially-varying Ward model. (c) Brushed metal dish and balls with spatially-varying Ashikhmin-Shirley model. (d) Red satin cloth and greeting card with measured SVBRDFs.

Abstract

We describe a technique for real-time rendering of dynamic, spatially-varying BRDFs in static scenes with all-frequency shadows from environmental and point lights. The 6D SVBRDF is represented with a general microfacet model and spherical lobes fit to its 4D spatially-varying normal distribution function (SVNDF). A sum of spherical Gaussians (SGs) provides an accurate approximation with a small number of lobes. Parametric BRDFs are fit on-the-fly using simple analytic expressions; measured BRDFs are fit as a preprocess using nonlinear optimization. Our BRDF representation is compact, allows detailed textures, is closed under products and rotations, and supports reflectance of arbitrarily high specularity. At run-time, SGs representing the NDF are warped to align the half-angle vector to the lighting direction and multiplied by the microfacet shadowing and Fresnel factors. This yields the relevant 2D view slice on-the-fly at each pixel, still represented in the SG basis. We account for macro-scale shadowing using a new, nonlinear visibility representation based on spherical signed distance functions (SSDFs). SSDFs allow per-pixel interpolation of high-frequency visibility without ghosting and can be multiplied by the BRDF and lighting efficiently on the GPU.

CR Categories: I.3.7 [Three-Dimensional Graphics and Realism]: Color, shading, shadowing and texture

1 Introduction

Real-world reflectance varies across object surfaces. To preserve visual realism, detailed spatial and angular reflectance variation must be considered along with natural lighting and shadowing. Such reflectance is represented by the 6D spatially varying bidirectional reflectance distribution function (SVBRDF). Reflectance also varies with time, for physical reasons such as wetting/drying, or to support interactive design and editing. Our aim is real-time rendering of direct effects from environmental lighting incident on detailed and dynamic SVBRDFs. To make this practical, we restrict ourselves to static geometry and precomputed visibility.

Precomputed Radiance Transfer (PRT) [Sloan et al. 2002; Ng et al. 2003; Ng et al. 2004] achieves real-time rendering with natural lighting and shadowing by representing the geometry's spatially-varying response to parameterized lighting. Handling detailed, dynamic SVBRDFs with all-frequency lighting has so far eluded the approach. This is because the precomputed data is huge; the more so when considering complex lighting and reflectance, sharp shadows, and glossy surfaces. To obtain reasonable data sizes and performance, some previous methods limit the directional variation of lighting, visibility and reflectance, by using bases such as spherical harmonics (SH). They thus ignore sharp shadows and highly specular surfaces. Other methods combine reflectance and visibility within the precomputed light transport and sample it at a sparse (typically per-vertex) set of surface points. This precludes BRDFs that are dynamic or have significant spatial variation. We solve the problem by decoupling reflectance from visibility in the light transport and using new, nonlinear representations for each.

Ignoring inter-reflection between objects, PRT rendering becomes:

$$R(\mathbf{o}) = \int_{\mathbb{S}^2} L(\mathbf{i})\, V(\mathbf{i})\, \rho_{\mathbf{o}}(\mathbf{i})\, \max(0, \mathbf{i}\cdot\mathbf{n})\, d\mathbf{i}. \tag{1}$$

Unit vector $\mathbf{n}$ denotes the surface normal, $\mathbf{i}$ the (input) lighting direction, and $\mathbf{o}$ the (output) view direction. L and V denote the lighting and visibility spherical functions, and ρ the 4D BRDF function. $\mathbb{S}^2$ denotes the unit sphere in $\mathbb{R}^3$.

A representation for reflectance ρ should satisfy several properties. It should fit arbitrary, including highly specular, BRDFs using a small number of parameters. It should permit fast extraction of the 2D view slice, $\rho_{\mathbf{o}}$, from the 4D BRDF. It should then allow fast integrated products of this slice with the lighting and visibility to calculate the shading. Finally, the representation should permit fast and accurate fitting to existing BRDF models.

We base our representation on the microfacet model, as the product of analytic Fresnel and self-shadowing factors with a *normal distribution function* (NDF). This reduces a 6D SVBRDF to a 4D SVNDF [Wang et al. 2008]. We further approximate the NDF as a mixture (sum) of *spherical Gaussians* (SGs), symmetric lobes

[*]email:{jiapw,mgong,johnsny,bainguo}@microsoft.com

around different axes. For parametric SVBRDFs, we derive the approximation analytically and on-the-fly at each pixel. For measured SVBRDFs, we do the fitting as a preprocess. At run-time, for every pixel, we warp the NDF to obtain the needed view-dependent BRDF slice, using a simple lobe reflection and scaling operation.

For visibility V, we use a per-vertex *spherical signed distance function* (SSDF) which encodes the angular distance to the closest visibility boundary at each direction. SSDFs allow ghost-free, per-pixel interpolation and fast integrated products with SG lobes. We use PCA to further compress this data, and decompress it on the fly on the GPU. We then compute the integrated product of four factors at each pixel: lighting, visibility, BRDF slice, and clamped cosine, where the last two are represented using SG mixtures. Lighting is represented as a filtered cubemap pyramid for an environmental source or a spherical Gaussian for a point source.

Our main contribution is to introduce a new reflectance representation for PRT: one that is structured, specialized to reflectance, and based on the microfacet model. This representation has many advantages over generic bases such as SH or wavelets. It supports reflectance of arbitrary specularity, and approximates both parametric and measured BRDF models well using just a few lobes. It can be derived on-the-fly from parametric models, allowing direct texture mapping of the model's parameters. It is closed under rotations and vector products [Tsai et al. 2008]. It allows fast extraction of the BRDF view slice using a simple spherical warp of each SG lobe, and provides fast inner/vector products with visibility and lighting. We further introduce the SSDF representation to provide per-pixel interpolation of high-frequency precomputed visibility. Our system is the first to demonstrate real-time, all-frequency PRT rendering with dynamic and detailed SVBRDFs, as shown in Figure 1.

2 Previous Work

PRT and Factorization Precomputed Radiance Transfer (PRT) [Sloan et al. 2002] combines visibility, surface reflectance, and inter-reflection as a whole light transport operator for efficiently relighting a static scene. The original method used spherical harmonics (SH) as a basis for lighting and precomputed transfer, and so supports only low-frequency (soft) shading effects. Substituting other bases, such as Haar wavelets over cube maps [Ng et al. 2003], allows "all-frequency" effects, including sharp shadows and specular reflections. Unfortunately, precomputing an object's spatially-varying response to all-frequency light requires huge data storage.

To address this problem, previous work has applied view-light (in-out) *BRDF factorization* [Wang et al. 2004; Liu et al. 2004; Wang et al. 2006; Tsai and Shih 2006]. The idea is to separate reflectance into a sum of factor pairs where one depends only on the view direction and the other only on the light direction. The lighting reflectance factor is then pre-multiplied by static visibility, which is also view-independent. This method works well for relatively diffuse BRDFs but requires impractically many terms to approximate specular surfaces [Mahajan et al. 2008]. It also mixes visibility and reflectance in the precomputed transfer, and so is limited to static, per-object, homogeneous reflectance.

PRT with Spherical Lobes Other methods use spherical lobes in PRT, as does ours. [Green et al. 2006] approximates the combined light transport due to reflectance and visibility as a sum of Gaussian lobes. The representation is unsuitable for detailed visibility functions, and produces only very soft shadows. Mixing reflectance and visibility in the precomputation means that surface reflectance must be static.

[Tsai and Shih 2006] use a scattered mixture of spherical radial basis functions (SRBFs) to represent environmental lighting. The method is based on BRDF factorization and represents the product of the light-dependent BRDF factor with visibility using fixed, rather than optimized, SRBF lobe axes. As with other BRDF factorization approaches in PRT, results are limited to homogeneous, static reflectance that is not highly specular.

[Green et al. 2007] decomposes reflectance and visibility and uses a mixture of Gaussian lobes for reflectance. A separate SG mixture is fit to each BRDF slice at a sampling of elevation angles. The representation is unsuitable for anisotropic BRDFs and requires costly precomputation and heavyweight storage even for simple isotropic ones: 45 lobes compared to our one or two. The visibility basis uses SH, limiting the technique to soft shadows.

Though we use similar basis functions, our use of SG mixtures differs from previous work. We apply the representation to surface reflectance only (rather than mixing reflectance and visibility), and do the fitting to the normal distribution function using the microfacet model. The result is much more compact than previous methods. For parametric BRDF models, we can evaluate reflectance directly from the BRDF model's parameters, on-the-fly rather than in a costly precomputation that transforms the entire 4D BRDF sample matrix into a generic basis such as SH or wavelets. We also exploit special properties of the SG basis for the first time, including fast spherical rotation/warping and products. This allows us to support fully dynamic and per-pixel varying BRDFs with all-frequency shadows in real time.

PRT with Spatially-Varying Reflectance Several PRT methods support spatially varying reflectance. [Sloan et al. 2003] decouples macro-scale from micro-scale transfer, and supports spatially varying BTFs for the micro-scale using radiance transfer textures represented in terms of spherical harmonics. To support spatial variation on deformable geometry, [Sloan et al. 2005] uses the zonal harmonic basis (a sum of broad lobes which are subsets of the spherical harmonic basis but defined around an arbitrary center) to represent micro-scale PRT. Both methods are based on low-order SH and capture only soft shading effects.

[Ng et al. 2004] uses wavelet triple products to separate visibility and reflectance and allow all-frequency, interactive rendering. In theory, such separation should permit dynamic and spatially varying reflectance. Unfortunately, the wavelet basis used does not support fast spherical rotation and in fact requires a heavyweight 6D representation that pre-rotates the BRDF. Dynamic and spatially varying reflectance is not practical.

BRDF Editing Interactive BRDF editing systems have approximated parametric BRDFs in static scenes using lower-dimensional functions (1D for isotropic and 2D for anisotropic BRDFs). [Ben-Artzi et al. 2006] represents the 1D function with a wavelet basis and precomputes light transport relative to each basis component. Dynamic BRDFs are enabled based on a 1D wavelet projection. [Ben-Artzi et al. 2008] extends the approach to include inter-reflection. Precomputation and memory costs limit these approaches to fixed views. Another technique extends factorization to represent a linear space of precomputed reflectance [Sun et al. 2007]. Lack of compactness in the BRDF representation restricts all three of these techniques to use constant rather than spatially varying reflectance over each object.

Other Methods An extension of light cuts [Walter et al. 2005; Walter et al. 2006], called visibility cuts [Cheslack-Postava et al. 2008], renders per-pixel shaded surfaces with dynamic BRDFs. Their technique is interactive but does not attain real-time performance (less than 10fps). Highly specular surfaces are problematic because importance sampling is driven by lighting rather than reflectance. Visibility cuts are sampled per-vertex, leading to ghosting artifacts if the mesh is tessellated coarsely and poor performance if it is tessellated densely. In fact, our SSDF is an example of a nonlinear visibility representation sought by [Cheslack-Postava et al. 2008] in their conclusion.

[Křivánek and Colbert 2008] importance sample analytic BRDFs on the GPU. As with much other previous work, visibility is represented with low-order SH which cannot capture all-frequency shadows. Only simple, parametric reflectance models are demonstrated. Complex and measured models must be precomputed and tabulated, limiting them to static, homogeneous (untextured) surfaces. Our microfacet-based representation supports on-the-fly evaluation of a very general class of parametric models, and compact approximation of measured BRDFs, allowing reflectance to be textured over the surface (via an SVBRDF).

[Xu et al. 2008] propose a visibility representation similar to our SSDF, but based on parametric domain distance instead of geodesic (spherical) distance. The representation is used as a replacement for summed area tables, to quickly compute average visibility over a square in the parameter domain. We exploit the idea for a new purpose: to provide a ghost-free, dense interpolation of a visibility function from a sparse set of spatial samples.

3 Spherical Gaussians

We represent reflectance and clamped cosine factors from Equation (1) as *spherical Gaussians* (SGs). SGs are a type of *spherical radial basis function* (SRBF), a family of spherical functions or *lobes* which are symmetric around a specified lobe axis. In addition to SGs, introduced in [Tsai and Shih 2006], several types of SRBFs have been studied in graphics, including the von Mises-Fisher distribution [Han et al. 2007], the Abel-Poisson kernel [Tsai and Shih 2006], zonal harmonics [Sloan et al. 2005], and high-order monomials [Arvo et al. 1994]. SGs have a number of important properties that make them particularly useful for rendering.

A spherical Gaussian has the form

$$G(\mathbf{v};\, \mathbf{p}, \lambda, \mu) = \mu\, e^{\lambda(\mathbf{v}\cdot\mathbf{p}-1)} \tag{2}$$

where $\mathbf{p} \in \mathbb{S}^2$ is the *lobe axis*, $\lambda \in (0, +\infty)$ is the *lobe sharpness*, and $\mu \in \mathbb{R}$ is the *lobe amplitude* ($\mu \in \mathbb{R}^3$ for RGB color). The direction $\mathbf{v} \in \mathbb{S}^2$ is the spherical parameter of the resulting function.

SGs are compactly ε-supported. This means that the closure of the region in which $G(\mathbf{v})$ attains values larger than a threshold ε covers less than the entire sphere. The area of this region, denoted f_a, is a function of lobe sharpness and becomes arbitrarily small as $\lambda \to \infty$:

$$f_a(\lambda) = \int_{G(\mathbf{v};\mathbf{z}^+,\lambda,1)\geq\varepsilon} \mathrm{d}\mathbf{v} = -2\pi \frac{\ln\varepsilon}{\lambda}, \qquad \lambda > -\ln\varepsilon. \tag{3}$$

Constraining $\lambda > -\ln\varepsilon$ also ensures that the lobe's thresholded support lies within a hemisphere. We will use this property later in lobe fitting and warping. Specifically, we use $\varepsilon = 0.1$ to define f_a. This value does not cause cropping of lobes in any computation, but only changes the definition of the area function.

Since an SG is symmetric around its center, its spherical rotation is given simply by a spherical rotation of $\mathbf{p}$, while preserving the values of the other two lobe parameters. Products of SGs also have a simple form. In the following, let two arbitrary SGs be given by $G_i(\mathbf{v}) = G(\mathbf{v};\, \mathbf{p}_i, \lambda_i, \mu_i)$ for $i = 1, 2$.

The *inner product*, or integral of two spherical functions yielding a scalar, is derived in [Tsai and Shih 2006] for SGs and given by

$$G_1 \cdot G_2 = \int_{\mathbb{S}^2} G_1(\mathbf{v})\, G_2(\mathbf{v})\, \mathrm{d}\mathbf{v} = \frac{4\,\pi\,\mu_1\,\mu_2}{e^{\lambda_1+\lambda_2}}\, \frac{\sinh(d_m)}{d_m} \tag{4}$$

where $d_m = \|\lambda_1 \mathbf{p}_1 + \lambda_2 \mathbf{p}_2\|$.

The *vector product*, or direction-wise multiplication of two spherical functions yielding a product spherical function, is closed in the SG basis. In other words, the vector product of two SGs is represented exactly as another SG. The product SG is given by

$$\begin{aligned}(G_1 \otimes G_2)(\mathbf{v}) &= G_1(\mathbf{v})\, G_2(\mathbf{v}) \\ &= G\Big(\mathbf{v};\, \frac{\mathbf{p}_m}{\|\mathbf{p}_m\|},\, \lambda_m \|\mathbf{p}_m\|,\, \mu_1\,\mu_2\, e^{\lambda_m(\|\mathbf{p}_m\|-1)}\Big)\end{aligned} \tag{5}$$

where $\mathbf{p}_m = (\lambda_1\, \mathbf{p}_1 + \lambda_2\, \mathbf{p}_2)/(\lambda_1 + \lambda_2)$ and $\lambda_m = \lambda_1 + \lambda_2$. A similar result is derived in [Tsai et al. 2008, Eqn. 10], but ignoring proper normalization. Our derivation appears in the appendix.

By summing SGs around various lobes, called a *mixture model of scattered SGs*, or more simply an *SG mixture*, we can represent an arbitrary smooth spherical function to any degree of accuracy. Mathematically, our model is given by

$$F^*(\mathbf{v}) = \sum_{i=1}^{n} G(\mathbf{v}; \mathbf{p}_i, \lambda_i, \mu_i). \tag{6}$$

An SG mixture is closed under rotations. A rotated version of an SG mixture $F^*(\mathbf{i})$ is computed by rotating its lobe centers via

$$\mathfrak{R} F^*(\mathbf{v}) = \sum_{i=1}^{n} G(\mathbf{v},\, \mathfrak{R}\,\mathbf{p}_i,\, \lambda_i,\, \mu_i) \tag{7}$$

where $\mathfrak{R}$ denotes the rotation matrix.

The SG mixture basis is not orthogonal. An inner or vector product of two, n-term SG mixtures must consider all pairs of SG lobes and so has complexity $O(n^2)$. We avoid the difficulty by using SG mixtures only for sparse spherical functions requiring small n. Complex spherical functions such as visibility and environmental lighting incident on specular surfaces are represented using a different representation, as we will see.

4 Reflectance Representation using SGs

We decompose the BRDF $\rho(\mathbf{o}, \mathbf{i})$ at a surface point into a diffuse component k_d and a specular component $k_s \rho_s$:

$$\rho_{\mathbf{o}}(\mathbf{i}) = \rho(\mathbf{o}, \mathbf{i}) = k_d + k_s\, \rho_s(\mathbf{o}, \mathbf{i}). \tag{8}$$

These two components of shading are calculated separately. The following describes how the specular part is represented and how the BRDF slice for one viewing direction $\mathbf{o}$ is evaluated.

4.1 Parametric BRDFs

We focus on BRDF models based on microfacet theory [Torrance and Sparrow 1967], which includes a large variety of commonly-used reflectance types. The specular lobe ρ_s in microfacet theory can be formalized as

$$\rho_s(\mathbf{o}, \mathbf{i}) = M_{\mathbf{o}}(\mathbf{i})\, D(\mathbf{h}) \qquad \mathbf{h} = \frac{\mathbf{o}+\mathbf{i}}{\|\mathbf{o}+\mathbf{i}\|}, \tag{9}$$

which expresses reflectance in terms of the normal distribution function (NDF), $D(\mathbf{h})$, and the remaining factor, $M_{\mathbf{o}}(\mathbf{i})$, combining microfacet shadowing and Fresnel reflection. These latter factors are very smooth [Ashikmin et al. 2000; Ngan et al. 2005], and so can be compactly approximated, while the NDF often contains high-frequency information. We model the NDF $D(\mathbf{h})$ using a single SG lobe for isotropic models or multiple lobes for anisotropic models.

For example, the Cook-Torrance model [Cook and Torrance 1981] is represented in terms of a roughness parameter m:

$$\rho_s(\mathbf{o}, \mathbf{i}) = \frac{F_{\mathrm{CT}}(\mathbf{o}, \mathbf{i})\, S_{\mathrm{CT}}(\mathbf{o}, \mathbf{i})}{\pi (\mathbf{n}\cdot\mathbf{i})\,(\mathbf{n}\cdot\mathbf{o})}\, e^{-(\theta/m)^2}, \tag{10}$$

$$M_{\mathbf{o}}(\mathbf{i}) = \frac{F_{\mathrm{CT}}(\mathbf{o}, \mathbf{i})\, S_{\mathrm{CT}}(\mathbf{o}, \mathbf{i})}{\pi (\mathbf{n}\cdot\mathbf{i})\,(\mathbf{n}\cdot\mathbf{o})}, \qquad D(\mathbf{h}) = e^{-(\arccos(\mathbf{h}\cdot\mathbf{n})/m)^2}. \tag{11}$$

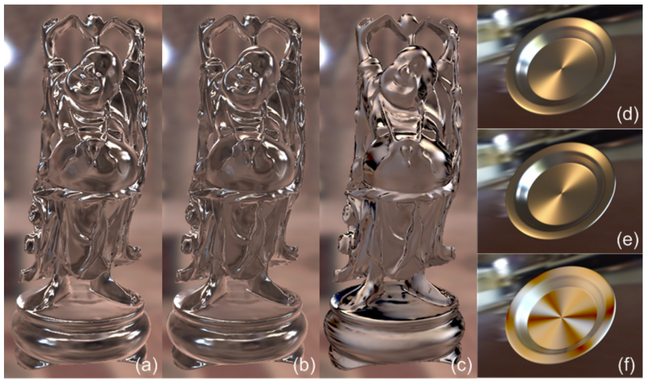

Figure 2: Fitting SG mixtures to parametric BRDF models (unshadowed rendering). The three left columns show the Cook-Torrance model ($m = 0.045$): (a) ground truth, (b) single-lobe SG model, (c) 256-term BRDF factorization. The right column shows the Ashikhmin-Shirley model ($n_u = 75, n_v = 1200$): (d) ground truth, (e) 7-lobe SG model, (f) 256-term BRDF factorization.

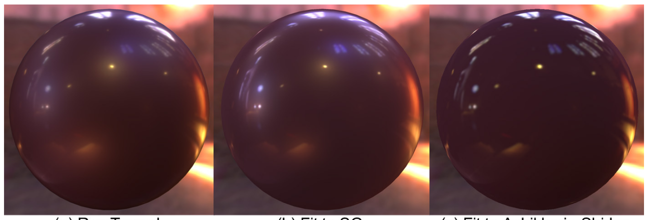

Figure 3: BRDF fitting, violet acrylic example from [Matusik et al. 2003]. Our method (3 SG lobes) is compact and more accurate than fitting parametric models like Ashikhmin-Shirley in (c).

The functions F_{CT} and S_{CT} denote the Fresnel and shadowing functions, respectively, and have an analytic form. We approximate the NDF with a single SG lobe via

$$D(\mathbf{h}) = e^{-(\arccos(\mathbf{h}\cdot\mathbf{n})/m)^2} \approx G(\mathbf{h};\mathbf{n},2/m^2,1). \tag{12}$$

This approximation matches the original model quite well; a detailed error analysis is provided in the supplemental material. Other frequently-used parametric BRDF models such as (isotropic) Blinn-Phong [Blinn 1977], isotropic and anisotropic Ward [Ward 1992], and (anisotropic) Ashikhmin-Shirley [Ashikhmin and Shirley 2000] can also be represented using (9). Analytic approximations for these models using SG fits to the NDF, similar to (12), are given in the supplement.

Spatial variation is encoded simply by texturing over the surface the parameters k_s, k_d from (8) as well as the specific BRDF model parameters, such as m for Cook-Torrance. This is a compact representation that allows high-frequency spatial variation. Surface reflectance at each pixel is determined on-the-fly in rendering without any precomputation or fitting, which allows arbitrary temporal and spatial changes to reflectance in runtime.

Figure 2 shows parametric BRDF models represented and rendered with our method. The SG mixture model achieves better rendering quality than does BRDF factorization even with 256 terms. Such a large number of terms produces an extremely heavyweight representation that prohibits spatial variation or interactive performance. To put this number in perspective, previously published PRT factorization methods have used many fewer terms: 4 in [Wang et al. 2004], 10 in [Liu et al. 2004], and 16 in [Tsai and Shih 2006].

4.2 Measured BRDFs

We represent measured BRDFs with the general microfacet model using a tabulated NDF D and shadowing factor S at each surface

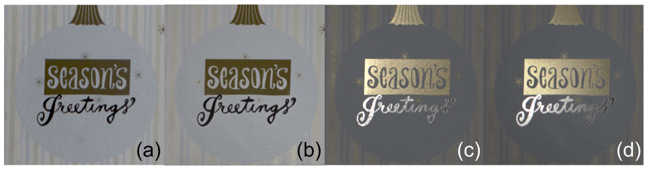

Figure 4: SVBRDF fitting on a measured example from [Lawrence et al. 2006]. Two light/view settings are shown: (a,c) original, (b,d) our approximation with 2 SG lobes per pixel.

point. The representation is similar to [Wang et al. 2008] except that we then fit the NDF with a small number of SG lobes. The overall model is given by:

$$\rho_s(\mathbf{o},\mathbf{i}) = \frac{F(\mathbf{o},\mathbf{i})S(\mathbf{o})S(\mathbf{i})}{\pi(\mathbf{i}\cdot\mathbf{n})(\mathbf{o}\cdot\mathbf{n})} D(\mathbf{h}). \tag{13}$$

The Fresnel factor F is given by [Cook and Torrance 1981]:

$$F(\mathbf{o},\mathbf{i}) = \frac{(g-c)^2}{2(g+c)^2}\left(1+\frac{\left(c(g+c)-1\right)^2}{\left(c(g-c)+1\right)^2}\right) \tag{14}$$

where $g = \sqrt{\eta^2+c^2-1}$, $c = |\mathbf{i}\cdot\mathbf{h}|$, and η denotes the relative index of refraction. As in [Wang et al. 2008], we constrain the shadowing factor to be isotropic. The resulting 1D shadowing function at each surface point is packed into 90D vectors and compressed to 8D using PCA. The compressed vector and the relative index of refraction can be stored in textures for rendering. We currently use a constant index of refraction $\eta = 1.33$. Figures 3 and 4 show results of our microfacet-based SG mixture representation on real measured BRDFs and SVBRDFs. More fitting results are included in the supplement.

For isotropic BRDFs, we fit the NDF $D(\mathbf{h})$ in (13) with SG mixtures having 1 or 2 lobes around the canonical axis $\mathbf{p} = \mathbf{z}^+$ as a preprocess. The fitting is done by Levenberg-Marquardt optimization [Nocedal and Wright 1999]. When SG mixtures have more than one lobe, the lobes are sorted in increasing order of sharpness for texture interpolation.

For anisotropic BRDFs, more lobes are typically required and are fit using a variable axis rather than the fixed $\mathbf{z}^+$. We fix the number of lobes in the SG approximation for all surface points and then compute an independent nonlinear optimization to fit each point's NDF using the L-BFGS-B solver [Zhu et al. 1997] as in [Tsai and Shih 2006]. The resulting parameters of the SG mixtures are encoded with multiple textures for rendering. Independent optimization produces unaligned lobes at adjacent texels whose axes may be far apart and thus unsuitable for interpolation. In this case, we avoid texture filtering and instead apply nearest-neighbor texture sampling and spatial supersampling.

After fitting the NDF, the resulting SG parameters are stored as a texture map. The representation requires 4 floats for each lobe in the isotropic case (3 for RGB lobe amplitude and 1 for lobe sharpness). An additional 3 floats are needed in the anisotropic case to encode the lobe direction.

4.3 Obtaining the View Slice using a Spherical Warp

Given a view direction $\mathbf{o}$, we require the corresponding 2D BRDF slice $\rho_s(\mathbf{i};\mathbf{o})$ to integrate with visibility and lighting. We obtain it by a vector product of the NDF's SG approximation, $D^*(\mathbf{h}) = \sum_{i=1}^{n} G\left(\mathbf{h};\mathbf{p}_i^D,\lambda_i^D,\mu_i^D\right)$, with the remaining factor $M_{\mathbf{o}}(\mathbf{i})$:

$$\rho_s(\mathbf{i};\mathbf{o}) = M_{\mathbf{o}}(\mathbf{i}) \otimes D^*(\mathbf{h}). \tag{15}$$

To compute this, we first express the NDF in terms of the lighting vector $\mathbf{i}$ instead of the half-way vector $\mathbf{h}$:

$$W^*(\mathbf{i}) = \sum_{i=1}^{n} G\left(\,\mathbf{i};\mathbf{p}_i^W,\lambda_i^W,\mu_i^W\,\right) \approx D^*(\mathbf{h}). \tag{16}$$

This involves a *spherical warp* on D^* via the transformation

$$\mathbf{i} = \psi(\mathbf{h}) = 2(\mathbf{o}\cdot\mathbf{h})\mathbf{h} - \mathbf{o}. \tag{17}$$

We can approximate this warp as a sum of simple, per-lobe warps on D^*. Each warped lobe has ε-support area in Equation (3) equal to its original unwarped area times the differential area of the warp at the lobe center, $\mathbf{p}_i^D$. The warped lobe's amplitude is preserved. The formula is given by

$$\mathbf{p}_i^w = \psi(\mathbf{p}_i^D) = 2(\mathbf{o}\cdot\mathbf{p}_i^D)\mathbf{p}_i^D - \mathbf{o}, \tag{18}$$

$$\lambda_i^w = f_a^{-1}\big(f_a(\lambda_i^D)\cdot\tau(\mathbf{p}_i^D)\big) = \frac{\lambda_i^D}{\tau(\mathbf{p}_i^D)}, \tag{19}$$

$$\mu_i^w = \mu_i^D. \tag{20}$$

Differential area of this warp, $\tau(\mathbf{h})$, is the determinant of the Jacobian of the warp function ψ and given by

$$\tau(\mathbf{h}) = \frac{d\mathbf{i}}{d\mathbf{h}} = \frac{\left\| \frac{\partial\psi(\mathbf{h})}{\partial\theta} \times \frac{\partial\psi(\mathbf{h})}{\partial\phi} \right\|}{\left\| \frac{\partial\mathbf{h}}{\partial\theta} \times \frac{\partial\mathbf{h}}{\partial\phi} \right\|} = 4\,|\mathbf{h}\cdot\mathbf{o}| \tag{21}$$

where $\mathbf{h}(\theta,\phi) = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$. Other types of per-lobe warps can be computed similarly by replacing the transform ψ and its distortion factor τ.

Finally, we observe that the factor $M_{\mathbf{o}}(\mathbf{i})$ is smooth [Ngan et al. 2005]. Our approximation assumes $M_{\mathbf{o}}$ is constant across the support of each SG in the warped NDF W^*, so that the lobe amplitude is simply multiplied by the value of $M_{\mathbf{o}}$ evaluated at the lobe center, yielding the following simple formula:

$$\rho_s^*(\mathbf{i};\mathbf{o}) = M_{\mathbf{o}}(\mathbf{i}) \otimes W^*(\mathbf{i}) \approx \sum_{i=1}^{n} G\big(\,\mathbf{i};\ \mathbf{p}_i^w, \lambda_i^w, M_{\mathbf{o}}(\mathbf{p}_i^w)\,\mu_i^w\,\big). \tag{22}$$

Our fast method for spherical warping yields a BRDF slice represented as an SG mixture. Since we approximate each warped lobe as isotropic, our method incurs error especially at oblique viewing directions where the warp has large anisotropic distortion. Figures 3 and 4 include error from the fast spherical warp, as well as error from fitting our representation. The visual difference is small. Error from spherical warping is further analyzed in the supplement. Future work could reduce this error by fitting multiple-lobe SGs to the warped result, or using anisotropic basis functions.

5 Visibility Representation using SSDFs

We represent spatially-varying visibility with a nonlinear representation called a *spherical signed distance function* (SSDF). An SSDF is stored at each mesh vertex and is interpolated per-pixel over triangles. The benefit of this representation is that it provides ghost-free interpolation as shown in Figure 5. Visibility edges (row 3), reconstructed with bilinear interpolation on SSDFs (row 2), are sharp yet still make smooth transitions. Direct interpolation of the binary function (row 1) yields double images.

As is usual in PRT methods, the mesh tessellation must adequately sample this visibility signal. For example, representing a floor as a pair of triangles with only four SSDFs at its vertices will not permit detailed shadows cast by objects resting on it. Unlike previous PRT work, the tessellation can ignore variation in reflectance, which is handled by texture mapping SG mixtures. The SSDF representation permits spatially-varying visibility to be represented at a relatively sparse set of surface vertices.

5.1 Spherical Signed Distance for Visibility

Visibility, $V_{\boldsymbol{x}}(\mathbf{i})$, is a spatially-varying binary spherical function indicating whether the direction $\mathbf{i}$ to a distant light source is occluded or not, at each surface point $\boldsymbol{x}$. To make the notation clearer, we will drop the $\boldsymbol{x}$ subscript. We map binary visibility to an SSDF, $V^d(\mathbf{i})$. The function's sign encodes whether the direction $\mathbf{i}$ is occluded (negative) or not (positive), and its value encodes angular distance, θ_d, to the nearest direction $\mathbf{t}$ on the shadow boundary. An example is shown in Figure 6a.

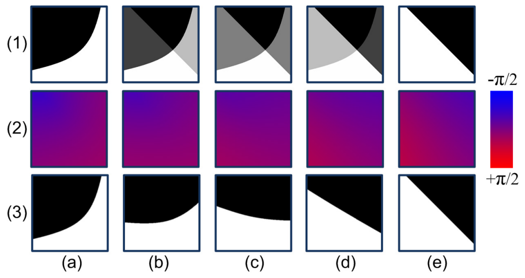

Figure 5: Ghost-free visibility interpolation using nonlinear SSDFs. Two visibility functions (top row, a1 and e1) are linearly interpolated with blending weights 0.25 in (b), 0.5 in (c) and 0.75 in (d), resulting in ghosting artifacts. SSDFs corresponding to (a1) and (e1) are shown in (a2) and (e2), while (b2), (c2), and (d2) are the linearly blended SSDFs. Reconstruction from interpolated SSDFs, shown in (b3), (c3), and (d3), yields smooth transitions in orientation while preserving edge sharpness.

Mathematically, the SSDF is given by

$$V^d(\mathbf{i}) = \begin{cases} +\min\limits_{V(\mathbf{t})=0} \arccos(\mathbf{t}\cdot\mathbf{i}), & \text{if } V(\mathbf{i}) = 1; \\ -\min\limits_{V(\mathbf{t})=1} \arccos(\mathbf{t}\cdot\mathbf{i}), & \text{if } V(\mathbf{i}) = 0. \end{cases} \tag{23}$$

When multiplying an SSDF by an SG around a lobe center $\mathbf{p}$, evaluating the SSDF $V^d(\mathbf{p})$ essentially reconstructs a hemispherical blocker based on the closest occluder to $\mathbf{p}$ (compare a and b in Figure 6). For narrow SG lobes, this provides a good approximation to visibility in the neighborhood of $\mathbf{p}$, For broader lobes, detailed knowledge of the visibility function is unnecessary, so the approximation is reasonable in that case as well.

5.2 Precomputing and Compressing SSDFs

As a preprocess, we convert the binary visibility function to an SSDF using equation (23) at each vertex. We sample only the upper normal hemisphere and reparameterize in a square image [Shirley and Chiu 1997]. To avoid aliasing, we downsample an initial 512×512 image to 128×128.

We then compress the spatially-varying SSDF using PCA:

$$V_{\boldsymbol{x}}^d(\mathbf{i}) \approx \sum_{j=1}^{N_V} \mathbb{V}_j(\mathbf{i})\, w_{\boldsymbol{x},j}^V \tag{24}$$

where N_V denotes the number of PCA terms and $\mathbb{V}_j$ is the j-th eigenvector. The PCA coefficients $w_{\boldsymbol{x},j}^V$ are stored as vertex attributes and interpolated to each pixel during rasterization. The eigenvectors are encoded in multiple textures and fetched by bilinear filtering. Our current implementation uses $N_V = 48$. More sophisticated compression techniques, such as local PCA or tensor compression, could replace this simple approach.

5.3 Integrating Products with Visibility

We are given an SG lobe with center $\mathbf{p}$ and wish to compute products between this lobe and an SSDF. This is done by reconstructing a hemispherical visibility function and integrating the lobe over the intersection of its hemispherical support (around the lobe axis $\mathbf{p}$) with the visibility hemisphere (whose boundary passes through $\mathbf{t}$ nearest to $\mathbf{p}$). The absolute direction $\mathbf{t}$ relative to $\mathbf{p}$ is irrelevant since SGs are isotropic; only the angle between them matters: $\theta_d = V^d(\mathbf{p})$.

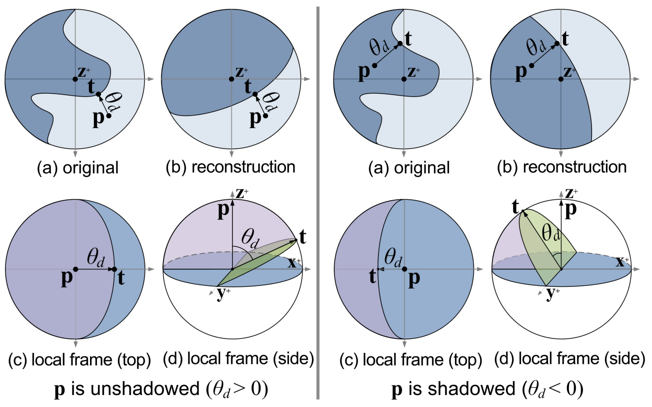

Figure 6: Center-dependent visibility reconstruction with an SSDF. Two cases are shown: on the left, signed distance is positive (lobe axis **p** is in visible region), while on the right it is negative (lobe axis is occluded). (a) Original visibility function (top view). **t** is the nearest visibility boundary to the lobe center **p**. (b) Reconstructed visibility function around **p**. The visible region is a hemisphere whose boundary intersects **t**. (c) Reconstructed visible region (purple) in the local coordinate frame (top view). (d) Reconstructed visible region (side-view). $\mathbf{z}^+$ points to the lobe center and $\mathbf{y}^+$ points to the cross product of **t** and **p**.

Let the reconstructed visibility function be denoted $V'(\mathbf{i})$. For convenience, we represent it in the local frame of the SG lobe as shown in Figure 6cd. This coordinate frame orients **p** along $\mathbf{z}^+$ and $\mathbf{t}\times\mathbf{p}$ along $\mathbf{y}^+$. In this coordinate system, the visibility domain is the purple region and defined as:

$$V'(\mathbf{i}) = \begin{cases} 1, & \delta(\mathbf{i}) \geq \frac{\pi}{2} - \theta_d \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

where δ is the elevation angle of **i** when projected into the xz plane:

$$\delta(\mathbf{i}) = \frac{\pi}{2} - \arctan\left(\frac{\mathbf{i}\cdot\mathbf{x}^+}{\mathbf{i}\cdot\mathbf{z}^+}\right). \tag{26}$$

To compute inner products, we parameterize the interior of the visible region via

$$\mathbf{i}(\xi,\delta) = \left(\sin\xi\sin\delta,\, \cos\xi,\, \sin\xi\cos\delta\right). \tag{27}$$

Parameter $\xi \in [0,\pi]$ represents the angle made by **i** with $\mathbf{y}^+$ while $\delta \in [0,\pi]$ represents the angle around $\mathbf{y}^+$. The inner product is then given by

$$G(\mathbf{i};\mathbf{p},\lambda,\mu)\cdot V(\mathbf{i}) \approx G(\mathbf{i};\mathbf{p},\lambda,\mu)\cdot V'(\mathbf{i}) \tag{28}$$

$$= \mu \int_{\delta_0}^{\pi}\int_0^{\pi} G\left(\mathbf{i};\mathbf{z}^+,\lambda,1\right)\,\sin\xi\,\mathrm{d}\xi\,\mathrm{d}\delta \tag{29}$$

$$= \mu\, f_h(\theta_d,\lambda). \tag{30}$$

where $\delta_0 = \frac{\pi}{2} - \theta_d$. We precompute the 2D scalar function $f_h(\theta_d,\lambda)$ and approximate it using a simple model consisting of a sigmoid composed with a polynomial, as described in the appendix.

The vector product is approximated by making its amplitude yield the same integrated value as the original lobe and preserving its center. This yields the formula

$$G(\mathbf{i};\mathbf{p},\lambda,\mu)\otimes V(\mathbf{i}) \approx G\left(\mathbf{i};\mathbf{p},\lambda,\frac{f_h(\theta_d,\lambda)}{f_h(\frac{\pi}{2},\lambda)}\,\mu\right) \tag{31}$$

where $f_h(\frac{\pi}{2},\lambda) = \int_{\mathbb{S}^2} G(\mathbf{i};\mathbf{z}^+,\lambda,1)\,\mathrm{d}\mathbf{v} = (2\pi/\lambda)(1-e^{-\lambda})$.

Figure 7 compares our SSDF representation and method for computing products with ray tracing. For narrow SGs (a), our approximation yields a small error and little visual difference to the ground truth. Broader SGs (c) produce a very soft shadow making the small shift that occurs essentially imperceptible. Refer to the supplement for a more detailed error analysis.

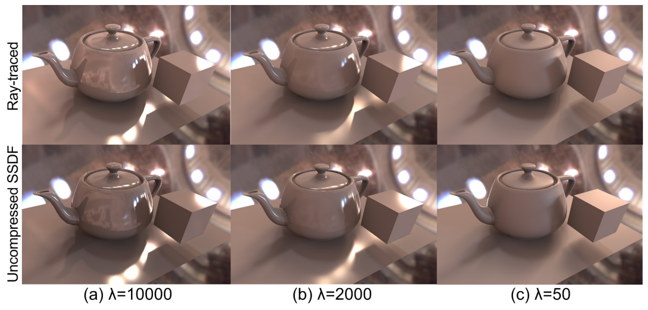

Figure 7: SSDF product accuracy. A good approximation is produced for different values of lobe sharpness, λ. The figure is rendered with a single-lobe fit to the Cook-Torrance BRDF.

6 Lighting Representation

Local Point Lights can be dynamically added and animated in our system and are approximated with a single-lobe SG model. Such a light source yields a spatially varying radiance field which we compute per pixel. A point light source located at 3D position $\boldsymbol{l}$ with radius r and intensity s yields a radiance field at surface location $\boldsymbol{x}$ that we approximate by

$$L^*_{\boldsymbol{x}}(\mathbf{i}) = G\left(\mathbf{i};\, \frac{\boldsymbol{l}-\boldsymbol{x}}{\|\boldsymbol{l}-\boldsymbol{x}\|},\, f_a^{-1}\left(\frac{2\pi r^2}{\|\boldsymbol{l}-\boldsymbol{x}\|^2}\right),\, \frac{s}{\|\boldsymbol{l}-\boldsymbol{x}\|^2}\right). \tag{32}$$

The radius parameter actually models a spherical light source: increasing r increases the softness of shadow boundaries.

For infinitely-distant lighting from direction **l**, the SG approximation is given by

$$L^*(\mathbf{i}) = G\left(\mathbf{i};\mathbf{l},\, f_a^{-1}(2\pi r^2),\, s\right). \tag{33}$$

Distant Environmental Lighting is represented by the spherical radiance function $L(\mathbf{i})$. We use two different representations for L depending on whether the lighting is applied to the diffuse or specular surface component. For the diffuse component, we fit the environmental radiance map with an SG mixture $L^*(\mathbf{i})$ as in [Tsai and Shih 2006]. Preserving just the brightest areas is sufficient for diffuse rendering and requires only a few lobes (< 10). Of course, this ignores details in the environment which are revealed in specular reflections.

For the specular component, we preconvolve environmental radiance with SG kernels of varying lobe sharpness to permit fast run-time evaluation of inner products. The idea is similar to prefiltered environment maps [Kautz et al. 2000; McAllister et al. 2002] used for unshadowed rendering, and applied to low-frequency shadowed rendering in [Green et al. 2006], but instead based on our SG kernel. The inner product of lighting with an arbitrary SG is given by

$$G(\mathbf{i};\mathbf{p},\lambda,\mu)\cdot L(\mathbf{i}) = \mu \int_{\mathbb{S}^2} G(\mathbf{i};\mathbf{p},\lambda,1)\,L(\mathbf{v})\,\mathrm{d}\mathbf{i} \tag{34}$$

$$= \mu\,\Gamma_L(\mathbf{p},\lambda). \tag{35}$$

The 3D function Γ_L is precomputed and tabulated in terms of λ and **p**. The result is represented as an image pyramid (MIPMAP), indexed by λ, over cubemaps each representing a spherical function indexed by **p**.

7 Run-Time Rendering

Per-vertex mesh information needed by the renderer includes position, texture coordinates, local coordinate frame, and PCA coefficients $w^V_{\boldsymbol{x},j}$ for the SSDF from equation (24). These quantities are interpolated across mesh triangles and used by the pixel shader. Textures store the BRDF parameters or, for non-parametric BRDFs, the tabulated SG lobes, D^*, and PCA-compressed shadow factor, $\hat{S}$.

We separate the rendering equation into a view-independent diffuse component, R_d, and view-dependent specular component, $R_s(\mathbf{o})$, by plugging equation (8) into equation (1) to obtain:

$$R(\mathbf{o}) = k_d R_d + k_s R_s(\mathbf{o}), \tag{36}$$

$$R_d = \int_{\mathbb{S}^2} L(\mathbf{i}) V(\mathbf{i}) \max(0, \mathbf{i}\cdot\mathbf{n})\, \mathrm{d}\mathbf{i}, \tag{37}$$

$$R_s(\mathbf{o}) = \int_{\mathbb{S}^2} L(\mathbf{i}) \rho_s(\mathbf{o},\mathbf{i}) V(\mathbf{i}) \max(0, \mathbf{i}\cdot\mathbf{n})\, \mathrm{d}\mathbf{i}. \tag{38}$$

The two components are evaluated, summed, and tone-mapped in a single shading pass.

The diffuse component is calculated as

$$R_d = \left(C^*(\mathbf{i};\mathbf{n}_{\boldsymbol{x}}) \otimes L^*(\mathbf{i})\right) \cdot V^d_{\boldsymbol{x}}(\mathbf{i}) \tag{39}$$

where $C^*(\mathbf{i};\mathbf{n}_{\boldsymbol{x}})$ is an SG mixture approximation of the clamped cosine, $\max(0, \mathbf{i}\cdot\mathbf{n}_{\boldsymbol{x}})$. We fit it as a single-lobe SG:

$$C^*(\mathbf{i};\mathbf{n}_{\boldsymbol{x}}) = G(\mathbf{i};\mathbf{n}_{\boldsymbol{x}},\lambda_c,\mu_c), \quad \lambda_c = 2.133, \quad \mu_c = 1.170, \tag{40}$$

where $\mathbf{n}_{\boldsymbol{x}}$ is the per-pixel interpolated surface normal. Error of this single lobe fit is analyzed in the supplement. We represent visibility $V^d_{\boldsymbol{x}}$ as an SSDF in the local coordinate frame of the vertex and decode its PCA representation as explained in Section 5.2. The SG mixture for the lighting is rotated to the local frame before computing the vector product with the clamped cosine. Finally, we apply equation (30) to compute the inner product of this result with visibility. We approximate f_h using the simple formula described in the appendix.

The specular component is calculated as

$$R_s(\mathbf{o}) = \left(C^*(\mathbf{i};\mathbf{n}_{\boldsymbol{x}}) \otimes \rho^*_{s,\boldsymbol{x}}(\mathbf{i};\mathbf{o}) \otimes V^d_{\boldsymbol{x}}(\mathbf{i})\right) \cdot L(\mathbf{i}). \tag{41}$$

The BRDF slice ρ^*_s is defined in the local coordinate system, as is the clamped cosine and visibility factors. The vector product of these three is then rotated to the world coordinate system and an inner product with environmental radiance computed via equation (35). By prefiltering the environment map, this inner product reduces to a simple "MIPMAP of cubemaps" texture fetch for each SG lobe in the vector product.

We determine the SG mixture for the 2D BRDF slice $\rho^*_{s,\boldsymbol{x}}$ on-the-fly by a spherical warp of $D^*_{\boldsymbol{x}}(\mathbf{h})$ according to equation (22). $M_{\boldsymbol{x},\mathbf{o}}(\mathbf{i})$ has an analytic form that is evaluated in a pixel shader. The NDF $D^*_{\boldsymbol{x}}(\mathbf{h})$ is represented as a single-lobe SG for isotropic, parametric BRDFs, a two-lobe SG for isotropic, non-parametric models, and 4-8 lobes for anisotropic models. For parametric models, D^* is computed per-pixel as an analytic function of the BRDF parameters for the current rasterization fragment. For non-parametric models, it is indexed from a texture.

Local Light Sources are represented as a single-lobe SG using equation (32) for point lights or (33) for directional lights. Products are computed differently than for environmental lighting. In this case, we rotate the simple lighting to the local coordinate system of the surface rather than rotating the reflectance and visibility to the world coordinate system. The rendering equations, (39) and (41), are modified to become:

$$R_d = \left(L^*_{\boldsymbol{x}}(\mathbf{i}) \otimes C^*(\mathbf{i};\mathbf{z}^+)\right) \cdot V^d_{\boldsymbol{x}}(\mathbf{i}) \tag{42}$$

$$R_s(\mathbf{o}) = \left(L^*_{\boldsymbol{x}}(\mathbf{i}) \otimes C^*(\mathbf{i};\mathbf{z}^+) \otimes \rho^*_{s,\boldsymbol{x}}(\mathbf{i};\mathbf{o})\right) \cdot V^d_{\boldsymbol{x}}(\mathbf{i}) \tag{43}$$

This avoids a vector product with visibility.

Point light sources should be located outside the object's convex hull since precomputed visibility represents occlusion to distant lighting only.

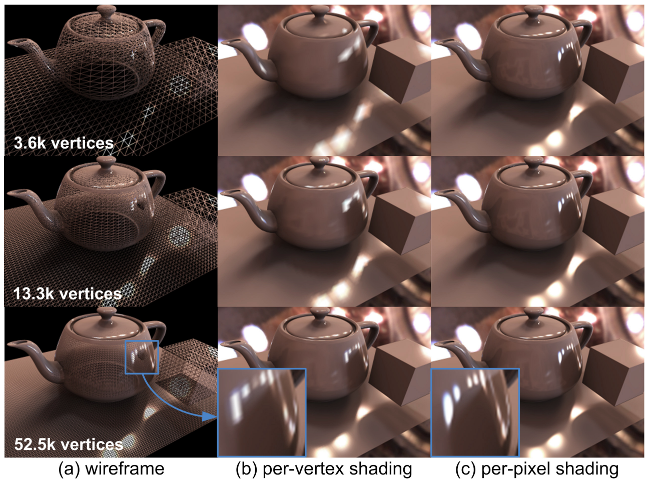

Figure 8: Per-vertex vs. per-pixel shading.

Normal and Tangent Maps simulating surface meso-structure are easily integrated. We generalize both by a quaternion map $\mathbf{q}(\boldsymbol{x})$, which describes a rotation matrix $\mathfrak{R}_{\boldsymbol{x}}$ defined in the local coordinate frame at each surface point $\boldsymbol{x}$. For diffuse rendering, the clamped cosine $C^*_{\boldsymbol{x}}(\mathbf{i})$ is replaced by its rotated version $\mathfrak{R}_{\boldsymbol{x}} C^*_{\boldsymbol{x}}(\mathbf{i})$ in equations (39) and (42). Rotation of an SG mixture is computed via equation (7). For specular rendering, the vector product $C^*_{\boldsymbol{x}}(\mathbf{i}) \otimes \rho^*_{s,\boldsymbol{x}}(\mathbf{i};\mathbf{o})$ is replaced by its rotated version $\mathfrak{R}_{\boldsymbol{x}}\left(C^*_{\boldsymbol{x}}(\mathbf{i}) \otimes \rho^*_{s,\boldsymbol{x}}(\mathbf{i};\mathbf{o}')\right)$ in equations (41) and (43). The viewing direction $\mathbf{o}$ for the rotated BRDF slice must also be changed to $\mathbf{o}' = \mathfrak{R}_{\boldsymbol{x}}\,\mathbf{o}$. Since the clamped cosine and BRDF slice are determined on the fly anyway, this approach allows run-time changes to the mapped normals and/or tangents.

8 Experimental Results and Discussion

We implemented our rendering algorithm on a PC with an Intel Core$^{\text{TM}}$2 Duo 3.2G CPU, 4GB memory, and an nVidia Geforce 8800Ultra graphics card. We used the GPU rasterizer to sample the visibility function at each vertex and then convert it to an SSDF as described in Section 5.2. The PCA coefficients of the compressed SSDFs are stored as vertex attributes and interpolated during rasterization. Distant environmental and local point light sources are both rendered in a single pass of a pixel shader.

Table 7 lists the precomputation and run-time performance statistics for all our examples. Detailed textures for parametric BRDFs (first three rows) are specified directly in terms of the model parameters, require only a few MB of memory, and need no precomputation. SG mixture fits to measured BRDFs must be precomputed, requiring 2 hours for the "greeting card" example [Lawrence et al. 2006] and 5 hours for the red satin and green velvet examples [Wang et al. 2008] in Figure 12d-h.

Decoupling textured reflectance from visibility and using our SSDF visibility representation allows sparse geometry tessellations (see the "Vert." column in the table). This reduces precomputation time and memory, and also increases run-time performance. Without complicated optimizations such as view-dependent culling or lazy updating, we achieve real-time frame rates in the range 40-150fps while handling dynamic viewing, lighting, and reflectance and evaluating the shading per-pixel.

Figure 8 compares per-vertex and per-pixel rendering results. Our result in the (c) column remains consistent even for quite sparse tessellation densities. The per-vertex shading result, in the (b) column, yields significant blurring and triangulation artifacts even with a fairly dense tessellation. These are especially visible in high-frequency shadows and highlights, as shown in the blue boxed inset.

Figure 9 isolates error between the SSDF representation itself and its compression. PCA-based compression in (d,e) produces a

Scene	BRDF Type	Texture Res.	BRDF Size	# Vert.	# E.L.	SSDF Samp.	SSDF Comp.	SSDF Size	Env. FPS	Pt. FPS
Teapot: Fig.1a, Fig.11ab	Cook-Torrance, (iso, 1 SG)	1024×1024	7.2MB	17k	8	20 min.	30D (25 min.)	2MB	171	250
Dragon: Fig.1b, Fig.11cd	Ward (iso, 1 SG)	512×512	1.8MB	37k	8	50 min.	30D (35 min.)	4.4MB	165	231
Dish+Balls: Fig.1c, Fig.12abc	Ashikhmin-Shirley (aniso, 7 SGs)	512×1024	4.1MB	28.5k	6	45 min.	20D (30 min.)	2.3MB	55	30
Dish+Cloth: Fig.1d, Fig.12efg	measured card (iso, 2 SGs)	512×512	4.2MB	20k	6	25 min.	10D (20 min.)	0.8MB		
	measured satin (aniso, 5 SGs)	850×850	22.4MB						48	35
	measured velvet (aniso, 2 SGs)	850×850	9.4MB						168	145

Table 1: Experimental performance. "# E.L." is the number of SG lobes fit to environmental light for rendering the diffuse component. "SSDF Samp." is the total time for sampling the visibility function at each vertex, converting to an SSDF, and down-sampling. "SSDF Comp." lists the reduced dimension and the total computation time (in parentheses) for SSDF compression using PCA. The rightmost two columns show rendering performance with environmental lighting, "Env. FPS", and two point light sources, "Pt. FPS".

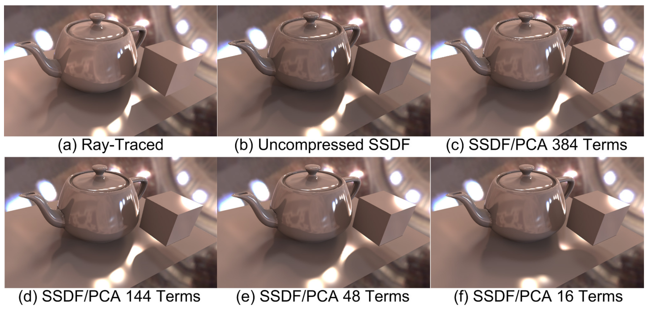

Figure 9: PCA compression of SSDFs.

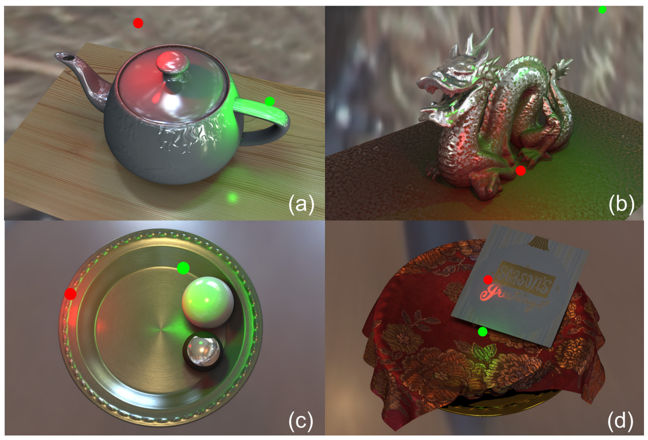

Figure 10: Results with local point and distant environmental lights.

reasonable result even with a small number of terms. Without compression, our result (b) matches the ray-traced ground truth with minimal visual difference.

Figure 10 shows rendering results illuminated with both nearby point light sources and distant environmental light. Our rendering algorithm supports dynamic local light sources; no additional precomputation is needed. Note the colored shadows and reflections, and high-frequency shadow edges.

Figure 11 shows results for isotropic parametric BRDF models. Realistic reflectance is generated as well as detailed spatial variation in the engraved mark on the teapot and rust pattern on the dragon. Bump maps can be adjusted at runtime as shown in (b). BRDF parameters can also be changed on the fly as shown in (d). Notice the all-frequency shadows on the floor (a,c) and on the neck and mouth of the dragon (c,d). Refer to the video for animations of bump maps and rust.

Figure 12abc shows results for anisotropic parametric BRDF models. Fan-shaped highlights are convincingly captured from the circular brush marks on the metal platter. Shadows in both diffuse and specular reflection are also reproduced. Note especially the deformed shadow of the white ball on the platter in (c), due to anisotropic reflection. User-specified local tangent rotation via interactive painting is also demonstrated in (b,c).

Figure 12d-h shows measured and tabulated SVBRDFs from real materials. The cloth is red satin (e,f,g) and velvet (d,h); fine features in the needlework and anisotropic reflectance are well-preserved. The greeting card is isotropic with spatially varying specularity, and changes its appearance realistically as the viewing/lighting conditions are animated. We also support real-time editing of SVBRDFs as shown in (f,g). In this example, we adjusted the shininess by changing the sharpness parameter of the fitted lobes to simulate a wetting effect within the painted region. Refer to the video and supplement for further results.

9 Conclusion

Illumination effects from highly specular, spatially varying, and dynamic materials under natural lighting add realism to synthetic imagery but have so far been too expensive for real-time rendering. We introduce two new, nonlinear representations to solve this problem: SG mixtures for microfacet-based reflectance and SSDFs for visibility. Fitting a small number of SG lobes to the microfacet NDF represents both parametric and measured BRDFs, including highly specular ones, compactly and accurately. It also allows fast rotation, warping, and products that speed up rendering. Compressed SSDFs compactly represent spatially-varying visibility and provide ghost-free, per-pixel interpolation from a sparse, per-vertex set of samples.

Our approach is subject to a number of limitations. Precomputed visibility restricts us to static scenes. However, our method can be combined with shadow fields [Zhou et al. 2005], local deformation [Sloan et al. 2005], or skinned shadowing proxies [Ren et al. 2006] to handle dynamic geometry. We consider only direct shadowing effects; inter-reflection (both specular and diffuse) is ignored. Since SGs are isotropic, our model requires many lobes to represent highly anisotropic BRDFs. Independent fitting of multi-lobe SG models for measured anisotropic reflectance requires nearest-neighbor filtering of the encoded texture. We are interested in addressing these drawbacks in future work, as well as investigating more sophisticated methods for compressing SSDFs.

References

ARVO, J., TORRANCE, K., AND SMITS, B. 1994. A framework for the analysis of error in global illumination algorithms. In *Proceedings of SIGGRAPH 1994*, ACM, 75–84.

ASHIKHMIN, M., AND SHIRLEY, P. 2000. An anisotropic Phong BRDF model. *Journal of Graphics Tools 5*, 2, 25–32.

ASHIKMIN, M., PREMOŽE, S., AND SHIRLEY, P. 2000. A microfacet-based BRDF generator. In *Proceedings of SIGGRAPH 2000*, ACM, 65–74.

BEN-ARTZI, A., OVERBECK, R., AND RAMAMOORTHI, R. 2006. Real-time BRDF editing in complex lighting. *ACM Transactions on Graphics 25*, 3, 945–954.

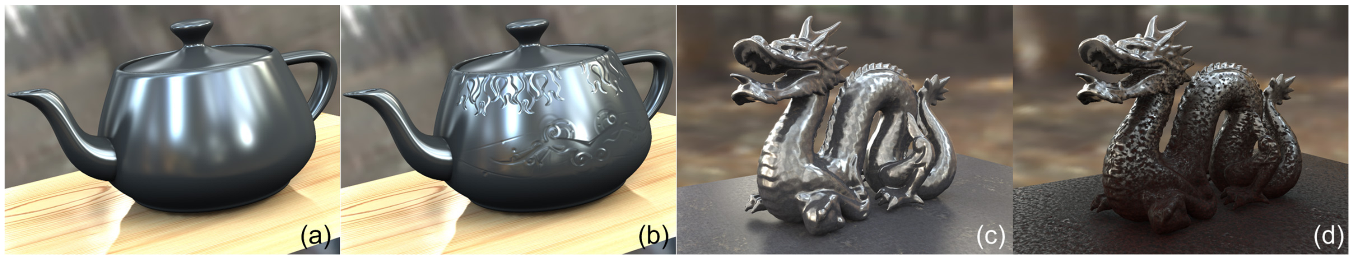

Figure 11: Rendering results with isotropic BRDF models. (a/b) Metal teapot using Cook-Torrance, without/with bump map. (c/d) Iron dragon using Ward, new/rusted.

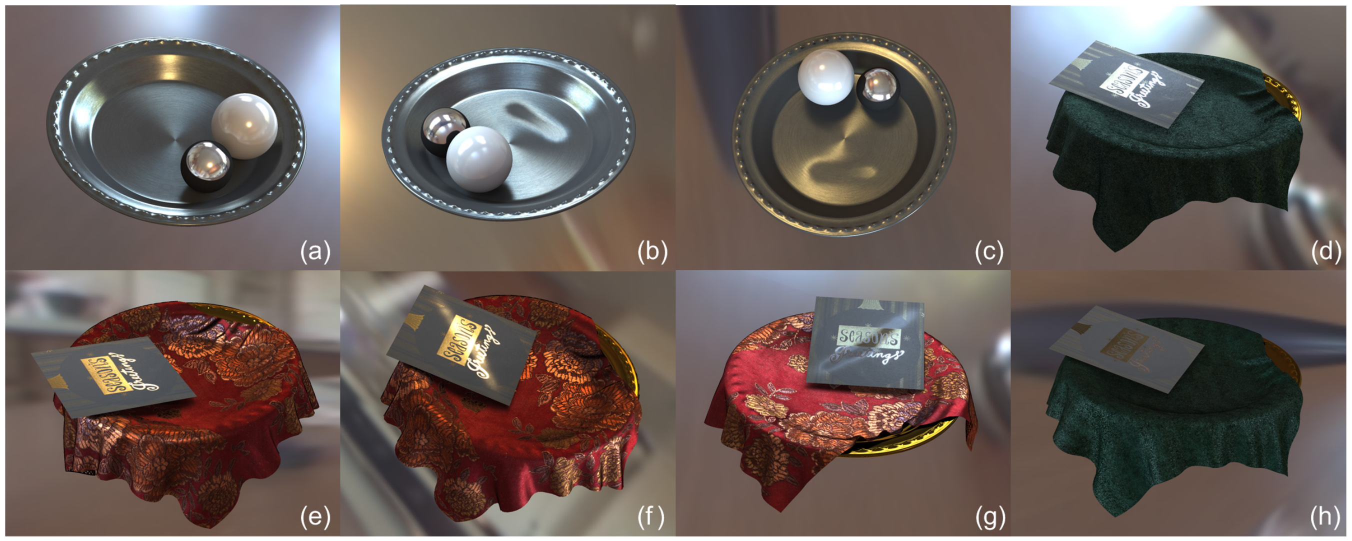

Figure 12: Rendering results with anisotropic BRDF models. (a) Dishes and balls using Ashikhmin-Shirley model. (b,c) Painted tangent rotation. (d-h) Measured cloth (5-lobe SG) and greeting card (2-lobe SG). (f,g) Painted SG warp suggesting wet paint.

BEN-ARTZI, A., EGAN, K., RAMAMOORTHI, R., AND DURAND, F. 2008. A precomputed polynomial representation for interactive BRDF editing with global illumination. *ACM Transactions on Graphics 27*, 2, 13:1–13:13.

BLINN, J. F. 1977. Models of light reflection for computer synthesized pictures. In *Computer Graphics (Proceedings of SIGGRAPH 77)*, ACM, vol. 11, 192–198.

CHESLACK-POSTAVA, E., WANG, R., AKERLUND, O., AND PELLACINI, F. 2008. Fast, realistic lighting and material design using nonlinear cut approximation. *ACM Transactions on Graphics 27*, 5, 128:1–128:10.

COOK, R. L., AND TORRANCE, K. E. 1981. A reflectance model for computer graphics. In *Computer Graphics (Proceedings of SIGGRAPH 81)*, ACM, vol. 1, 307–316.

GREEN, P., KAUTZ, J., MATUSIK, W., AND DURAND, F. 2006. View-dependent precomputed light transport using nonlinear gaussian function approximations. In *I3D '06: Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, ACM, 7–14.

GREEN, P., KAUTZ, J., AND DURAND, F. 2007. Efficient reflectance and visibility approximations for environment map rendering. *Computer Graphics Forum (Proc. EUROGRAPHICS) 26*, 3, 495–502.

HAN, C., SUN, B., RAMAMOORTHI, R., AND GRINSPUN, E. 2007. Frequency domain normal map filtering. *ACM Transactions on Graphics 26*, 3, 28:1–28:11.

KAUTZ, J., VÁZQUEZ, P.-P., HEIDRICH, W., AND SEIDEL, H.-P. 2000. Unified approach to prefiltered environment maps. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, Springer-Verlag, London, UK, 185–196.

KŘIVÁNEK, J., AND COLBERT, M. 2008. Real-time shading with filtered importance sampling. *Computer Graphics Forum (Proc. of Eurographics Symposium on Rendering) 27*, 4.

LAWRENCE, J., BEN-ARTZI, A., DECORO, C., MATUSIK, W., PFISTER, H., RAMAMOORTHI, R., AND RUSINKIEWICZ, S. 2006. Inverse shade trees for non-parametric material representation and editing. *ACM Transactions on Graphics 25*, 3, 735–745.

LIU, X., SLOAN, P. P., SHUM, H. Y., AND SNYDER, J. 2004. All-frequency precomputed radiance transfer for glossy objects. In *Proceedings of the Eurographics Symposium on Rendering*, Eurographics Association, 337–344.

MAHAJAN, D., TSENG, Y.-T., AND RAMAMOORTHI, R. 2008. An analysis of the in-out BRDF factorization for view-dependent relighting. In *Eurographics Symposium on Rendering*, vol. 27.

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

MCALLISTER, D. K., LASTRA, A., AND HEIDRICH, W. 2002. Efficient rendering of spatial bi-directional reflectance distribution functions. In *HWWS '02: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, Eurographics Association, Aire-la-Ville, Switzerland, 79–88.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2003. All-frequency shadows using non-linear wavelet lighting approximation. *ACM Transactions on Graphics 22*, 3, 376–381.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2004. Triple product wavelet integrals for all-frequency relighting. *ACM Transactions on Graphics 23*, 3, 477–487.

NGAN, A., DURAND, F., AND MATUSIK, W. 2005. Experimental analysis of BRDF models. In *Rendering Techniques 2005: 16th Eurographics Workshop on Rendering*, 117–126.

NOCEDAL, J., AND WRIGHT, S. J. 1999. Numerical optimization. *Springer Series in Operations Research, Springer-Verlag*.

REN, Z., WANG, R., SNYDER, J., ZHOU, K., LIU, X., SUN, B., SLOAN, P.-P., BAO, H., PENG, Q., AND GUO, B. 2006. Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. *ACM Transactions on Graphics 25*, 3, 977–986.

SHIRLEY, P., AND CHIU, K. 1997. A low distortion map between disk and square. *J. Graph. Tools 2*, 3, 45–52.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Proceedings of SIGGRAPH 2002*, ACM, 527–536.

SLOAN, P.-P., LIU, X., SHUM, H.-Y., AND SNYDER, J. 2003. Bi-scale radiance transfer. *ACM Transactions on Graphics 22*, 3 (July), 370–375.

SLOAN, P.-P., LUNA, B., AND SNYDER, J. 2005. Local, deformable precomputed radiance transfer. *ACM Transactions on Graphics 24*, 3, 1216–1224.

SUN, X., ZHOU, K., CHEN, Y., LIN, S., SHI, J., AND GUO, B. 2007. Interactive relighting with dynamic BRDFs. *ACM Transactions on Graphics 26*, 3, 27:1–27:10.

TORRANCE, K. E., AND SPARROW, E. M. 1967. Theory for off-specular reflection from roughened surfaces. In *Journal of the Oprical Society of America*, vol. 57.

TSAI, Y.-T., AND SHIH, Z.-C. 2006. All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Transactions on Graphics 25*, 3, 967–976.

TSAI, Y.-T., CHANG, C.-C., JIANG, Q.-Z., AND WENG, S.-C. 2008. Importance sampling of products from illumination and BRDF using spherical radial basis functions. *Vis. Comput. 24*, 7, 817–826.

WALTER, B., FERNANDEZ, S., ARBREE, A., BALA, K., DONIKIAN, M., AND GREENBERG, D. P. 2005. Lightcuts: a scalable approach to illumination. *ACM Transactions on Graphics 24*, 3, 1098–1107.

WALTER, B., ARBREE, A., BALA, K., AND GREENBERG, D. P. 2006. Multidimensional lightcuts. *ACM Transactions on Graphics 25*, 3, 1081–1088.

WANG, R., TRAN, J., AND LUEBKE, D. 2004. All-frequency relighting of non-diffuse objects using separable BRDF approximation. In *Rendering Techniques*, Eurographics Association, 345–354.

WANG, R., TRAN, J., AND LUEBKE, D. 2006. All-frequency relighting of glossy objects. *ACM Transactions on Graphics 25*, 2, 293–318.

WANG, J., ZHAO, S., TONG, X., SNYDER, J., AND GUO, B. 2008. Modeling anisotropic surface reflectance with example-based microfacet synthesis. *ACM Transactions on Graphics 27*, 3, 41:1–41:9.

WARD, G. J. 1992. Measuring and modeling anisotropic reflection. In *Computer Graphics (Proceedings of ACM SIGGRAPH 92)*, ACM, vol. 26, 265–272.

XU, K., JIA, Y.-T., FU, H., HU, S., AND TAI, C.-L. 2008. Spherical piecewise constant basis functions for all-frequency precomputed radiance transfer. *IEEE Transactions on Visualization and Computer Graphics 14*, 2, 454–467.

ZHOU, K., HU, Y., LIN, S., GUO, B., AND SHUM, H.-Y. 2005. Precomputed shadow fields for dynamic scenes. *ACM Transactions on Graphics 24*, 3, 1196–1201.

ZHU, C., BYRD, R. H., LU, P., AND NOCEDAL, J. 1997. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software 23*, 4, 550–560.

Appendix

Vector Product of Spherical Gaussians in Eq. (5)

The vector product of a pair of SG lobes is given by

$$\begin{aligned} &G(\mathbf{v};\mathbf{p}_1,\lambda_1,\mu_1)\,G(\mathbf{v};\mathbf{p}_2,\lambda_2,\mu_2) \\ &\qquad = \mu_1\,\mu_2\,\exp\left((\mathbf{v}\cdot\mathbf{p}_1-1)\,\lambda_1+(\mathbf{v}\cdot\mathbf{p}_2-1)\,\lambda_2\right) \\ &\qquad = \mu_1\,\mu_2\,\exp\left(\mathbf{v}\cdot(\lambda_1\mathbf{p}_1+\lambda_2\mathbf{p}_2)-(\lambda_1+\lambda_2)\right) \\ &\qquad = \mu_1\,\mu_2\,\exp\left(\left(\mathbf{v}\cdot\frac{\lambda_1\mathbf{p}_1+\lambda_2\mathbf{p}_2}{\lambda_1+\lambda_2}-1\right)(\lambda_1+\lambda_2)\right). \end{aligned}$$

Letting $\mathbf{p}_m = (\lambda_1\mathbf{p}_1+\lambda_2\mathbf{p}_2)/(\lambda_1+\lambda_2)$ and $\lambda_m = \lambda_1+\lambda_2$, then

$$G(\mathbf{v};\mathbf{p}_1,\lambda_1,\mu_1)\,G(\mathbf{v};\mathbf{p}_2,\lambda_2,\mu_2) = \mu_1\,\mu_2\,e^{(\mathbf{v}\cdot\mathbf{p}_m-1)\lambda_m}. \tag{44}$$

Equation (44) is almost an SG except that the vector $\mathbf{p}_m$ is not normalized:

$$\begin{aligned} &G(\mathbf{v};\mathbf{p}_1,\lambda_1,\mu_1)\,G(\mathbf{v};\mathbf{p}_2,\lambda_2,\mu_2) \\ &\quad = \mu_1\,\mu_2\,\exp\left(\left(\mathbf{v}\cdot\frac{\mathbf{p}_m}{\|\mathbf{p}_m\|}\right)\lambda_m\|\mathbf{p}_m\|-\lambda_m\right) \\ &\quad = \mu_1\,\mu_2\,\exp\left(\left(\mathbf{v}\cdot\frac{\mathbf{p}_m}{\|\mathbf{p}_m\|}-1\right)\lambda_m\|\mathbf{p}_m\|\right)\exp\left(\lambda_m\|\mathbf{p}_m\|-\lambda_m\right) \\ &\quad = G\left(\mathbf{v};\,\frac{\mathbf{p}_m}{\|\mathbf{p}_m\|},\,\lambda_m\|\mathbf{p}_m\|,\,\mu_1\,\mu_2\,e^{(\|\mathbf{p}_m\|-1)\lambda_m}\right). \end{aligned} \tag{45}$$

Analytic Approximation for f_h in Eq. (30)

We define the normalized function

$$\hat{f}_h(\theta_d,\lambda) = \frac{f_h(\theta_d,\lambda)}{f_h(\frac{\pi}{2},\lambda)}, \tag{46}$$

where $f_h(\frac{\pi}{2},\lambda) = \frac{2\pi}{\lambda}(1-e^{-\lambda})$. We then approximate $\hat{f}_h(\theta_d,\lambda)\in(0,1)$ as a sigmoid function σ of the following form:

$$\hat{f}_h(\theta_d,\lambda) \approx \sigma(\theta_d,k_\lambda) = \frac{a}{1+k_\lambda e^{\theta_d}} + \frac{1-a}{2}. \tag{47}$$

Since the unscaled (a=1) sigmoid function never reaches 0 or 1, we scale it slightly by setting $a = 1.05$. Then we represent k_λ as a low-order polynomial in λ:

$$k_\lambda \approx f_k(\lambda) = 0.204\lambda^3 - 0.892\lambda^2 + 2.995\lambda + 0.067. \tag{48}$$

These coefficients were derived using a least-squares fit to a densely sampled tabulation of $\hat{f}_h$. The final form of our approximation is

$$f_h(\theta_d,\lambda) \approx \chi\left(\theta_d, f_k(\lambda)\right) f_h\left(\frac{\pi}{2},\lambda\right). \tag{49}$$

The integrated squared fitting error is

$$\int_{-\frac{\pi}{2}}^{+\frac{\pi}{2}}\int_{1}^{10^5}\left(\hat{f}_h(\theta_d,\lambda)-\chi\left(\theta_d; f_k(\lambda)\right)\right)^2 \mathrm{d}\lambda\,\mathrm{d}\theta_d = 5.5\times10^{-4}.$$