HAL
open science

Linear Efficient Antialiased Displacement and Reflectance Mapping

Jonathan Dupuy, Eric Heitz, Jean-Claude Iehl, Pierre Poulin, Fabrice Neyret, Victor Ostromoukhov

To cite this version:

Jonathan Dupuy, Eric Heitz, Jean-Claude Iehl, Pierre Poulin, Fabrice Neyret, et al.. Linear Efficient Antialiased Displacement and Reflectance Mapping. ACM Transactions on Graphics, 2013, Proceedings of Siggraph Asia 2013, 32 (6), pp.Article No. 211. 10.1145/2508363.2508422 . hal-00858220v3

HAL Id: hal-00858220

https://inria.hal.science/hal-00858220v3

Submitted on 24 Jan 2014 (v3), last revised 31 Mar 2014 (v4)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

To appear in ACM TOG 32(6).

Linear Efficient Antialiased Displacement and Reflectance Mapping

Jonathan Dupuy[1,3] † Eric Heitz[2,1] † Jean-Claude Iehl[3] Pierre Poulin[1] Fabrice Neyret[2] Victor Ostromoukhov[3] *
[1]LIGUM, Dept. I.R.O., Université de Montréal [2]INRIA/LJK [3]LIRIS, Université Lyon 1
†Jonathan Dupuy and Eric Heitz are joint first authors

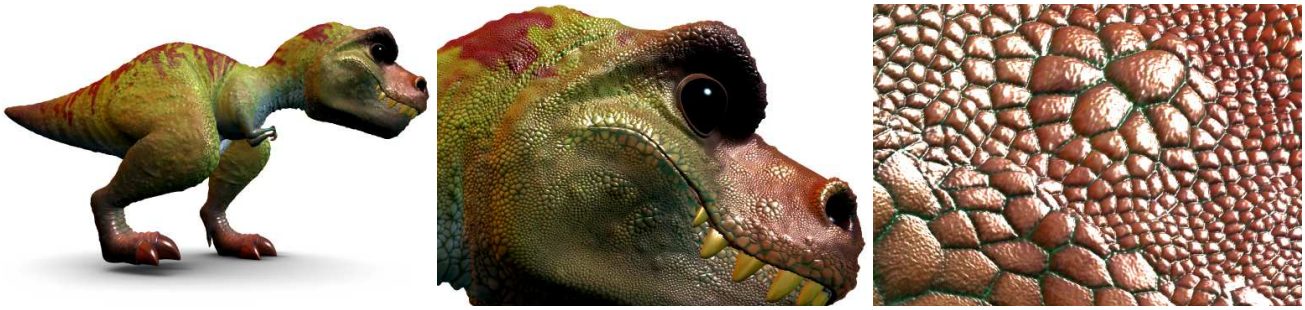

Figure 1: *A high-quality animated production model (Ptex T-rex model © Walt Disney Animation Studios.) rendered in real time under directional and environment lighting using LEADR mapping on an NVidia GTX 480 GPU. The surface appearance is preserved at all scales, using a single shading sample per pixel. Combined with adaptive GPU tessellation, our method provides the fastest, seamless, and antialiased progressive representation for displaced surfaces.*

Abstract

We present Linear Efficient Antialiased Displacement and Reflectance (LEADR) mapping, a reflectance filtering technique for displacement mapped surfaces. Similarly to LEAN mapping, it employs two mipmapped texture maps, which store the first two moments of the displacement gradients. During rendering, the projection of this data over a pixel is used to compute a noncentered anisotropic Beckmann distribution using only simple, linear filtering operations. The distribution is then injected in a new, physically based, rough surface microfacet BRDF model, that includes masking and shadowing effects for both diffuse and specular reflection under directional, point, and environment lighting. Furthermore, our method is compatible with animation and deformation, making it extremely general and flexible. Combined with an adaptive meshing scheme, LEADR mapping provides the very first seamless and hardware-accelerated multi-resolution representation for surfaces. In order to demonstrate its effectiveness, we render highly detailed production models in real time on a commodity GPU, with quality matching supersampled ground-truth images.

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

Keywords: BRDF, microfacet, filtering, LEAN mapping, GPU

* {jdupuy | jciehl | victor.ostromoukhov}@liris.cnrs.fr
{eric.heitz | fabrice.neyret}@inria.fr
poulin@iro.umontreal.ca

1 Introduction

Rendering applications such as video games commonly employ bump or normal textures (henceforth interchangeably referred to as normal textures) to enhance surface appearance (e.g., [Kilgard 2000]). These textures perturb or modify the normal of a simple underlying surface to emulate geometric variations through shading perturbations. Similarly to albedo textures, normal textures must be filtered for antialiasing purposes. But since shading with linearly filtered normals does not result in proper reflectance filtering, these textures cannot exclusively rely on simple methods such as mipmapping. Recently introduced by Olano and Baker [2010], LEAN mapping is an elegant solution to this problem, that has found widespread adoption because of its effectiveness, efficiency, and simplicity [Baker 2011].

Normal mapping is an inherited paradigm from the 1980's. At that time, geometric models were coarse because of computing capabilities and memory constraints. Textures cheaply enhanced visual details without increasing geometric complexity. The discrepancy between the resolutions of geometry and texture was the key assumption for texture filtering: within the same large and flat triangle, visibility, curvature, and orientation were considered constant. Consequently, filtering in texture space or in geometry space could reasonably be considered equivalent. This assumption does not hold anymore because mesh and texture resolutions can be matched with negligible overhead on modern GPUs.

Filtering appearance of small-scale geometry is thus a critical emerging problem. Indeed, small-scale geometry produces view- and light-dependent effects that include *masking*, *shadowing*, and *projection weighting* (i.e., the cosine term). Filtering this small-scale geometry while neglecting these visual effects violates energy conservation and can result in objectionable aliasing, popping artifacts, and inconsistent appearances throughout scales. Methods for filtering normal maps do not account for these effects, which is why filtering reflectance from normal mapping is not the same problem as filtering reflectance of small-scale geometry [Han et al. 2007; Bruneton and Neyret 2012] (see Figure 2).

We propose a solution to this problem in the important case where the small-scale geometry is generated by *displacement mapping*.

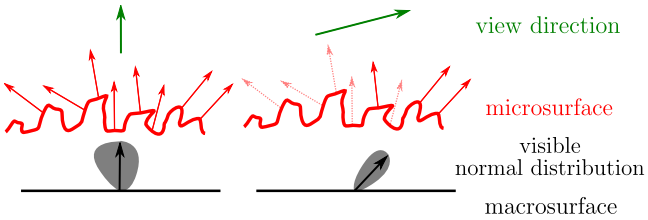

Figure 2: *On real surfaces the distribution of visible normals depends on the view direction because of masking and projection weighting.*

We introduce Linear Efficient Antialiased Displacement and Reflectance (LEADR) mapping, a technique motivated by LEAN mapping but adapted to reflectance of displacement maps. We use the same lightweight memory representation, but with a physically based BRDF that correctly accounts for view- and light-dependent effects under directional, point, and environment lighting. Our model also leverages commodity GPU features, and produces high-quality images matching supersampled ground-truth images. When coupled with an adaptive meshing scheme, LEADR mapping provides a seamless, multi-resolution, and hardware-accelerated surface representation. Our contributions can be summarized as follows.

- We model physically based microfacet BRDFs with *noncentered* Beckmann microfacet distributions.
- We provide anisotropic BRDF equations for specular and diffuse surfaces of arbitrary roughness.
- We derive an analytical solution for the noncentered anisotropic specular microfacet BRDF with a directional/point light.
- We propose a simple, controllable, noise- and artifact-free sampling scheme for our microfacet model under environment lighting.
- We show how surface height scaling, and tangent stretching and shearing can be integrated efficiently in our reflectance model.

Overview

In order to give a general understanding of the workings of our method, and before diving into the mathematical derivations for our solutions, we provide in Figure 3 an overview of its GPU pipeline.

A series of surface statistics are precomputed from a displacement map, and stored in mipmapped textures. In the vertex processing stages, the geometry is adaptively tessellated and displaced for best rasterization performance. The displacements that are too small to be represented by the final geometry are accounted for in our reflectance model during fragment processing, thus guaranteeing consistent and seamless surface shape and appearance at any scale. First, a noncentered Beckmann distribution of the slopes projected onto the footprint of the pixel is computed. This process leverages modern GPU filtering capabilities and only consists of texture fetch operations. The resulting PDF and incident lighting are then combined to compute the diffuse and specular reflectance from the surface using our two new BRDF models (Sections 4 and 5).

Note that a number of additional mathematical derivations, properties, explanations, and pseudocodes are provided in the associated supplemental document.

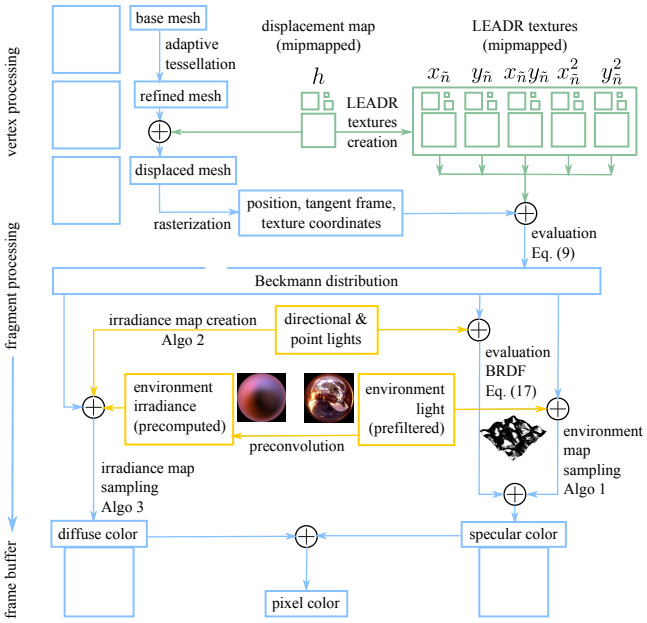

Figure 3: *The GPU pipeline for our method. The following data is given or precomputed once per scene: LEADR textures and light sources (points and environments). The processing pipeline is divided in two stages. First, the vertex processing computes all the data projected into each pixel. Then, the fragment processing calculates the noncentered Beckmann distribution in the pixel to be combined with the incident lighting in order to compute the diffuse and specular shading contributions to the pixel.*

2 Related Work

Convolution-based normal map filtering methods [Fournier 1992; Toksvig 2005; Han et al. 2007] use the fact that, at any scale, the BRDF is the convolution of a base BRDF and a Normal Distribution Function. Because masking, shadowing, and projection weighting are nonlinear functions of the view, light, and normal directions, incorporating these effects into the convolution is difficult. Tan et al. [2005; 2008] use several Gaussian lobes with a masking-shadowing term, but omit the important view-dependent projection weighting effect and do not normalize their BRDF. Heitz et al. [2013] derive an analytical approximation to evaluate view-dependent normal distributions on the fly, in order to filter albedo textures correlated with visibility on displaced surfaces. Neither the BRDF nor reflectance are accounted for in their model. Other methods precompute the complete BRDF at any location and for any scale, and store it in a Bidirectional Texture Function [Cabral et al. 1987; Becker and Max 1993; Ma et al. 2005; Wu et al. 2009]. These methods are capable of capturing view-dependent effects, and could theoretically work with displacement maps; however, the dissuasive memory requirements to store 6D spatially varying BRDFs strongly reduce their practicality.

Our approach concerns three main areas of previous work: LEAN mapping, physically based BRDFs, and filtered importance sampling. We review their respective related work in the following paragraphs.

LEAN Mapping. Olano and Baker [2010] introduce a lightweight representation capable of leveraging the tangent-plane parameterization to express the moments of the normal distribution in the same reference frame. Their formulation allows for linear filtering of the

data in a manner that properly captures filtered reflectance. Furthermore, their representation supports anisotropy and multilayer superposition, making it overall an excellent tool for real-time rendering. The main limitation of LEAN mapping in the context of our work comes from the non-physically based BDRF employed by the model: It lacks proper normalization, as the Jacobian of the reflection operation is missing, and does not incorporate the important masking, shadowing, and projection weighting effects. Moreover, it is only designed for specular microfacets under point and directional lighting.

Physically based BRDFs. Microfacet theory has been used extensively by the graphics community to derive physically based BRDF models [Beckmann and Spizzichino 1963; Cook and Torrance 1982; Oren and Nayar 1994; Walter et al. 2007]. It is only recently though that both production and real-time rendering communities have begun leveraging their use [McAuley et al. 2012]. Despite the fact that microfacet BRDF derivations bare many similarities with the problem of filtering reflectance from displaced surfaces, they cannot be used directly. Indeed, BRDF formulations do not account for multiple scales, and more importantly assume centered microfacet distributions. This implies a constant average microsurface normal pointing in the "up" direction of the local frame of parameterization, and prevents their use with LEAN mapping, where the average normal can deviate significantly from this direction. As another illustration, Bruneton et al. [2010] use a physically based BRDF model [Ross et al. 2005] to compute the reflectance of procedural ocean displacements at any scale. Since their formulation does not account for noncentered microfacet normal distributions, they are forced to use an approximation, which arguably works for small angular deviations, but fails when the average microfacet normal deviates significantly from the vertical.

Filtered Importance Sampling. For lighting from an environment map, the BRDF must be integrated over all incident lighting directions. Monte Carlo integration is typically used for this problem but produces noisy results and is not adequate for real-time rendering. Colbert and Křivánek [2007; 2008] propose a sampling scheme that leverages filtered environment map representations. Instead of sampling directions from the unfiltered environment map, they use samples from the filtered version of the environment map. For a fixed budget of N samples, the algorithm always produces smooth and noise-free results, while exhibiting convergence as N increases. Such behavior is well suited for real-time rendering purposes, where artifact-free tradeoffs between quality and performance are important. We extend this idea to our more complex BRDF models.

3 Preliminaries

In this section, we show how displacement map filtering can be directly transposed into the problem of deriving a microfacet reflection model. This key observation allows us to use a derivative of LEAN mapping to solve our new shading equations efficiently in the next sections.

Let $\mathcal{P}$ be a flat geometric patch that projects onto a pixel, and whose area is normalized under the measure of a low-pass filter $k_{\mathcal{P}}$ satisfying $\int_{\mathcal{P}} k_{\mathcal{P}}(p)\,dp = 1$. We refer to the normal of $\mathcal{P}$ as the *macronormal* ω_g, which verifies $\omega_g = (0,0,1)^t$ in its tangent space. At a smaller scale, $\mathcal{P}$ is perturbed by displacements. This process generates micronormals $\omega_n(p)$ indexed by locations $p \in \mathcal{P}$. These micronormals average to $\omega_{\bar{n}}$, referred to as the *mesonormal* of $\mathcal{P}$. Note that contrary to previous microfacet models, we do not assume that $\omega_g = \omega_{\bar{n}}$. This configuration is illustrated in Figure 4. For any

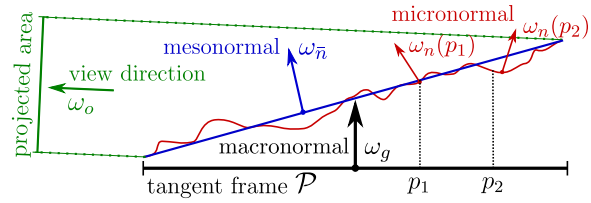

Figure 4: *Macro-, meso-, and micronormals. The displacement is applied to points p_i on patch $\mathcal{P}$, resulting in the process into different sets of normals.*

observation direction ω_o, the projected area of the plane defined by the mesonormal is

$$\frac{\omega_{\bar{n}} \cdot \omega_o}{\omega_{\bar{n}} \cdot \omega_g} = \int_{\mathcal{P}} \frac{\omega_n(p) \cdot \omega_o}{\omega_n(p) \cdot \omega_g}\, k_{\mathcal{P}}(p)\, dp. \tag{1}$$

Here, the dot products $\omega_{\bar{n}} \cdot \omega_o$ and $\omega_n(p) \cdot \omega_o$ are the projection weights due to the observation direction. The two other dot products $\omega_{\bar{n}} \cdot \omega_g$ and $\omega_n(p) \cdot \omega_g$ are due to parameterization Jacobians, and respectively measure the areas of the meso- and microsurfaces. Note that if $\omega_g = \omega_{\bar{n}}$, then $\omega_{\bar{n}} \cdot \omega_g = 1$.

In order to shade $\mathcal{P}$, we define the intensity I of the pixel it covers as the integration of a filtering operation over all incident lighting directions Ω:

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \int_{\Omega} L(\omega_i) \int_{\mathcal{P}} R(p, \omega_o, \omega_i)\, k_{\mathcal{P}}(p)\, dp\, d\omega_i. \tag{2}$$

The term $\int_{\mathcal{P}} R(p, \omega_o, \omega_i)\, k_{\mathcal{P}}(p)\, dp$ is the "filtering integral" and involves a local reflectance function $R(p, \omega_o, \omega_i)$, assumed constant over $\mathcal{P}$. Additionally, $L(\omega_i)$ is the incident radiance from direction ω_i, and $\frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o}$ is a renormalization factor measuring projected area in direction ω_o. Under the geometric optics approximation, we define our local reflectance function as

$$\begin{aligned} R(p, \omega_o, \omega_i) = {} & V(p, \omega_i)\, V(p, \omega_o) \times \\ & \rho\left(\omega_n(p), \omega_o, \omega_i\right) \frac{\langle \omega_n(p), \omega_o \rangle\ \langle \omega_n(p), \omega_i \rangle}{\omega_n(p) \cdot \omega_g} \end{aligned} \tag{3}$$

where at location p, $V(p, \omega_o)$ (resp. $V(p, \omega_i)$) denotes the masking (resp. shadowing) Heaviside function, and $\rho(\omega_n(p), \omega_o, \omega_i)$ the surface's BRDF. The operator $\langle -, - \rangle$ gives the cosine of the angle formed between two vectors, clamped to zero if it is negative.

Similarly to PDFs, (low-pass) energy-conserving filters integrate to one over their domain. Therefore, the filtering integral in Equation (2) can be interpreted as the first moment of a function of random variable p, with PDF $k_{\mathcal{P}}(p)$[1]:

$$\int_{\mathcal{P}} R(p, \omega_o, \omega_i)\, k_{\mathcal{P}}(p)\, dp = \mathbb{E}[R(p, \omega_o, \omega_i)]. \tag{4}$$

This observation directly links our filtering problem with microfacet BRDF models, since they also integrate reflectance using the statistics of the surface. This allows us to follow an important assumption: Previous work has shown that treating the visibility functions $V(p, -)$ independently from the other parameters of R is an accurate approximation [Ross et al. 2005; Bruneton et al. 2010; Heitz and Neyret 2012]. Intuitively, this is because occlusions never occur at position p directly, as they are due to distant geometry,

[1] Some filters (such as $k_{\mathcal{P}}(p) = \mathrm{sinc}(p)$) produce negative weights and may not be considered as valid PDFs, but this does not impact the equations.

contrary the other parameters in R that depend only on the normal $\omega_n(p)$. Based on this observation, we can reasonably write

$$\mathbb{E}[R(p,\omega_o,\omega_i)] = \mathbb{E}[V(p,\omega_o)V(p,\omega_i)]\,\mathbb{E}[R_n(\omega_n,\omega_o,\omega_i)],$$

having

$$\begin{aligned}&\mathbb{E}[R_n(\omega_n,\omega_o,\omega_i)]\\ &= \int_\Omega \rho(\omega_n,\omega_o,\omega_i)\,\langle\omega_n,\omega_o\rangle\,\langle\omega_n,\omega_i\rangle\,D(\omega_n)\,d\omega_n. \qquad (5)\end{aligned}$$

Here, D is the *exact* normal distribution function over $\mathcal{P}$:

$$D(\omega) = \int_{\mathcal{P}} \frac{\delta_\omega(\omega_n(p))}{\omega_n(p)\cdot\omega_g}\,k_{\mathcal{P}}(p)\,dp, \qquad (6)$$

and satisfies $\int_\Omega(\omega_n\cdot\omega_g)\,D(\omega_n)\,d\omega_n = 1$. Thus, using Equations (2) and (4), we can derive our "microfacet equation":

$$\begin{aligned}I =& \frac{\omega_{\tilde{n}}\cdot\omega_g}{\omega_{\tilde{n}}\cdot\omega_o}\times\\ &\int_\Omega L(\omega_i)\,\mathbb{E}[V(p,\omega_o)V(p,\omega_i)]\,\mathbb{E}[R_n(\omega_n,\omega_o,\omega_i)]\,d\omega_i, \qquad (7)\end{aligned}$$

expressing the pixel intensity as an integral over all incident lighting directions of the masking-shadowing moment $\mathbb{E}[V(p,\omega_o)V(p,\omega_i)]$ and the reflectance moment $\mathbb{E}[R_n(\omega_n,\omega_o,\omega_i)]$. Compared to Equation (2), we only assume that the statistics of $V(p,-)$ and ω_n are uncorrelated.

4 Gaussian Surfaces

The distribution of normals as formulated in Equation (6) makes Equation (7) inadequate to be solved under real-time constraints. Therefore, we will assume that D takes the form of a Beckmann distribution at all scales. Given this assumption, we discuss the memory requirements of our method, derive an analytical formulation for $\mathbb{E}[V(p,\omega_o)V(p,\omega_i)]$, and instantiate Equation (7) for specular and diffuse microfacets. Our choice of distribution is further motivated in Section 6.

4.1 Properties

Elevation and slopes are usually the two parameters considered when studying the statistics of random rough surfaces [Bourlier et al. 2000]. Choosing such features to follow Gaussian distributions is especially interesting for our needs because they are flexible and lightweight, but also quite general. For instance, many classes of real-world surfaces, such as stochastic, fractal, or multi-layered surfaces, produce Gaussian statistics that can exhibit behaviors that range from perfectly smooth to very rough. Denoting surface slopes as $\tilde{n} = (x_{\tilde{n}}, y_{\tilde{n}})^t$, we write their 2D PDF as

$$P_{22}(\tilde{n}) = \frac{\exp\left(-\frac{1}{2}\,(\tilde{n}-\mathbb{E}[\tilde{n}])^t\,\Sigma^{-1}\,(\tilde{n}-\mathbb{E}[\tilde{n}])\right)}{2\pi\sqrt{|\Sigma|}}, \qquad (8)$$

where Σ is the covariance matrix

$$\Sigma = \begin{bmatrix}\sigma_x^2 & c_{xy}\\ c_{xy} & \sigma_y^2\end{bmatrix} \quad\text{with}\quad \begin{aligned}\sigma_x^2 &= \mathbb{E}[x_{\tilde{n}}^2]-\mathbb{E}^2[x_{\tilde{n}}]\\ \sigma_y^2 &= \mathbb{E}[y_{\tilde{n}}^2]-\mathbb{E}^2[y_{\tilde{n}}]\\ c_{xy} &= \mathbb{E}[x_{\tilde{n}}y_{\tilde{n}}]-\mathbb{E}[x_{\tilde{n}}]\,\mathbb{E}[y_{\tilde{n}}]\end{aligned} \qquad (9)$$

Now, recalling that slopes are mathematically linked to normals by the relations

$$x_{\tilde{n}} = -x_n/z_n \quad\text{and}\quad y_{\tilde{n}} = -y_n/z_n,$$

our normal distribution function D following an anisotropic Beckmann formulation can be expressed as

$$D(\omega_n) = \frac{P_{22}(\tilde{n})}{(\omega_n\cdot\omega_g)^4}. \qquad (10)$$

The factor $\frac{1}{(\omega_n\cdot\omega_g)^4}$ is due to the Jacobian $\left\|\frac{\partial\tilde{n}}{\partial\omega_n}\right\| = \frac{1}{(\omega_n\cdot\omega_g)^3}$ and the inverse projection $\frac{1}{(\omega_n\cdot\omega_g)}$ that normalizes the PDF. Since the moments in Equation (9) are linear, we store them in mipmapped textures. Linear (mipmapped) filtering can then be exploited to compute the adequate normal distribution at any scale [Olano and North 1997; Olano and Baker 2010]. Note also that we can retrieve the mesonormal $\omega_{\bar{n}}$ directly from this linear data:

$$\omega_{\bar{n}} = \frac{(-\mathbb{E}[x_{\tilde{n}}], -\mathbb{E}[y_{\tilde{n}}], 1)^t}{\sqrt{1+\mathbb{E}^2[x_{\tilde{n}}]+\mathbb{E}^2[y_{\tilde{n}}]}}. \qquad (11)$$

Thus, similarly to LEAN mapping, we store the terms $\mathbb{E}[x_{\tilde{n}}]$, $\mathbb{E}[y_{\tilde{n}}]$, $\mathbb{E}[x_{\tilde{n}}^2]$, $\mathbb{E}[y_{\tilde{n}}^2]$, and $\mathbb{E}[x_{\tilde{n}}y_{\tilde{n}}]$, i.e., five scalar values, in two mipmapped texture maps.

Since we use the same representation as LEAN mapping, our method inherits other important properties from it: the base surface roughness can be modeled by offsetting the parameters of the covariance matrix, the model is compatible with typical Blinn-Phong shading since the Beckmann distribution can be used to approximate it, and the linearity of the parameters allows for the combination of displacement layers. Another important property is its compatibility with animation, which is addressed next.

4.2 Macrosurface Animation

In this section, we study Gaussian microsurface statistics in the case where the macrosurface is subject to deformations.

Surface Stretching and Shearing. In practice, we store the moments of the slopes defined in texture space (u,v), denoted as $\mathbb{E}[u_{\tilde{n}}]$, $\mathbb{E}[v_{\tilde{n}}]$, $\mathbb{E}[u_{\tilde{n}}^2]$, $\mathbb{E}[v_{\tilde{n}}^2]$, and $\mathbb{E}[u_{\tilde{n}}v_{\tilde{n}}]$. They are linearly filtered at runtime and transformed into texture space slope moments using Equation (9). In order to solve Equation (7), this distribution must be expressed in the macrosurface tangent frame (x,y,ω_g). Since meshes are rarely perfectly parameterized, the texture parameterization $u(x,y), v(x,y)$ usually produces distortions. This phenomenon is further accentuated in the case of animation. If we denote as $h(u,v)$ the amplitude of the displacement map, the texture space slopes are then defined as the displacement gradient

$$(u_{\tilde{n}}, v_{\tilde{n}}) = \left(\frac{\partial h}{\partial u}, \frac{\partial h}{\partial v}\right).$$

Therefore, the slopes in world space can be defined by

$$(x_{\tilde{n}}, y_{\tilde{n}}) = \left(\frac{\partial h}{\partial x}, \frac{\partial h}{\partial y}\right) = \left(\frac{\partial h}{\partial u}\frac{\partial u}{\partial x} + \frac{\partial h}{\partial v}\frac{\partial v}{\partial x}, \frac{\partial h}{\partial u}\frac{\partial u}{\partial y} + \frac{\partial h}{\partial v}\frac{\partial v}{\partial y}\right).$$

If the distortions occurring on the macrosurface have low spatial variations, then we can reasonably assume that the mapping distortions represented by the terms $\frac{\partial u}{\partial x}$, $\frac{\partial v}{\partial x}$, $\frac{\partial u}{\partial y}$, and $\frac{\partial v}{\partial y}$ are locally constant. It follows that the appropriate slope distribution in world

space verifies

$$\begin{aligned}
\mathbb{E}[x_{\tilde{n}}] &= \frac{\partial u}{\partial x}\mathbb{E}[u_{\tilde{n}}]+\frac{\partial v}{\partial x}\mathbb{E}[v_{\tilde{n}}] \;\; ; \;\; \mathbb{E}[y_{\tilde{n}}] = \frac{\partial u}{\partial y}\mathbb{E}[u_{\tilde{n}}]+\frac{\partial v}{\partial y}\mathbb{E}[v_{\tilde{n}}] \\
\sigma_x^2 &= \left(\frac{\partial u}{\partial x}\right)^2 \sigma_u^2 + \left(\frac{\partial v}{\partial x}\right)^2 \sigma_v^2 + 2\,\frac{\partial u}{\partial x}\frac{\partial v}{\partial x} c_{uv} \\
\sigma_y^2 &= \left(\frac{\partial u}{\partial y}\right)^2 \sigma_u^2 + \left(\frac{\partial v}{\partial y}\right)^2 \sigma_v^2 + 2\,\frac{\partial u}{\partial y}\frac{\partial v}{\partial y} c_{uv} \\
c_{xy} &= \frac{\partial u}{\partial x}\frac{\partial u}{\partial y}\sigma_u^2 + \frac{\partial v}{\partial x}\frac{\partial v}{\partial y}\sigma_v^2 + \left(\frac{\partial u}{\partial x}\frac{\partial v}{\partial y} + \frac{\partial v}{\partial x}\frac{\partial u}{\partial y}\right) c_{uv}
\end{aligned} \tag{12}$$

Height Scaling. One may wish to apply a scaling factor η on displacements h to increase or reduce surface displacement. If η also exhibits low spatial variation across the surface, then it suffices to scale the first order moments $\mathbb{E}[x_{\tilde{n}}]$ and $\mathbb{E}[y_{\tilde{n}}]$ by η, and the second order moments $\mathbb{E}[x_{\tilde{n}}^2]$, $\mathbb{E}[y_{\tilde{n}}^2]$, and $\mathbb{E}[x_{\tilde{n}} y_{\tilde{n}}]$ by η^2.

Computing the moments of D in this way allows us to support animated geometry without having to update the LEADR mipmap hierarchy during animation. The distribution is adapted on the fly from the distortions, and adequately modulates the resulting appearance, as illustrated in Figure 5.

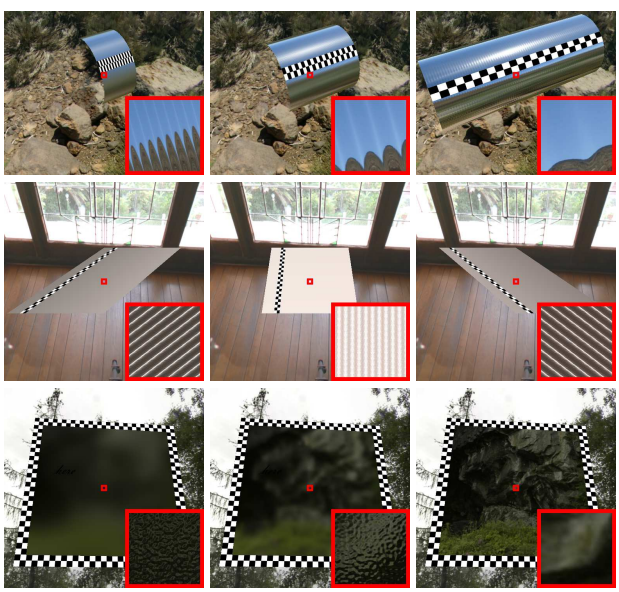

Figure 5: *Top row: Our representation allows for efficient rendering under surface stretching. Stretching the surface reduces the amplitude of the slopes. Because our representation is linear, this effect can be included at any scale. Note how the specular reflection becomes anisotropic under important surface compression. Middle row: A quad with diffuse microfacets is deformed by surface shearing. Shearing the surface changes the slope distribution and thus its appearance. As well as stretching, this effect can be accounted for at any scale. Bottom row: Height downscaling reduces the amplitude of the displacement map. At close view, the bumps disappear, and at a distance, the roughness decreases and the surface becomes specular, leading to a mirror-like reflection. All the insets show closer views of the actual displaced geometry.*

4.3 Masking-Shadowing

Another benefit of using Gaussian surfaces is the availability of an experimentally validated solution to the masking-shadowing moment [Smith 1967; Bourlier et al. 2000]:

$$\mathbb{E}[V(p,\omega_o)V(p,\omega_i)] = \frac{1}{1+\Lambda(\omega_o)+\Lambda(\omega_i)}. \tag{13}$$

Here, the Λ function from Smith [1967] is defined as follows for any direction $\omega = (\sin\theta\,\cos\phi, \sin\theta\,\sin\phi, \cos\theta)$:

$$\Lambda(\omega) = \tan\theta \int_{\cot\theta}^{+\infty} (x - \cot\theta)\, P_2(x)\, dx.$$

The PDF $P_2(x)$ is the slope density along a fixed incident direction ω. It is also Gaussian, hence $P_2(x) = \frac{1}{\sigma(\phi)\sqrt{2\pi}} \exp\left(-\frac{(x-\mu(\phi))^2}{2\,\sigma^2(\phi)}\right)$, with parameters

$$\mu(\phi) = \cos\phi\,\mathbb{E}[x_{\tilde{n}}] + \sin\phi\,\mathbb{E}[y_{\tilde{n}}], \tag{14}$$

$$\sigma^2(\phi) = \cos^2\phi\,\sigma_x^2 + \sin^2\phi\,\sigma_y^2 + 2\,\cos\phi\,\sin\phi\,c_{xy}. \tag{15}$$

Analytical solutions as well as approximations have been derived for Λ [Bourlier et al. 2000], but only under the assumption that the distribution is centered, i.e., $\mathbb{E}[x_{\tilde{n}}] = \mathbb{E}[y_{\tilde{n}}] = 0$. Fortunately, the formula can easily be extended to Gaussians with arbitrary mean by shifting its parameter by the mean slope along the incident direction ω. Using Equations (14) and (15), we generalize $\nu = \frac{\cot\theta - \mu(\phi)}{\sigma(\phi)\sqrt{2}}$ and combine it with the approximation given by Walter et al. [2007]:

$$\begin{aligned}
\Lambda(\omega) &= \frac{\exp\left(-\nu^2\right)}{2\nu\sqrt{\pi}} - \frac{\operatorname{erfc}(\nu)}{2} \\
&\approx \begin{cases} \frac{1.0-1.259\nu+0.396\nu^2}{3.535\nu+2.181\nu^2} & \text{if } \nu < 1.6 \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

Note that $\nu < 0$ may occur, implying that the incident direction ω is below the surface. This should not happen for the computation of $\Lambda(\omega_o)$ because the face would then be backface culled. However, this may happen for ω_i, and then $\Lambda(\omega_i) = +\infty$. In this case the masking-shadowing term is 0 because the surface is completely in shadow.

Now that we have expressions for D and $\mathbb{E}[V(p,\omega_o)V(p,\omega_i)]$, we turn to $\mathbb{E}[R_n(\omega_n,\omega_o,\omega_i)]$. This is the final step to getting a complete expression for Equation (7), and simply requires that we define the BRDF term ρ of the microgeometry.

4.4 Specular Microfacets

A (mirror) specular BRDF is defined as a function of the half-vector $\omega_h = \frac{\omega_o+\omega_i}{\|\omega_o+\omega_i\|}$ as follows

$$\begin{aligned}
\rho(\omega_n,\omega_o,\omega_i) &= \left\|\frac{\partial\omega_h}{\partial\omega_i}\right\| \frac{F(\omega_h,\omega_i)\,\delta_{\omega_h}(\omega_n)}{\langle\omega_h,\omega_i\rangle} \\
&= \frac{F(\omega_h,\omega_i)\,\delta_{\omega_h}(\omega_n)}{4\,\langle\omega_h,\omega_i\rangle^2},
\end{aligned}$$

where $\left\|\frac{\partial\omega_h}{\partial\omega_i}\right\| = \frac{1}{4\,|\omega_h\cdot\omega_i|}$ is the Jacobian of the reflection transformation [Walter et al. 2007], and F is the Fresnel term. After substitution in Equation (5) we get

$$\mathbb{E}[R_n(\omega_n,\omega_o,\omega_i)] = F(\omega_h,\omega_i)\,\frac{D(\omega_h)}{4},$$

thus leading to the following expression for Equation (7):

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \int_{\Omega_i} \frac{L(\omega_i)\, F(\omega_h, \omega_i)}{4\,[1 + \Lambda(\omega_o) + \Lambda(\omega_i)]}\, D(\omega_h)\, d\omega_i. \quad (16)$$

Equation (16) provides a complete shading formulation for rough specular surfaces. Its physical soundness can easily be validated since it integrates to one without the shadowing $\Lambda(\omega_i)$ and Fresnel F terms for any possible configuration $\{\omega_o, \mathbb{E}[x_{\tilde{n}}], \mathbb{E}[y_{\tilde{n}}], \sigma_x^2, \sigma_y^2, c_{xy}\}$:

$$\frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \int_{\Omega_i} \frac{1}{4\,[1 + \Lambda(\omega_o)]}\, D(\omega_h)\, d\omega_i = 1.$$

Point and Directional Lighting Under a point or directional light δ_{ω_i} emitting radiance L, Equation (16) takes an analytical form

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \frac{L\, F(\omega_h, \omega_i)}{4\,[1 + \Lambda(\omega_o) + \Lambda(\omega_i)]}\, D(\omega_h). \quad (17)$$

When the mesonormal is centered (i.e., $\omega_{\bar{n}} = \omega_g$), Equation (17) is equivalent to previous work [Ross et al. 2005; Bruneton et al. 2010]. Thus, our result can be interpreted as a generalization to the case of Gaussian distributions with arbitrary mean.

Environment Lighting For environment lighting, we reformulate the integral in Equation (16) in normal space Ω_n, rather than Ω_i. The Jacobian $\left\| \frac{\partial \omega_h}{\partial \omega_i} \right\|$ then cancels out, yielding

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \int_{\Omega_n} \frac{L(\omega_i)\, F(\omega_h, \omega_i)\, \langle \omega_n, \omega_o \rangle}{1 + \Lambda(\omega_o) + \Lambda(\omega_i)}\, D(\omega_n)\, d\omega_n.$$

Note that because of the change of variable, ω_i must be computed using the reflection operator

$$\omega_i = 2\,(\omega_n \cdot \omega_o)\, \omega_n - \omega_o. \quad (18)$$

With a change of variable $D(\omega_n)\, d\omega_n = \frac{P_{22}(\tilde{n})}{\omega_n \cdot \omega_g}\, d\tilde{n}$, we obtain

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \int \frac{L(\omega_i)\, F(\omega_h, \omega_i)\, \frac{\langle \omega_n, \omega_o \rangle}{\omega_n \cdot \omega_g}}{1 + \Lambda(\omega_o) + \Lambda(\omega_i)}\, P_{22}(\tilde{n})\, d\tilde{n}, \quad (19)$$

which expresses the outgoing radiance from the microsurface as an integration over its slopes statistics. This formulation is more convenient because efficient numerical solutions can be derived, as we will show in Section 5. Before diving into these details, we provide next our shading formulation for rough diffuse surfaces.

4.5 Diffuse Microfacets

A diffuse BRDF is constant $\rho(\omega_n, \omega_o, \omega_i) = \frac{1}{\pi}$. Substituting this formulation in Equation (5) yields

$$\mathbb{E}[R_n(\omega_n, \omega_o, \omega_i)] = \frac{1}{\pi} \int_{\Omega_n} \langle \omega_n, \omega_o \rangle \langle \omega_n, \omega_i \rangle\, D(\omega_n)\, d\omega_n.$$

As in the specular case, we first rewrite Equation (7) with this specific formulation of ρ, obtaining

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \int_{\Omega_i} L(\omega_i) \frac{\frac{1}{\pi} \int_{\Omega_n} \langle \omega_n, \omega_o \rangle \langle \omega_n, \omega_i \rangle\, D(\omega_n)\, d\omega_n}{1 + \Lambda(\omega_o) + \Lambda(\omega_i)}\, d\omega_i$$

which, after the same change of variable $D(\omega_n)\, d\omega_n = \frac{P_{22}(\tilde{n})}{\omega_n \cdot \omega_g}\, d\tilde{n}$, leads to

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \int_{\Omega_i} L(\omega_i) \frac{\frac{1}{\pi} \int \frac{\langle \omega_n, \omega_o \rangle \langle \omega_n, \omega_i \rangle}{\omega_n \cdot \omega_g}\, P_{22}(\tilde{n})\, d\tilde{n}}{1 + \Lambda(\omega_o) + \Lambda(\omega_i)}\, d\omega_i.$$

Even under a single directional light, this equation has no analytical solution. It can however be reformulated into another integral involving irradiance mapping [Ramamoorthi and Hanrahan 2001]. Denoting as $E(\omega_n)$ the irradiance map satisfying

$$E(\omega_n) = \int_{\Omega_i} \frac{L(\omega_i)\, \langle \omega_n, \omega_i \rangle}{1 + \Lambda(\omega_o) + \Lambda(\omega_i)}\, d\omega_i, \quad (20)$$

we derive our final shading expression for a rough diffuse surface:

$$I = \frac{\omega_{\bar{n}} \cdot \omega_g}{\omega_{\bar{n}} \cdot \omega_o} \frac{1}{\pi} \int E(\omega_n)\, \frac{\langle \omega_n, \omega_o \rangle}{\omega_n \cdot \omega_g}\, P_{22}(\tilde{n})\, d\tilde{n}. \quad (21)$$

5 Sampling Algorithms

The expressions we have derived for diffuse surfaces (Equation (21)) as well as specular surfaces in the special case of environment lighting (Equation (19)) must be computed numerically. In order to maintain high performance, we next derive an efficient sampling scheme, specifically adapted to solve these integrals as efficiently as possible.

5.1 Sampling Pattern

Our numerical solution is based on a sampling scheme whose samples are regularly and uniformly distributed on the $\mathbb{R}^2$ plane, and centered at the origin. For a 2D set of N^2 samples, we thus only need to store $\lceil \frac{N}{2} \rceil$ positions and weights. We initialize each sample by uniformly sampling a standard 1D normal distribution $\mathcal{N}(0, 1)$ so that the set covers an arbitrary interval $[-\lambda, \lambda] \times [-\lambda, \lambda]$, chosen for best results (Figure 6(a)). For instance, we set $\lambda = 1.8$ for $N = 5$. The normalized weight W_j associated to point p_j is given by

$$W_j = \frac{\exp\left(-\frac{1}{2}\, p_j^2\right)}{\sum_{k=1}^{N} \exp(-\frac{1}{2}\, p_k^2)}.$$

At runtime, we warp each sample in order to match P_{22} in Equations (19) and (21), which results in a new sample position $\tilde{n} = (x_{\tilde{n}}, y_{\tilde{n}})$ and weight $W_{\tilde{n}}$. We accomplish this by multiplying the 2D sample (p_j, p_k) by the Cholesky decomposition of the covariance matrix of P_{22} [Jäckel 2005]:

$$\begin{aligned} x_{\tilde{n}} &= p_j\, \sigma_x + \mathbb{E}[x_{\tilde{n}}] \\ y_{\tilde{n}} &= \left(r_{xy}\, p_j + \sqrt{1 - r_{xy}^2}\, p_k \right) \sigma_y + \mathbb{E}[y_{\tilde{n}}] \\ W_{\tilde{n}} &= W_j\, W_k \end{aligned}$$

where $r_{xy} = c_{xy}/(\sigma_x \sigma_y)$ is the correlation coefficient of Σ. This whole process is illustrated in Figure 6, and provides a lightweight collection of samples to compute Equations (19) and (21), further detailed in the next section. Our sample set converges to the exact value as N increases, and we further discuss their properties in Section 6.

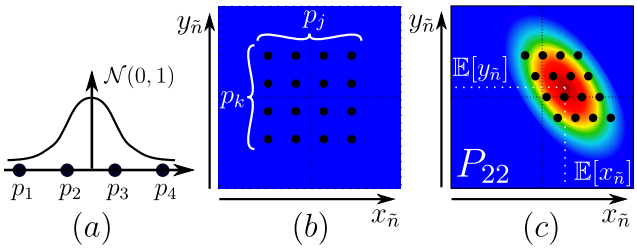

Figure 6: Sampling distribution P_{22}. *(a) We precompute uniform 1D samples p_j. (b) In the fragment shader we compute the Cartesian product and get 2D samples (p_j, p_k). (c) We scale, correlate, and shift the samples to match distribution P_{22}.*

5.2 Sampling Environment Lighting

We numerically compute Equation (19) using our sample set:

$$I = \frac{\omega_{\tilde{n}} \cdot \omega_g}{\omega_{\tilde{n}} \cdot \omega_o} \sum_{N^2} \frac{L(\omega_i)\, F(\omega_h, \omega_i)\, \frac{\langle \omega_n, \omega_o \rangle}{\omega_n \cdot \omega_g}}{1 + \Lambda(\omega_o) + \Lambda(\omega_i)} W_{\tilde{n}}. \tag{22}$$

For each slope sample $\tilde{n}$, we compute the associated normal ω_n and the reflected vector ω_i, using respectively Equations (11) and (18), to sample the environment map. Directly sampling the environment map from ω_i could produce noise or banding artifacts, especially with the small values for N that we target. To prevent this, we derive a method inspired by *filtered importance sampling* [Colbert and Křivánek 2007; Křivánek and Colbert 2008]: Instead of sampling with a set of delta directions, we emulate a smooth reconstruction filter using a prefiltered environment map. For each sample, we compute the level of detail according to the solid angle α covered in the incident direction ω_i

$$\alpha \approx \left\| \frac{\partial \omega_i}{\partial \tilde{n}} \right\| A,$$

where $A = (2\lambda \max(\sigma_x, \sigma_y)/N)^2$ is the surface area of a sample in slope space. Because we use isotropic samples, we further overestimate it by squaring the length of the sample in the largest direction of eccentricity of P_{22} (hence the term $\max(\sigma_x, \sigma_y)$). This guarantees that the solid angles overlap across neighboring samples and avoids banding artifacts for any N. For the sake of completeness, we provide the Jacobian of the slope to reflected vector:

$$\left\| \frac{\partial \omega_i}{\partial \tilde{n}} \right\| = \left\| \frac{\partial \omega_i}{\partial \omega_n} \right\| \left\| \frac{\partial \omega_n}{\partial \tilde{n}} \right\| = 4\, |\omega_n \cdot \omega_o|\, |\omega_n \cdot \omega_g|^3.$$

The whole computation is detailed in Algorithm 1 and illustrated in Figure 7, and provides efficient and artifact-free results to Equation (19).

5.3 Sampling Irradiance

We compute Equation (21) in three steps. The first two steps consist in building a spherical harmonic (SH) representation of the irradiance map [Ramamoorthi and Hanrahan 2001] that we defined in Equation (20), that we then sample.

Irradiance Map. Since our irradiance map formulation from Equation (20) depends on the masking-shadowing term, we compute the map on the fly using the local frame of $\mathcal{P}$. In Algorithm 2, we compute the first nine SH coefficients of the irradiance map due to point/directional lights, accounting for the masking-shadowing term by weighting their respective energies by $\frac{1}{1+\Lambda(\omega_o)+\Lambda(\omega_i)}$.

Unfortunately, interactively computing SH coefficients L_{lm}(environment) for an environment map is too costly. Therefore, we precompute them once on the CPU; at runtime, we multiply them by the masking term $\frac{1}{1+\Lambda(\omega_o)}$ alone, and add them to the coefficients computed for point/directional lights. With this approach, we correctly weight point/directional light contributions by the masking-shadowing term but only use the masking term for environment lights.

For completeness, we recall the formula to sample from the irradiance map given the SH coefficients [Ramamoorthi and Hanrahan

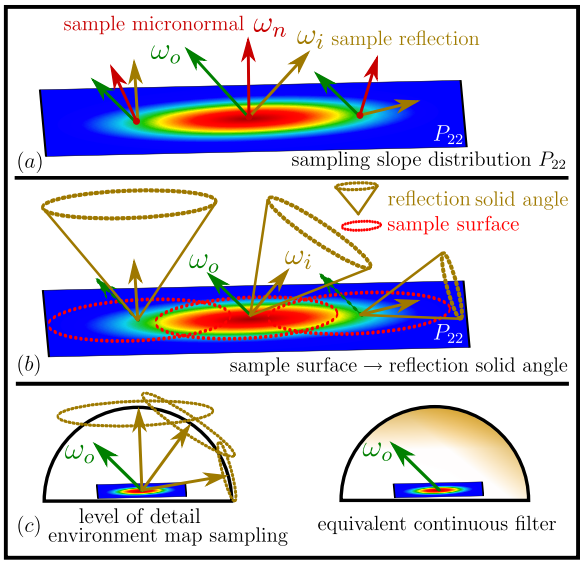

Figure 7: Filtered environment map sampling. *(a) We sample slope distribution P_{22} and for each micronormal sample ω_n we compute reflected vector ω_i. (b) The surface covered by the samples in slope space are transformed into solid angles by multiplying by the Jacobian of the transformation. (c) For each reflected direction ω_i we fetch the environment map with levels of detail determined by the solid angles. The reconstruction is smooth and noise-free.*

Algorithm 1 Environment map sampling

function ENVMAPSAMPLING($\omega_o, \mathbb{E}[\tilde{n}], \sigma_x, \sigma_y, c_{xy}$)
  $I = 0$
  for $j, k = 1..N$ **do**
    $x_{\tilde{n}} = p_j\, \sigma_x + \mathbb{E}[x_{\tilde{n}}]$
    $y_{\tilde{n}} = \left(r_{xy}\, p_j + \sqrt{1 - r_{xy}^2}\, p_k\right)\, \sigma_y + \mathbb{E}[y_{\tilde{n}}]$
    $W_{\tilde{n}} = W_j\, W_k$
    $\omega_n = (-x_{\tilde{n}}, -y_{\tilde{n}}, 1)/\sqrt{x_{\tilde{n}}^2 + y_{\tilde{n}}^2 + 1}$
    $\omega_i = 2\,(\omega_n \cdot \omega_o)\, \omega_n - \omega_o$
    $J = 4 \times |\omega_n \cdot \omega_o| \times |\omega_n \cdot \omega_g|^3$
    $\alpha = J \times A$
    $I += W_{\tilde{n}} \times \frac{F(\omega_n, \omega_i)\, \frac{\langle \omega_n, \omega_o \rangle}{\omega_n \cdot \omega_g}}{1+\Lambda(\omega_o)+\Lambda(\omega_i)} \times \text{textureLod}(\omega_i, \text{LOD}(\alpha))$
  end for
return $I \times \frac{\omega_{\tilde{n}} \cdot \omega_g}{\omega_{\tilde{n}} \cdot \omega_o}$
end function

Algorithm 2 Computing the irradiance map

function INITIRRADIANCEMAPCOEFS
  for $0 \leq l \leq 2$ and $-l \leq m \leq l$ **do**
    $L_{lm} = 0$
  end for
  for each directional light of radiance L and direction δ_{ω_i} **do**
    $P = L/(1 + \Lambda(\omega_o) + \Lambda(\omega_i))$
    $L_{00} += L_{00} + 0.282095 \times P$
    $(L_{11}, L_{10}, L_{1-1}) += 0.488603 \times (x_i, z_i, y_i) \times P$
    $(L_{21}, L_{2-1}, L_{2-2}) += 1.092548 \times (x_i z_i, y_i z_i, x_i y_i) \times P$
    $L_{20} += 0.315392 \times (3z_i^2 - 1) \times P$
    $L_{22} += 0.546274 \times (x_i^2 - y_i^2) \times P$
  end for
$L_{lm} += L_{lm}(\text{environment})/(1 + \Lambda(\omega_o))$
end function

2001]

$$\begin{aligned} E(\omega_n) = {} & 0.429043 L_{22}(x_n^2 - y_n^2) + 0.743125 L_{20} z_n^2 \\ & + 0.886227 L_{00} - 0.247708 L_{20} \\ & + 0.858086(L_{2-2} x_n y_n + L_{21} x_n z_n + L_{2-1} y_n z_n) \\ & + 1.023328(L_{11} x_n + L_{1-1} y_n + L_{10} z_n). \end{aligned}$$

Irradiance Integration. Once we have computed the irradiance map, we discretize Equation (21) as

$$I = \frac{\omega_{\tilde{n}} \cdot \omega_g}{\omega_{\tilde{n}} \cdot \omega_o} \frac{1}{\pi} \sum_{N^2} E(\omega_n) \frac{\langle \omega_n, \omega_o \rangle}{\omega_n \cdot \omega_g} W_{\tilde{n}} \tag{23}$$

and compute it numerically with Algorithm 3. We observed that, contrary to specular sampling, irradiance maps need not be prefiltered since they are already low frequency. This prevents artifacts during sampling.

Algorithm 3 Irradiance map sampling

function IRRADIANCEMAPSAMPLING($\omega_o, \mathbb{E}[\tilde{n}], \sigma_x, \sigma_y, c_{xy}$)
  $I = 0$
  for $j, k = 1..N$ **do**
    $x_{\tilde{n}} = p_j\, \sigma_x + \mathbb{E}[x_{\tilde{n}}]$
    $y_{\tilde{n}} = \left(r_{xy}\, p_j + \sqrt{1 - r_{xy}^2}\, p_k\right)\, \sigma_y + \mathbb{E}[y_{\tilde{n}}]$
    $W_{\tilde{n}} = W_j\, W_k$
    $\omega_n = (-x_{\tilde{n}}, -y_{\tilde{n}}, 1)/\sqrt{x_{\tilde{n}}^2 + y_{\tilde{n}}^2 + 1}$
    $I += W_{\tilde{n}} \times \frac{\langle \omega_n, \omega_o \rangle}{\omega_n \cdot \omega_g} \times E(\omega_n)$
  end for
return $I \times \frac{\omega_{\tilde{n}} \cdot \omega_g}{\omega_{\tilde{n}} \cdot \omega_o} \times \frac{1}{\pi}$
end function

6 Discussion

In this section we review the properties of our models and algorithms, and discuss choices we made to elaborate our representation.

BRDF Models. Our BRDF models are designed to be parameterized with noncentered anisotropic Beckmann distributions, and therefore may be interpreted as generalizations of previous work, thus remaining compatible with them. In the case of centered distributions, our specular model corresponds to the one proposed by Ross et al. [2005], which is essentially the same as Walter et al.'s model [2007] used with Beckmann distributions. The only difference lies in the masking-shadowing function: Walter et al. use the separable form $\frac{1}{1+\Lambda(\omega_o)} \times \frac{1}{1+\Lambda(\omega_i)}$ while we use the joint form $\frac{1}{1+\Lambda(\omega_o)+\Lambda(\omega_i)}$ proposed by Ross et al., which better models the correlation between visibility for view and light directions. In the case of centered and isotropic Beckmann distributions, our diffuse BRDF model corresponds to Oren and Nayar's model [1994]. Our diffuse BRDF model is more general because it includes noncentered and anisotropic distributions. Within the scope of this paper we focus on Beckmann distributions because they are convenient for multi-scale representations. However, our general formulation for noncentered distributions presented in Equation (7) can be used with other normal distribution functions, such as Phong or GGX [Walter et al. 2007].

Mono- vs. Multi-lobe Normal Distributions. Some previous work uses multiple lobes to represent and store normal distributions [Fournier 1992; Tan et al. 2005; Han et al. 2007; Tan et al. 2008]. While multiple lobes can represent more complex distributions, they are more expensive. Also, their main problem lies in the precomputations. To make interpolation possible, every lobe in a texel must match the same lobe in the neighboring texels [Tan et al. 2005; Han et al. 2007; Tan et al. 2008]. Solving this problem requires heavy nonlinear optimizations, and matching failures may result in visual artifacts when two nonmatching lobes are interpolated. To allow for lightweight memory storage, simple and fast precomputations, as well as no-failure interpolation, we chose to use a single anisotropic lobe representation.

Shadowing. Physically based rendering techniques should account for shadowing to ensure energy conservation. Our BRDF model accounts for this effect: shadowing that occurs at the level of the macrosurface (typically using a shadow map or a horizon map) is still represented at the level of the microsurface by the BRDF model; energy is conserved as well. However, in a real-life implementation one may decide to neglect shadowing because of performance or memory constraints. In this case, to ensure consistency across scales, the BRDF masking-shadowing term $\frac{1}{1+\Lambda(\omega_o)+\Lambda(\omega_i)}$ must be replaced by the masking term alone $\frac{1}{1+\Lambda(\omega_o)}$.

Performance of Our Algorithms. We provide an analytical formulation to evaluate our specular BRDF model with point and directional lights. It requires more computations than LEAN mapping alone because of our masking-shadowing term, but has similar cost than any other physically based model overall. For environment lights, we propose a sampling scheme. We rely on hardware mipmapping and select the best mipmap level to fetch the environment map by appropriately evaluating the solid angle of each fetch. This ensures noise- and artifact-free evaluations for any number of sample points as opposed to Monte Carlo sampling. The number of samples can then be chosen as a tradeoff between bias and performance. Our diffuse BRDF model has no analytical solutions and we resorted to our sampling strategy even for a single light source. We tried to find an analytical fitting with rational polynomials, using Pacanowski et al.'s [2012] fitting library. However, our BRDF is 8D (ω_o, ω_i, $\mathbb{E}[\tilde{n}]$, σ_x, σ_y) and the rational polynomials required to provide a good fit ended up having more than 70 coefficients, and resulted in lower performance than our integration scheme. To amortize the cost of the integration, we first splat all light directions in an irradiance map, and then make the numerical integration on the resulting irradiance map just once. For L lights and N sample points the complexity of the algorithm is $O(L) + O(N)$, instead of $O(L \times N)$. The cost of the integration is also amortized as the number of light sources increases. In practice, we use a 5×5 grid

of points for the diffuse integration, because they capture the main view-dependent effects. Note that the spherical harmonic representation of irradiance maps [Ramamoorthi and Hanrahan 2001] can result in errors of up to 9%. This is accurate enough for many applications, but one would need to sample the lights independently to produce exact results.

An important property of our algorithm is *scalability*. The further away the rendered object is, the more details get projected in the same pixel, and the more samples a typical Monte Carlo integration would require to solve Equation (2). Our algorithm is able to recover complex reflectance with computational complexity independent of the size of the filter extent. This key property comes from the fact that we replaced the nonscalable spatial integration (i.e., Equation (2)) by a scalable integration in the space of the surface statistics (i.e., Equation (7)).

Uniform Sampling. In Section 5.1 we showed how to sample a Gaussian distribution P_{22} with uniform 2D grids. We tried other sampling strategies before opting for the grid approach. The importance sampling of P_{22} with a method such as the one of Colbert and Křivánek [2008] is too sensitive to the random samples configuration for small values of N; the uniform grid provides better results (see our supplemental document for comparisons). We also tried to use Gauss-Hermite quadratures, which are specifically designed for Gaussian kernels. However, this quadrature scheme only works well with polynomial functions, and Equations (22) and (23) contain many terms that are hardly approximated by polynomials (the function is not even C^1). Again, our simple uniform sampling grid provided better results than Gauss-Hermite quadrature. The sampling pattern may be improved with further analysis [Heck et al. 2013; Subr and Kautz 2013], but this is out of the scope of this paper.

Domain of Validity of Our Method. Our model and its derivations are based on the fundamental assumption that the displacement map is applied over a locally planar patch. In theory, our method is not valid when the pixel footprint covers a curved macrosurface since the curvature must be filtered along with the displacement map. For instance, the claws of the *T-rex* model in Figure 1 are small, smooth, curved, and highly specular. As such, they exhibit aliasing that cannot be filtered by LEADR mapping alone. In practice, we deal with this problem by offsetting base roughness heuristically due to curvature, and add it to the slope covariance matrix of the displacement map. This way, we merge the normal distribution of the curved macrosurface with the LEADR map. This "trick" effectively removes the aliasing from small and highly curved macrosurfaces, but is not well defined, as we arbitrarily set the mapping from curvature to roughness. Properly filtering displacement maps along with large pieces of macrosurfaces is complex and remains an open problem [Bruneton and Neyret 2012].

7 Results

Renderings. We present a set of very rough spheres rendered under different configurations in Figure 8. The top two rows display ground truth computed with supersampling of geometry synthesized directly from normal and displacement maps, and the bottom row displays the results of our method. The images show that normal mapping fails to capture view-dependent effects due to masking and projection weighting. Since normal map filtering methods target ground truth when *only* normal maps are used, they cannot reproduce these more realistic effects. On the other hand, LEADR mapping accurately reproduces this more complex appearance. Figure 9 provides other comparisons with more complex geometry: the pillow (top) is illuminated with an environment light only, and the snake (bottom), with an environment map and four point light sources.

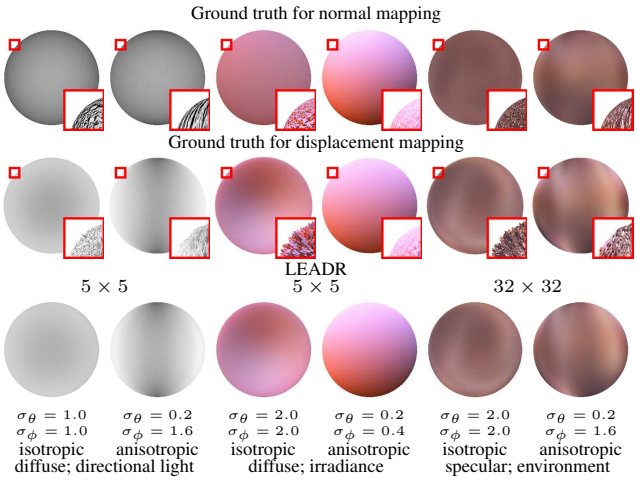

Figure 8: *Ground truth generated with procedural normal mapping (top row), and displacement mapping (middle row).* 5×5 *sampled LEADR mapping for diffuse microfacets, and* 32×32 *for specular microfacets (bottom row). More complete tables of rendered spheres are provided in the supplemental document.*

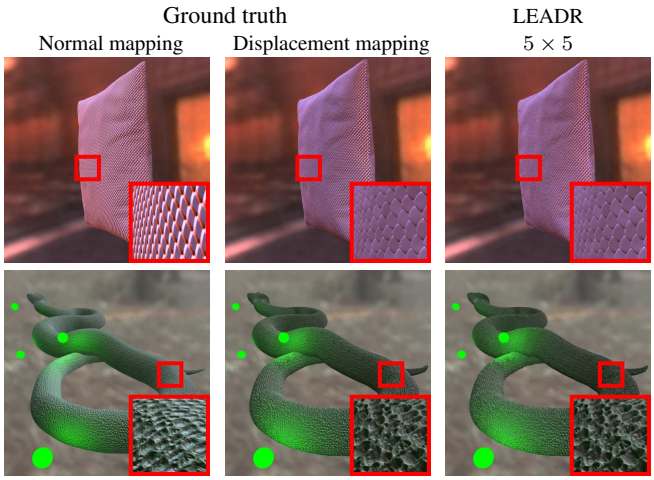

Figure 9: *Top row: Environment map applied to diffuse microfacet distributions. The ground truth for normal mapping (left) fails to capture the view dependence due to small-scale geometry in the ground-truth displaced supersampled result (center). Our method accurately captures these effects (right). Bottom row: Environment and point lighting applied to diffuse microfacet distributions. At close-up views (insets) the extent of the filter is appropriately reduced and our result matches the ground truth for displacement mapping.*

We also applied LEADR mapping on a multiscale ocean displacement map, animated with the method of Bruneton et al. [2010] (Figure 10). We use our analytic formula to evaluate shading from the sun (which is approximated as a directional light), and our numerical scheme to compute filtered reflection from the environment map. Figure 5 shows that our technique is robust to extreme deformation and animation.

Finally, we tested our approach on a complex, production-quality

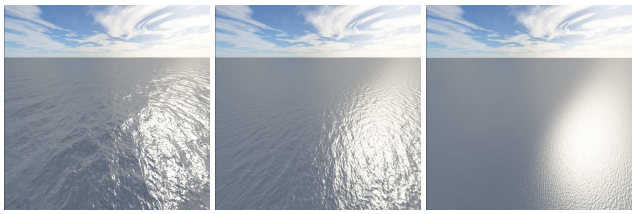

Figure 10: *Procedural ocean displacement map rendered with our specular microfacet BRDFs, at multiple scales within an image and between an animated zoom-out sequence. From left to right, the ocean is viewed from progressively increasing distances. The specular highlights are correctly and continuously antialiased while zooming out.*

geometry asset with several highly detailed displacement maps in Figure 1. This result is part of an animated walking cycle, available in the accompanying video, also computed interactively with our technique. While our model has been designed with real-time rendering in mind, it could prove accurate even for some productions.

Performance. We report various rendering times in the following tables. These numbers, all expressed in milliseconds, are provided as a mean to quantify the performance impact of our model over more naive, yet commonly used, shading methods. All numbers are captured on an Intel Core i7-930 CPU at 2.80 GHz, and an NVidia GTX 480 GPU. To obtain a baseline performance value, we rendered a full-screen quad over a 1024×1024 framebuffer with a LEADR map of matching resolution. While no filtering occurs, this configuration guarantees that the fragment shader load is the same across all shading configurations, and gives a lower performance bound since coarser mip levels could benefit from superior cache optimizations.

For one directional light source and specular reflection, the baseline times are

Constant	Blinn-Phong	LEAN	LEADR
0.055	0.146	0.266	0.267

Here, LEADR mapping is as fast as LEAN mapping despite requiring more arithmetic operations. This is most probably due to the bottleneck texture fetches during rendering from the LEADR map. Compared to a naive Blinn-Phong shader, the performance differences are also quite negligible. The constant color configuration outputs constant fragments, and is provided as a measure of the geometry processing overhead.

In the next table, we use the same configuration, but with environment lighting. We compute our specular (resp. diffuse) shading equations at three grid resolutions. Their timings are compared to a single texture fetch along the mean reflected direction for the specular case, and a single irradiance map evaluation using the mean normal for the diffuse case.

Reflection	One sample	3×3	5×5	7×7
specular	0.402	0.433	1.704	4.284
diffuse	0.148	1.067	2.750	5.027

As illustrated in Figure 11, LEADR mapping performance scales linearly according to the number of samples used. We believe that interesting gains can be made by optimizing the placement of the samples and/or employing sample pruning, as suggested by Jäckel [2005], in order to obtain similar quality but significantly fewer samples. We leave this to future work.

For the complex production-quality *T-rex* model in Figure 1, our

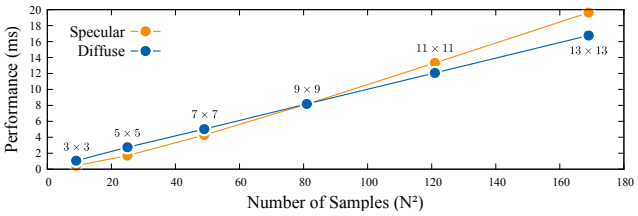

Figure 11: *LEADR mapping render times under environment light using an increasing number of samples on an NVidia GTX 480.*

framework achieves performances ranging between 40 and 60 milliseconds at an image resolution of 1024×1024. For strong close-up and distant views ($\approx 128 \times 128$ pixel coverage), the rendering times are about 200 and 9 milliseconds, respectively. Note that in the accompanying video, we used an NVidia Quadro 6000 GPU with 6 GB of memory so that we could fit the original, fully detailed 2.8 GB displacement textures. We observed similar performances on this platform.

Overall, our method demonstrates a clear quality improvement over naive shading methods, all while maintaining very high performance. Furthermore, our technique exhibits spatial and temporal coherence and always generates alias-free images closely matching ground truth, across all scales.

8 Conclusion

We have presented LEADR mapping, a lightweight unified representation for physically based BRDFs and small-scale geometries defined with displacement mapping. LEADR mapping generates smooth and antialiased transitions when zooming in and out of displaced surfaces. The BRDF seamlessly *emerges* from the microgeometry, and accurately captures complex shading effects. We proposed a simple, lightweight, and efficient sampling scheme to compute shading from complex environment lightings in real time. We validated our model against ground truth and successfully applied it on static and animating/deforming geometries. Unlike supersampling, our method properly reconstructs shading effects when dense subpixel geometric detail is present, even in the difficult case of highly specular microgeometry.

LEADR mapping is limited to the paradigm of displacement mapping, where the local tangent frame is available. Further filtering would require including the macrosurface itself, with curvature and nonlocality. Along with improving our sampling strategies, as discussed earlier, we believe that our approach will promote new work on this important problem.

Acknowledgements

The authors would like to sincerely thank Brent Burley and Disney for giving access to the *T-rex* model and its detailed textures. Laurent Belcour provided much appreciated feedback on an earlier version of the paper, and Derek Nowrouzezahrai was crucial with his timely suggestions. Dupuy and Heitz acknowledge financial support from Explo'ra Doc from the Rhône-Alpes region. In particular, Poulin and Dupuy acknowledge support from GRAND and NSERC, and Ostromoukhov and Dupuy are supported by the ANR excellence chair AMCQMCSCGA, from which Iehl also benefitted.

References

BAKER, D. 2011. Spectacular specular – LEAN and CLEAN specular highlights. In *Proc. Game Developer Conference 2011*.

BECKER, B. G., AND MAX, N. L. 1993. Smooth transitions between bump rendering algorithms. In *Proc. SIGGRAPH '93*, 183–190.

BECKMANN, P., AND SPIZZICHINO, A. 1963. *The scattering of electromagnetic waves from rough surfaces*. International series of monographs on electromagnetic waves. Pergamon Press; [distributed in the Western Hemisphere by Macmillan, New York].

BOURLIER, C., SAILLARD, J., AND BERGINC, G. 2000. Effect of correlation between shadowing and shadowed points on the Wagner and Smith monostatic one-dimensional shadowing functions. *IEEE Trans. on Antennas and Propagation 48*, 3, 437–446.

BRUNETON, E., AND NEYRET, F. 2012. A survey of non-linear pre-filtering methods for efficient and accurate surface shading. *IEEE Trans. on Visualization and Computer Graphics 18*, 2, 242–260.

BRUNETON, E., NEYRET, F., AND HOLZSCHUCH, N. 2010. Real-time realistic ocean lighting using seamless transitions from geometry to BRDF. *Comput. Graph. Forum 29*, 2, 487–496.

CABRAL, B., MAX, N., AND SPRINGMEYER, R. 1987. Bidirectional reflection functions from surface bump maps. In *Proc. SIGGRAPH '87*, 273–281.

COLBERT, M., AND KŘIVÁNEK, J. 2007. GPU-based importance sampling. In *GPU Gems 3*. Addison-Wesley, ch. 20.

COOK, R. L., AND TORRANCE, K. E. 1982. A reflectance model for computer graphics. *ACM Trans. Graph. 1*, 1 (Jan.), 7–24.

FOURNIER, A. 1992. Normal distribution functions and multiple surfaces. In *Proc. Graphics Interface '92 Workshop on Local Illumination*, 45–52.

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

HECK, D., SCHLÖMER, T., AND DEUSSEN, O. 2013. Blue noise sampling with controlled aliasing. *ACM Trans. on Graphics 32*, 3 (July), 25:1–12.

HEITZ, E., AND NEYRET, F. 2012. Representing appearance and pre-filtering subpixel data in sparse voxel octrees. In *Proc. High Performance Graphics*, ACM, 125–134.

HEITZ, E., NOWROUZEZAHRAI, D., POULIN, P., AND NEYRET, F. 2013. Filtering color mapped textures and surfaces. In *Proc. Symp. on Interactive 3D Graphics and Games*, ACM, 129–136.

JÄCKEL, P., 2005. A note on multivariate Gauss-Hermite quadrature. http://www.jaeckel.org/.

KILGARD, M. 2000. A practical and robust bump-mapping technique for todays GPUs. In *Proc. Game Developer Conference 2000*.

KŘIVÁNEK, J., AND COLBERT, M. 2008. Real-time shading with filtered importance sampling. *Computer Graphics Forum (Proc. Eurographics Symposium on Rendering) 27*, 4, 1147–1154.

MA, W.-C., CHAO, S.-H., TSENG, Y.-T., CHUANG, Y.-Y., CHANG, C.-F., CHEN, B.-Y., AND OUHYOUNG, M. 2005. Level-of-detail representation of bidirectional texture functions for real-time rendering. In *Proc. Symp. on Interactive 3D Graphics and Games*, ACM, 187–194.

MCAULEY, S., HILL, S., HOFFMAN, N., GOTANDA, Y., SMITS, B., BURLEY, B., AND MARTINEZ, A. 2012. Practical physically-based shading in film and game production. In *ACM SIGGRAPH 2012 Courses*, ACM, SIGGRAPH '12, 10:1–7.

OLANO, M., AND BAKER, D. 2010. LEAN mapping. In *Proc. Symp. on Interactive 3D Graphics and Games*, ACM, 181–188.

OLANO, M., AND NORTH, M. 1997. Normal distribution mapping. *Univ. of North Carolina Comput. Sc. Tech. Report*, 97–041.

OREN, M., AND NAYAR, S. K. 1994. Generalization of Lambert's reflectance model. In *Proc. SIGGRAPH '94*, 239–246.

PACANOWSKI, R., SALAZAR CELIS, O., SCHLICK, C., GRANIER, X., POULIN, P., AND CUYT, A. 2012. Rational BRDF. *IEEE Trans. on Visualization and Computer Graphics 18*, 11, 1824–1835.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. An efficient representation for irradiance environment maps. In *Proc. SIGGRAPH '01*, ACM, 497–500.

ROSS, V., DION, D., AND POTVIN, G. 2005. Detailed analytical approach to the gaussian surface bidirectional reflectance distribution function specular component applied to the sea surface. *J. Opt. Soc. Am. A 22*, 11 (Nov), 2442–2453.

SMITH, B. 1967. Geometrical shadowing of a random rough surface. *IEEE Trans. on Antennas and Propagation 15*, 668–671.

SUBR, K., AND KAUTZ, J. 2013. Fourier analysis of stochastic sampling strategies for assessing bias and variance in integration. *ACM Trans. on Graphics 32*, 4 (July), 128:1–12.

TAN, P., LIN, S., QUAN, L., GUO, B., AND SHUM, H.-Y. 2005. Multiresolution reflectance filtering. In *Proc. Eurographics Symposium on Rendering*, EGSR'05, 111–116.

TAN, P., LIN, S., QUAN, L., GUO, B., AND SHUM, H. 2008. Filtering and rendering of resolution-dependent reflectance models. *IEEE Trans. on Visualization and Computer Graphics 14*, 2, 412–425.

TOKSVIG, M. 2005. Mipmapping normal maps. *Journal of Graphics, GPU, and Game Tools 10*, 3, 65–71.

WALTER, B., MARSCHNER, S. R., LI, H., AND TORRANCE, K. E. 2007. Microfacet models for refraction through rough surfaces. In *Proc. Eurographics Symposium on Rendering*, EGSR'07, 195–206.

WU, H., DORSEY, J., AND RUSHMEIER, H. 2009. Characteristic point maps. *Computer Graphics Forum (Proc. of Eurographics Symposium on Rendering) 28*, 4, 1227–1236.