Accurate Appearance Preserving Prefiltering for Rendering Displacement-Mapped Surfaces

LIFAN WU, University of California, San Diego
SHUANG ZHAO, University of California, Irvine
LING-QI YAN, University of California, Santa Barbara
RAVI RAMAMOORTHI, University of California, San Diego

Fig. 1. We present a new approach to prefilter high-resolution displacement maps while preserving the input appearance. High-resolution displacement maps can produce rich geometric details (top-left) but they are difficult to prefilter. Our prefiltered model handles the change of shadowing, masking and interreflections caused by downsampling the displacement map. At a single scale, although the detailed micro-structures are different (see ours (64x)² at left), our prefiltered model preserves the original appearance accurately when we view the object from a distance (see the right image). We can also combine our models at multiple downsampling scales to form a mipmap, enabling accurate and anti-aliased LoD rendering.

Prefiltering the reflectance of a displacement-mapped surface while preserving its overall appearance is challenging, as smoothing a displacement map causes complex changes of illumination effects such as shadowing-masking and interreflection. In this paper, we introduce a new method that prefilters displacement maps and BRDFs jointly and constructs SVBRDFs at reduced resolutions. These SVBRDFs preserve the appearance of the input models by capturing both shadowing-masking and interreflection effects. To express our appearance-preserving SVBRDFs efficiently, we leverage a new representation that involves spatially varying NDFs and a novel scaling function that accurately captures micro-scale changes of shadowing, masking, and interreflection effects. Further, we show that the 6D scaling function can be factorized into a 2D function of surface location and a 4D function of direction. By exploiting the smoothness of these functions, we develop a simple and efficient factorization method that does not require computing the full scaling function. The resulting functions can be represented at low resolutions (e.g., 4x for the spatial function and 15x for the angular function), leading to minimal additional storage. Our method generalizes well to different types of geometries beyond Gaussian surfaces. Models prefiltered using our approach at different scales can be combined to form mipmaps, allowing accurate and anti-aliased level-of-detail (LoD) rendering.

CCS Concepts: • Computing methodologies → Rendering.

Additional Key Words and Phrases: multi-resolution, level of detail, prefiltering, global illumination

ACM Reference Format:

1 INTRODUCTION
High-resolution displacement maps are commonly used to describe detailed micro-geometries that can produce richly diverse appearances. Compared to normal mapping, displacement mapping is more physically consistent and can offer more realistic self-shadowing and silhouettes. However, such realism comes at the cost of difficult prefiltering: smoothing a displacement map usually weakens its intrinsic shadowing and results in brightened overall appearance. Therefore, rendering a high-resolution displacement map without introducing severe aliasing generally requires significant super-sampling, which is computationally expensive.

Previous displacement mapping techniques such as LEAN [Olano and Baker 2010] and LEADR [Dupuy et al. 2013] can produce anti-aliased renderings of rough surfaces. However, they assume the normals of the input surfaces to have Beckmann distributions, which is usually violated in practice and fundamentally limits the accuracy
of these methods (Figure 2(f)). To handle a wider range of surfaces, some bi-scale appearance models [Wu et al. 2011; Iwasaki et al. 2012] precompute the overall surface reflectance by averaging light reflections at the micro-scale. This yields one spatially macro-scale effective BRDF with no spatial variation, which has difficulties in reproducing micro-scale details at close-up views. Both approaches neglect interreflection and can lead to significant energy loss.

In this paper, we introduce a novel method to prefilter displacement-mapped (opaque) surfaces while accurately preserving their overall appearances under both direct and global illuminations (e.g., Figures 1, 2). Given a high-resolution displacement map and an isotropic base micro-BRDF, we seek displacement maps and accompanying surface reflectance models that have reduced resolutions and closely preserve the macro-scale appearance of the input model. To this end, we leverage a generalized version of the effective BRDF formulation [Wu et al. 2011] that takes both shadowing-masking and interreflection effects into account. Based on this formulation, we directly match the effective BRDFs of the input model and our prefiltered variants.

Our method (Figure 3) starts with downsampling the input displacement map. This is achieved via an optimization that aims to preserve the surface’s meso-scale geometries (e.g., normals) and minimize potential energy loss. The resulting surface, when coupled with the input reflectance model, generally leads to a different appearance (see Figure 2(d) for an example) due to the change of shadowing structures, distribution of normals, and interreflections caused by the downsampling. The main focus of this paper, therefore, is to find proper surface reflectance models (i.e., BRDFs) so that, when coupled with the downsampled geometries, they closely resemble the appearance of the input.

To fully preserve the detailed appearance of the input model using the downsampled geometry, a 6D spatially varying effective BRDF is generally required. Unfortunately, explicitly expressing this SVBRDF requires significant computation and storage, negating the benefits of displacement map downsampling. Instead, we decompose the 6D function into a 4D spatially varying normal distribution function (SVNDF) [Han et al. 2007] and a novel scaling function. The SVNDF, which neglects both shadowing-masking and interreflection, primarily encodes spatial variations of the effective BRDFs and can be represented as 2D spatially varying parametric distributions. The scaling function, in contrast, accounts for the shadowing-masking and interreflection effects and is vital for matching the input appearance. To efficiently describe this scaling function, which is 6D itself, we factorize it into a 2D spatial scaling function of surface location and a 4D angular scaling function of direction (e.g., Figure 10). Further, we exploit the fact that the scaling functions are usually low-frequency and smooth in practice to introduce a simple and efficient factorization method to compute them from a sparse set of samples. With a resolution up to $4^2$ for the spatial function and $15^2$ for the angular one, our scaling functions only consume 200–400 KB of storage, making our prefiltered models compact and practically useful. Our contributions include:

- We develop a novel method to prefilter the surface reflectance of a high-resolution displacement map while accurately preserving the overall appearance (Figures 1, 16, 17, 18). This is achieved by first downsampling the displacement map to minimize the meso-scale average surface slopes ($\S 4.1$), then separating a 6D SVBRDF into a SVNDF ($\S 4.2$) and a scaling function ($\S 4.3$).
- To match the target appearance, we utilize two scaling functions. One captures only shadowing-masking ($\S 4.3$) and the other effectively handles both shadowing-masking and interreflection ($\S 5.2$). To model interreflection within the micro-geometry, which is generally missing in previous work, we introduce a generalized formulation of effective BRDFs ($\S 5.1$).
- We show the scaling function can be factorized as the product of a single spatial scaling function and a single angular scaling function. Exploiting their low-frequency property and smoothness, we present a simple and efficient method to compute the spatial and angular scaling functions ($\S 6$).
- To enable level-of-detail (LoD) rendering, we present a linear interpolation method that provides smooth transitions between our models prefiltered at varying scales ($\S 7.2$).

2 RELATED WORK

We review several main research areas in the following paragraphs. Comparison between our method and the most related techniques is summarized in Table 1.

**Surface appearance prefiltering.** We refer readers to [Bruneton and Neyret 2012] for a comprehensive review of surface appearance prefiltering techniques. Traditional normal/BRDF map filtering methods [Fournier 1992; Tan et al. 2005; Toksvig 2005; Han et al.
Westin et al. [1992] explicitly simulate ray tracing on micro-geometry. Our method generates pure surface reflectance models that compress them as low-rank matrices. This work has been extended and bidirectional visible normal distribution functions (BVNDF) and scale material editing system. They precompute rotated BRDF values precomputed visibility. Wu et al. [2011] propose an interactive bi-

Heidrich et al. [2000] speed up the simulation on height fields using a single 4D angular scaling function. The latter scaling function can be further factorized into a single 2D spatial scaling function and a single 4D angular scaling function.

2007; Kaplanyan et al. 2016; Xu et al. 2017] ignore shadowing and masking effects. Tan et al. [2008] approximate shadowing and masking using horizon map distributions. LEADR [Dupuy et al. 2013] extends from LEAN mapping [Olano and Baker 2010] by incorporating a physically based shadowing-masking term derived from micro-facet theory. In LEADR, they use one-lobe NDFs for simplicity and efficiency in a real-time rendering context. We use multi-lobe NDFs to achieve better accuracy, especially at glossy highlights (Figure 16). Recently, Loubet and Neyret [2017] proposed a hybrid mesh-volume model to construct LoDs of complex objects. They leverage volumetric models to handle self-occlusions caused by micro-geometries, while our method generates pure surface reflectance models that are simpler and more efficient to use. In addition, none of these methods take interreflections into account, since interreflections combined with shadowing and masking are complicated to analyze and prefilter.

Bidirectional texture functions (BTF) [Cabral et al. 1987; Dana et al. 1999] capture spatially varying and view-dependent surface appearance. Generating 6D BTF data requires expensive precomputation and a large amount of storage. Although BTF filtering methods [Ma et al. 2005; Wu et al. 2009; Jarabo et al. 2014] provide a direct solution to surface appearance prefiltering, the high dimensionality limits their practical applications. Our method separates a full 6D function into a 4D SVNDF and a 6D scaling function. The latter scaling function can be further factorized into a single 2D spatial scaling function and a single 4D angular scaling function, which is cheaper to compute and compact to store. Complex shadowing-masking and interreflections can be captured by the spatial and angular scaling functions accurately.

Bi-scale material design. A series of works model object’s macro-scale appearance by manipulating its micro-scale details [Westin et al. 1992; Heidrich et al. 2000; Wu et al. 2011; Iwasaki et al. 2012], Westin et al. [1992] explicitly simulate ray tracing on micro-geometry. Heidrich et al. [2000] speed up the simulation on height fields using precomputed visibility. Wu et al. [2011] propose an interactive bi-scale material editing system. They precompute rotated BRDF values and bidirectional visible normal distribution functions (BVNDF) and compress them as low-rank matrices. This work has been extended to edit highly glossy materials by using mixtures of spherical Gaussians (SG) or anisotropic spherical Gaussians (ASG) [Iwasaki et al. 2012; Xu et al. 2013]. The shadowing-masking term is controlled by the weights of SG/ASG lobes. Most of these bi-scale appearance modeling techniques focus on the average large-scale appearance and do not consider spatial variation. In addition, the interactive approaches have no interreflection components. Recently, several methods for rendering glinty surfaces have been developed by simulating specular reflection on micro-surfaces [Yan et al. 2014; Jakob et al. 2014; Yan et al. 2016; Chermoukine et al. 2018]. Zirr and Kaplanyan [2016] propose a bi-scale microfacet model to render micro-details in real-time. Their methods focus on spatially varying NDFs, neglecting shadowing-masking and interreflections.

Microfacet models. Microfacet models describe the aggregate reflectance from a statistical representation of rough surfaces, i.e., the orientations of microfacets, resulting in a number of physically-based BRDFs [Cook and Torrance 1982; Oren and Nayar 1994; Walter et al. 2007]. The Smith model [Smith 1967; Heitz 2014] gives an accurate approximation of the microfacet shadowing-masking function with the assumption of independence between heights and normals. In particular, the shadowing-masking function has analytic solutions if Gaussian or GGX surfaces are given. Ashikhmin et al. [2000] derive a 4D BRDF from a 2D NDF, in which the shadowing-masking term is numerically computed from the NDF. LEADR [Dupuy et al. 2013] assumes Gaussian surfaces and leverages the Smith model to handle masking and shadowing effects. Recent works [Heitz et al. 2016; Lee et al. 2018; Xie and Hanrahan 2018] extend microfacet theory by modeling multiple scattering inside the micro-surface. Though microfacet models enable efficient computation of shadowing-masking and interreflections, they are limited by Gaussian surface, GGX surfaces or V-grooves. Our method can handle general surfaces without assumptions for specific micro-geometries.

Inverse rendering. Inverse rendering optimizes for scene parameters that produce the best match to target appearance. Previous methods [Hasan and Ramamoorthi 2013; Gkiolekas et al. 2013; Khungurn et al. 2015; Zhao et al. 2016; Gkiolekas et al. 2016] require global inverse rendering, which solves for a number of scattering parameters and involves expensive Monte Carlo path tracing during iterative optimizations. We just need to perform standard normal mapping and precompute simpler effective BRDFs once to obtain the prefiltered scattering parameters. Unlike inverse rendering techniques that depend on specific scene configurations, our model can generalize to different lighting and viewing conditions as we match the effective BRDFs rather than rendered images.

Height field rendering. Rendering displacement-mapped surfaces closely relates to height field rendering. Self-shadowing on height fields can be computed in real-time by determining horizon angles for a set of azimuthal directions [Snyder and Nowrouzezahrai 2008; Timonen and Wieserholm 2010]. Fast height field rendering with global illumination [Nowrouzezahrai and Snyder 2009] approximates visibility and indirect radiance using low-order SH basis functions, which cannot capture high-frequency reflection and shadowing effects. In these methods, height fields are attached on a planar base surface. Our method allows perturbing displacements on a general surface representation such as triangular meshes.

ACM Trans. Graph., Vol. 38, No. 4, Article 1. Publication date: July 2019.
Table 1. We compare our method with LEADR [Dupuy et al. 2013], bi-scale appearance models [Wu et al. 2013; Iwasaki et al. 2012] and multi-scattering microfacet models [Heitz et al. 2016; Lee et al. 2018; Xie and Hanrahan 2018]. Our method can handle both shadowing-masking and interreflections but is not limited to Gaussian/GGX surfaces, at the expense of some precomputation.

<table>
<thead>
<tr>
<th>Method</th>
<th>LEADR</th>
<th>bi-scale</th>
<th>microfacet</th>
<th>ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shadowing and masking</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Interreflections</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>General surfaces</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>No Precomputation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>

Table 2. Definitions of commonly used symbols.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Def.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>Surface patch</td>
<td>§3</td>
</tr>
<tr>
<td>$G(P)$</td>
<td>Micro-geometry defined by a surface patch $P$ of a displacement map</td>
<td>§3</td>
</tr>
<tr>
<td>$f$</td>
<td>Isotropic base BRDF</td>
<td>§3</td>
</tr>
<tr>
<td>$f_{\text{eff}}(G, f)$</td>
<td>Effective BRDF determined by micro-geometry $G$ and base BRDF $f$</td>
<td>§3, Eq. (6)</td>
</tr>
<tr>
<td>$G_{\text{orig}}$</td>
<td>Micro-geometry defined by the original high-resolution displacement map</td>
<td>§4</td>
</tr>
<tr>
<td>$f_{\text{orig}}$</td>
<td>Original input base BRDF</td>
<td>§4</td>
</tr>
<tr>
<td>$G_{\text{low}}$</td>
<td>Micro-geometry defined by the low-resolution displacement map</td>
<td>§4</td>
</tr>
<tr>
<td>$f_{\text{low}}$</td>
<td>Prefiltered spatially varying base BRDF</td>
<td>§4</td>
</tr>
<tr>
<td>$f_{\text{lr}}^{\text{eff}}$</td>
<td>Effective BRDF with interreflections</td>
<td>§5.1, Eq. (17)</td>
</tr>
<tr>
<td>$R, R_{\text{lr}}$</td>
<td>6D scaling function without/with interreflections</td>
<td>§4.3, Eq. (12)</td>
</tr>
<tr>
<td>$T, T_{\text{lr}}$</td>
<td>2D spatial scaling function without/with interreflections</td>
<td>§5.2, Eq. (20)</td>
</tr>
<tr>
<td>$S, S_{\text{lr}}$</td>
<td>4D angular scaling function without/with interreflections</td>
<td>§6, Eq. (21)</td>
</tr>
</tbody>
</table>

3 PRELIMINARIES

In this section, we provide preliminaries of displacement mapping, normal mapping, and effective BRDFs. Table 2 summarizes all symbols commonly used in this paper.

Displacement mapping. Displacement maps provide an efficient way to describe detailed micro-geometries. Mathematically, a displacement map is a function $h: [0, 1]^2 \rightarrow \mathbb{R}$ that specifies the distance which individual surface points are shifted along the normal directions. To be precise, given a texture-mapped base surface and a displacement map, the resulting geometry is obtained by moving each surface point with texture coordinate $(u, v)$ along its normal direction for a distance of $h(u, v)$. Further, a surface patch $\mathcal{P} \subseteq [0, 1]^2$ denotes a small and locally flat region of the base surface. For notational simplicity, the patch area is normalized using some filter kernel $k_P$ (i.e., $\int_\mathcal{P} k_P(p) \, dp = 1$). For each $p \in \mathcal{P}$, we denote its final position (perturbed by displacement mapping) and micro-normal as $x_m(p)$ and $\omega_m(p)$, respectively. Lastly, the micro-geometry given by a base patch $\mathcal{P}$ and a displacement map can be expressed by a collection of micro-positions and micro-normals: $G(\mathcal{P}) = \{(x_m(p), \omega_m(p)) | p \in \mathcal{P}\}$.

Meso-geometry. For a surface patch $\mathcal{P}$, we define a local tangent frame under which the macro-normal $\omega_o$ of the patch equals to $(0, 0, 1)$. Then, the meso-scale geometry of $G(\mathcal{P})$ can be described with the average slope:

$$\bar{s} = \int_\mathcal{P} s_m(p) k_\mathcal{P}(p) \, dp,$$

where $s_m(p) = (x_m / z_m - y_m / z_m)$ is the micro-slope with $x_m, y_m$, and $z_m$ given by the micro-normal at $p$ (i.e., $(x_m, y_m, z_m) = \omega_m(p)$) under the local tangent frame.

Normal mapping. Provided a displacement map, we can extract the normal distribution function (NDF) [Dupuy et al. 2013] for any given surface patch $\mathcal{P}$ via

$$D(\omega) = \int_\mathcal{P} \delta(\omega - \omega_m(p)) k_\mathcal{P}(p) \, dp,$$

where $(\cdot, \cdot)$ represents the dot product clamped to 0, and $\delta$ denotes the Dirac delta function. A physically valid NDF must satisfy

$$\int_{\mathcal{H}^2} D(\omega)(\omega, \omega_o) \, d\omega = \int_\mathcal{P} k_\mathcal{P}(p) \, dp = 1,$$

where $\mathcal{H}^2$ denotes the unit hemispherical domain. It means that the projected area of the micro-geometry onto the macro-normal $\omega_o$ should be equal to the area of the base surface patch.

An NDF $D$ can be approximated by a mixture of von Mises-Fisher (vMF) lobes [Han et al. 2007],

$$D(\omega) \approx \sum_{i=1}^m \alpha_i \gamma(\omega; \kappa_i, \mu_i),$$

where $\alpha_i$ represents the lobe’s amplitude. Each vMF lobe with bandwidth $\kappa_i$ and the center direction $\mu_i$ is defined as $\gamma(\omega; \kappa, \mu) = \frac{1}{4\pi \kappa} \exp(\kappa \cdot (\omega - \mu))$. This vMF-based parameterization allows efficient description of spatially varying NDFs (SVNDFs).

From an NDF $D$ and an isotropic base BRDF $f$, the composite multi-lobe BRDF can be formulated as [Han et al. 2007]

$$\rho(\omega_i, \omega_o; D, f) = \frac{1}{\cos \theta_i} \int_{\mathcal{H}^2} f(R_o(\omega_i), R_o(\omega_o))(\omega, \omega_o) D(\omega) \, d\omega.$$
Note that $\omega_l$ and $\omega_o$ are in the macro-scale tangent frame defined by the macro-normal $\omega_g$. We need a rotation function $R_{\omega_g}(\cdot)$ to transform $\omega_l$ and $\omega_o$ to the local frame defined by the micro-normal $\omega$. The multi-lobe BRDF keeps the orientations of the micro-normals, but neglects both shadowing-masking and interreflections caused by the micro-geometry.

Effective BRDFs. To include shadowing-masking effects, the effective BRDF $f_{\text{eff}}$ is used to describe the overall reflectance of a patch $P$ [Wu et al. 2011]. It depends on the micro-geometry $G(P)$ and the micro-BRDF $f$. The effective BRDF can be viewed as the average of the cosine-weighted and shadowed micro-BRDFs weighted by the visible projected area along the viewing direction $\omega_o$:

$$f_{\text{eff}}(\omega_l, \omega_o; G, f) = \frac{1}{A_G(\omega_o)} \int_P f(x_m(p), \omega_l, \omega_o)(\omega_m(p), \omega_l) V(x_m(p), \omega_l) A_G(p, \omega_o) k_p(p) \, dp,$$

(6)

where $V(x, \omega)$ denotes the binary visibility function (indicating whether a ray starting from point $x$ along direction $\omega$ is occluded), and $A_G(p, \omega_o)$ is the visible projected area along $\omega_o$ given by

$$A_G(p, \omega_o) = \langle \omega_o, \omega_m(p) \rangle V(x_m(p), \omega_o).$$

(7)

The total visible projected area of $G(P)$ is the normalization factor of the weighted average [Dupuy et al. 2013; Heitz 2014].

$$A_G(\omega_o) = \int_P \langle \omega_o, \omega_m(p) \rangle V(x_m(p), \omega_o) k_p(p) \, dp = \langle \omega_o, \omega_m \rangle.$$

(8)

The effective BRDF (6) has the cosine term and the visibilities baked in and captures the shadowing-masking caused by micro-scale self-occlusions. We will generalize the formulation in §5.1 to capture interreflections.

4 PREFILTERING REFLECTANCE PARAMETERS

Our method takes as input a high-resolution displacement map $h_{\text{orig}}$ and an isotropic base micro-BRDF $f_{\text{orig}}$ (which could be spatially varying, e.g., Figure 14). We denote $G_{\text{orig}}(P)$ as the micro-geometry defined by the original displacement map on a surface patch $P$. Then, we prefILTER the input model and obtain a lower-resolution displacement map $h_{\text{low}}$ (with $G_{\text{low}}(P)$ representing its micro-geometry on $P$) associated with a new spatially varying BRDF $f_{\text{low}}$. Our goal is to have the appearance of the prefiltered model closely resemble the input. Notice that, even if the input base BRDF $f_{\text{orig}}$ is spatially invariant, the prefiltered reflectance $f_{\text{low}}$ may need to have spatial variations to accurately reproduce the detailed appearance of the input model.

To this end, our technique starts with computing the downsampled displacement map $h_{\text{low}}$ by minimizing differences of the mesoscale slopes (§4.1). Then, we seek a prefiltered SVBRDF $f_{\text{low}}$ that preserves the original appearance for the downsampled displacement map. This, however, is nontrivial as $f_{\text{low}}$ generally needs to be spatially varying. We introduce a novel two-step approach to compute $f_{\text{low}}$. First, for each base patch of the downsampled displacement map $h_{\text{low}}$, we compute its corresponding patch NDF from the original displacement map $h_{\text{orig}}$. Each patch NDF implies a spatially varying multi-lobe BRDF without considering shadowing-masking and interreflections. This step generates a spatially varying base BRDF $f_{\text{low}}^{\text{init}}$ as an initial solution (§4.2). Then, we scale $f_{\text{low}}^{\text{init}}$ by a 6D scaling function $R(x, \omega_l, \omega_o)$ to match the effective BRDFs (§4.3). The final prefiltered SVBRDF $f_{\text{low}} = R \cdot f_{\text{low}}^{\text{init}}$ is able to reproduce the original appearance. Our method is also illustrated in Figure 3.

Although the scaling function $R(x, \omega_l, \omega_o)$ is 6D, we will show in §6 that $R$ can be factorized into a 2D spatial function of location $x$ and a 4D angular function of directions $\omega_l$ and $\omega_o$. Further, these functions are usually smooth in practice, and therefore can be computed efficiently as low-resolution tabulated functions, allowing our prefiltered models to be compactly represented.

4.1 Downsampling Displacement Maps

As stated in §3, displacement maps are defined mathematically as continuous 2D scalar functions. In practice, we represent displacement maps as piecewise linear functions using 2D textures. Specifically, the original high-resolution displacement map $h_{\text{orig}}$ is defined at each vertex of a dense grid: $(u \cdot 2^{-l} \cdot v \cdot 2^{-l})$ for $u, v = 0, \ldots, 2^l$. The prefiltered low-resolution displacement map $h_{\text{low}}$ uses a coarse grid $(u \cdot 2^{l'} \cdot v \cdot 2^{l'})$ for $u, v = 0, \ldots, 2^{l'}$ with some $l' < l$.

Obtaining $h_{\text{low}}$ requires specifying the displacement values at the vertices of the coarse grid. For every patch $P_{uv} = [u\cdot 2^{-l'} \cdot v \cdot 2^{-l'}] \times [v \cdot 2^{-l'} \cdot (u+1) \cdot 2^{-l'}]$ covering a grid cell of $h_{\text{low}}$, our goal is to find out the optimal heights at its four corners (i.e., grid points) such that the average slope $s_{\text{uv}}$ of the bilinear patch closely matches the reference slope $s_{\text{uv}}^{\text{orig}}$ of the original micro-surface $G_{\text{orig}}(P_{uv})$. The reference average slope $s_{\text{uv}}^{\text{orig}}$ can be calculated using Eq. (1). On the other hand, the average slope of the bilinear patch $P_{uv}$ is

$$s_{\text{uv}} = \frac{h_{11} + h_{10} - h_{01} - h_{00}}{2}, \quad s_{\text{uv}}^{\text{orig}} = \frac{h_{11} + h_{10} - h_{01} - h_{00}}{2},$$

(9)

where $h_{00}, h_{10}, h_{11}$ denote the displacement values at the four cell corners. Please refer to Appendix A for the derivation. To minimize the differences between the two average slopes robustly, we further introduce a regularization term for local smoothness, yielding the final objective function:

$$L(h_{\text{low}}) = \sum_u \sum_v \left( \frac{\|s_{\text{uv}} - s_{\text{uv}}^{\text{orig}}\|^2}{w} + \Delta h_{\text{low}} \left( \frac{u}{2^l} - \frac{v}{2^l} \right) \right)^2,$$

(10)

where $\Delta$ is the Laplacian operator and $w = 0.01$ is the weight of the regularization term. By minimizing this objective function using least-squares, the optimal solution gives the downsampled displacement map $h_{\text{low}}$ and its corresponding micro-geometry $G_{\text{low}}$.

4.2 Spatially Varying Multi-Lobe BRDF

For a cell from the coarse grid of the downsampled displacement map $h_{\text{low}}$ and its corresponding patch $P_{uv}$, the patch NDF $D_{uv}(\omega)$ defined in Eq. (2) provides a compact approximation of the original micro-geometry $G_{\text{orig}}(P_{uv})$. Previous works such as LEADR [Dupuy et al. 2013] typically assume the NDFs to follow Beckmann distributions. Our method, in contrast, does not enforce any restriction on the NDFs and is therefore more general. We use the normal mapping technique [Han et al. 2007] to fit an NDF with a mixture of vMF lobes (4), resulting in a spatially varying NDF at a lower resolution. The SVNDF is technically 4D but can be described compactly using spatially varying vMF parameters (stored as 2D textures).

From these NDFs that approximate $G_{\text{orig}}$ and the original base micro-BRDF $f_{\text{orig}}$, we use Eq. (5) to formulate the initial multi-lobe
BRDF for each patch $\mathcal{P}_{uv}$:
\[ f'_{\text{low}}(x, \omega_i, \omega_o) = \rho(\omega_i, \omega_o) \cdot D_{uv}(x, \omega_o). \]  
Here the micro-position $x$ is within the patch $\mathcal{P}_{uv}$.

4.3 Scaling Function

Although the multi-lobe SVBRDF $f'_{\text{low}}$ is a good start for matching the input appearance, it is incomplete as it neglects the shadowing, masking and interreflection effects. To address this problem, we first consider the single-bounce case (i.e., direct illumination that only involves shadowing and masking). We will discuss the multiple-bounce case that handles interreflections in §5.

To capture shadowing and masking, we multiply the initial multi-lobe SVBRDF $f'_{\text{low}}$ with another scaling function $R$. Suppose $R$ has a spatial resolution of $M^2$, we uniformly subdivide the base surface into $M^2$ patches. Let $\mathcal{P}_x$ denote the patch containing $x$. We define the scaling function $R(x, \omega_i, \omega_o)$ as the ratio\(^2\) between the original effective BRDF (6) and the prefiltered effective BRDF over $\mathcal{P}_x$:
\[ R(x, \omega_i, \omega_o) = \frac{f_{\text{eff}}(\omega_i, \omega_o; \mathcal{G}_{\text{orig}}(\mathcal{P}_x), f_{\text{orig}})}{f_{\text{eff}}(\omega_i, \omega_o; \mathcal{G}_{\text{low}}(\mathcal{P}_x), f_{\text{low}})}, \]  
(12)

The final prefiltered base SVBRDF can then be expressed as
\[ f_{\text{low}}(x, \omega_i, \omega_o) = R(x, \omega_i, \omega_o) f'_{\text{low}}(x, \omega_i, \omega_o). \]  
(13)

It is easy to verify that the final effective BRDF $f_{\text{eff}}(\mathcal{G}_{\text{low}}, f_{\text{low}})$ (with $\omega_i$ and $\omega_o$ omitted for notational simplicity) over $\mathcal{P}_x$ equals the original effective BRDF, since the single-bounce effective BRDF is linear in the scaling factors, i.e., $f_{\text{eff}}(\mathcal{G}_{\text{low}}, R \cdot f_{\text{low}}) = R \cdot f_{\text{eff}}(\mathcal{G}_{\text{low}}, f_{\text{low}})$.

The identical effective BRDFs indicate a good match of aggregate reflectance between the original and the prefiltered models.

Due to the high dimensionality of the scaling function $R$, it is challenging to obtain and store. We will show in §6 that $R$ can be accurately approximated as the product of a function $T(x)$ of location and a function $S(\omega_i, \omega_o)$ of directions:
\[ R(x, \omega_i, \omega_o) = T(x) \cdot S(\omega_i, \omega_o). \]  
(14)

We first generalize our scaling function to interreflections in §5. Then, in §6, we develop the factorization above and show that the factorized scaling functions $T$ and $S$ can be computed directly without fully computing the 6D function $R$.

5 INTERREFLECTIONS

We now describe how to capture interreflection effects using a scaling function similar to Eq. (12). We first generalize the effective BRDF formulation (6) to measure the average surface reflectance with interreflections (§5.1). Then, we provide an approximate solution to the scaling function with interreflections handled (§5.2).

5.1 Effective BRDF with Interreflections

The effective BRDF formulation depicted in §3 models only direct (i.e., single-bounce) illumination of the micro-geometry. This leads to significant energy loss due to the neglect of interreflections within the micro-geometry. To address this problem, we adapt Veach’s path integral formulation [1997] to express effective BRDFs with interreflections.

A ray that bounces within the micro-geometry $\mathcal{G} = \{(x_m, \omega_m)\}$ can be described by a light transport path $\tilde{x} = (x_1, x_2, \ldots, x_k)$ as well as the viewing direction $\omega_o$ and the lighting direction $\omega_i$. Notice that, unlike the traditional formulation where the endpoints of a path respectively lay on the light source and the sensor, all the path vertices $x_1, \ldots, x_k$ are located on the micro-surface in our case. The path contribution $h(\tilde{x})$ is given by the product of BRDF terms and cosine terms, followed by a visibility term at the last path vertex $x_k$ along the lighting direction $\omega_i$:
\[ h(\tilde{x}) = V(x_k, \omega_o) \prod_{j=1}^{k} f(x_j, \omega_{j\rightarrow j+1}, \omega_{j\rightarrow j-1}) \cdot n(x_j), \omega_{j\rightarrow j+1}). \]  
(15)

where $\omega_{j\rightarrow j} := \frac{x_j - x_{j-1}}{||x_j - x_{j-1}||}$ indicates the normalized direction from $x_i$ to $x_j$ (with $\omega_{i\rightarrow 0} := \omega_o$ and $\omega_{k\rightarrow k+1} := \omega_i$), and $n(x_j)$ denotes the micro-normal at $x_j$.

We define the path space $\Omega(x, \omega_i, \omega_o)$ as a collection of light transport paths $\tilde{x} = (x_1, \ldots, x_k)$ with $x_1 = x$, and the viewing and illumination directions given by $\omega_o$ and $\omega_i$, respectively. At a given position $x_m(p) \in \mathcal{G}$, the reflectance including interreflections aggregates contributions from all the valid paths:
\[ r(x_m(p), \omega_i, \omega_o) = \int_{\Omega(x_m(p), \omega_i, \omega_o)} h(\tilde{x}) \, dp(\tilde{x}), \]  
(16)

where $dp(\tilde{x})$ denotes the path throughput measure, which is a product of solid angle measures on the path directions.

According to Eqs. (6) and (16), we express the multi-bounce effective BRDF as the average path contribution over $\mathcal{G}$ weighted by the visible projected area along $\omega_o$:
\[ f'_{\text{low}}(\omega_i, \omega_o; \mathcal{G}, f) = \frac{1}{\mathcal{A}(\omega_o)} \int_{\mathcal{P}} r(x_m(p), \omega_i, \omega_o) \cdot A_G(p, \omega_o) \cdot k(p) \, dp. \]  
(17)

Under this formulation, the single-bounce effective BRDF (6) becomes a special case where each path $\tilde{x}$ is restricted to contain only one vertex.
5.2 Computing the Scaling Function

Following a similar idea as §4.3, we scale the initial multi-lobe SVBRDF $f'_{\text{low}}$ by a scaling function $R_{\text{ir}}$:

$$f_{\text{low}}(x, \omega_i, \omega_o) = R_{\text{ir}}(x, \omega_i, \omega_o) f'_{\text{low}}(x, \omega_i, \omega_o),$$

so that the resulting multi-bounce effective BRDF (17) of the prefiltered model matches that of the input. Namely, $f_{\text{ir}}^{\text{eff}}(G_{\text{low}}, f_{\text{low}}) = f_{\text{ir}}^{\text{eff}}(G_{\text{orig}}, f_{\text{orig}})$. Since $f_{\text{ir}}^{\text{eff}}$ is nonlinear in $R_{\text{ir}}$ due to multiple scattering (i.e., $f_{\text{ir}}^{\text{eff}}(G_{\text{low}}, R_{\text{ir}} \cdot f_{\text{low}}' \neq R_{\text{ir}} \cdot f_{\text{ir}}^{\text{eff}}(G_{\text{low}}, f_{\text{low}}')$ in general), $R_{\text{ir}}$ cannot be obtained using Eq. (12). Although one can optimize $R_{\text{ir}}$ iteratively using inverse rendering methods (e.g., [Khungurn et al. 2015; Zhao et al. 2016]), the optimization would be very expensive, if not impractical, due to the large number of unknowns involved.

Instead, we seek $R_{\text{ir}}$ without performing global optimizations. To this end, we make a simplifying assumption: the high-order components in $f_{\text{ir}}^{\text{eff}}(G_{\text{low}}, f_{\text{low}})$ are negligible. In other words,

$$f_{\text{ir}}^{\text{eff}}(G_{\text{low}}, f_{\text{low}}) \approx f_{\text{ir}}^{\text{eff}}(G_{\text{low}}, f_{\text{low}}').$$

This is because the high-order component mainly consists of distant interreflections (i.e., those across multiple surface patches at the reduced resolution) that are supposed to be handled with explicit path tracing (see the blue arrows in Figure 5). The local interreflection, on the other hand, is usually weakened due to the downsampling of an object’s micro-geometry, which we compensate using the scaling function $R_{\text{ir}}$ (see the red arrows in Figure 5).

Under this formulation, our goal becomes to match $f_{\text{ir}}^{\text{eff}}(G_{\text{orig}}, f_{\text{orig}})$ and $R_{\text{ir}} \cdot f_{\text{ir}}^{\text{eff}}(G_{\text{low}}, f_{\text{low}}')$, yielding the scaling function:

$$R_{\text{ir}}(x, \omega_i, \omega_o) = \frac{f_{\text{ir}}^{\text{eff}}(\omega_i, \omega_o, G_{\text{orig}}(\mathbf{p}_x), f_{\text{orig}})}{f_{\text{ir}}^{\text{eff}}(\omega_i, \omega_o, G_{\text{low}}(\mathbf{p}_x), f_{\text{low}}')}$$

In §8, we show that the simplification usually has minimal affect on accuracy (Figures 17, 18).

6 PROPERTIES OF THE SCALING FUNCTION

Upon establishing the scaling function $R_{\text{ir}}$ (20), our problem of prefiltering displacement-mapped surfaces boils down to efficiently representing and computing this function, which, unfortunately, is nontrivial as $R_{\text{ir}}$ is 6D and generally too expensive to compute and store in brute-force ways.

We tackle this challenge by factorizing the 6D scaling function $R_{\text{ir}}(x, \omega_i, \omega_o)$ into a single spatial scaling function $T_{\text{ir}}(x)$ and a single angular scaling function $S_{\text{ir}}(\omega_i, \omega_o)$. In practice, both $T_{\text{ir}}$ and $S_{\text{ir}}$ are smooth (i.e., low-frequency). We exploit their smoothness to achieve efficient factorization using only a sparse sampling of $R_{\text{ir}}$ (without explicitly precomputing the full 6D function). In the rest of this section, we provide more details on our factorization method.

**Rank-1 factorization.** Our experiments indicate that the scaling functions are typically low-rank (Figure 6). To demonstrate this property, we compute the full 6D scaling functions for two example models: “silk” with only direct illumination; and “twill” with global illumination (details for these models are described in §8.2). In both cases, the scaling matrices $R_{\text{ir}}$ are tabulated with a resolution $16^3 \times 16^3$.
increasing the spatial resolution has diminishing benefits. This is because the scaling function varies in a very smooth fashion spatially. We thus use $M = 4$ in most of our results (unless we state explicitly). In certain cases, even using $M = 1$ (that causes the scaling function $T_{ir}$ to be constant-valued) can offer sufficient accuracy.

**Efficient factorization.** Although computing the full 6D scaling function $R_{ir}$ and decomposing it using SVD provides optimal factorization results, doing so is very costly since both steps require massive computation. We introduce a simple and efficient factorization method that constructs $T_{ir}(x)$ and $S_{ir}(\omega_i, \omega_o)$ from sparse samples of $R_{ir}$.

According to Eq. (21), $T_{ir}(x)$ can be computed by averaging the 6D scaling function $R_{ir}$ over the angular domain:

$$ T_{ir}(x) = \frac{1}{C} \int_{\mathcal{H}^2} \int_{\mathcal{H}^2} R_{ir}(x, \omega_i, \omega_o) \, d\omega_i \, d\omega_o, \quad (22) $$

where

$$ C = \int_{\mathcal{P}_{all}} \int_{\mathcal{H}^2} \int_{\mathcal{H}^2} R_{ir}(x_m(p), \omega_i, \omega_o) k_{\mathcal{P}_{all}}(p) \, d\omega_i \, d\omega_o \, dp. \quad (23) $$

is a normalization factor, and $\mathcal{P}_{all} = [0, 1]^2$ covers the whole displacement map. To obtain $T_{ir}$, as a tabulated function, we estimate its value at each bin using Monte Carlo integration based on a sparse (joint) sampling of $x$ (among all locations within the bin), $\omega_i$, and $\omega_o$. We provide more implementation details in §7.

On the other hand, the angular scaling function $S_{ir}(\omega_i, \omega_o)$ can be obtained by averaging $R_{ir}$ spatially:

$$ S_{ir}(\omega_i, \omega_o) = \int_{\mathcal{P}_{all}} R_{ir}(x_m(p), \omega_i, \omega_o) k_{\mathcal{P}_{all}}(p) \, dp. \quad (24) $$

As a tabulated function, $S_{ir}$ can be computed in a similar way as $T_{ir}$ using Monte Carlo integration.

It is easy to verify that our definitions of $T_{ir}$ and $S_{ir}$ in Eqs. (22, 24) are consistent with Eq. (21). Please see Appendix B for more details.

Figure 10 compares factorization results obtained using our method and SVD (that uses full $R_{ir}$). Our method offers similar reconstruction errors as SVD does, which is demonstrated in Figure 10(b, c). Further, the corresponding renderings shown in Figure 10(d–f) indicate that the loss in visual quality caused by our method is negligible.

7 IMPLEMENTATION

We now provide details of our implementation. First, we provide details on how our technique works at individual scales (§7.1). Then, we describe how mipmaps of our prefiltered models can be created, enabling anti-aliased level-of-detail (LoD) rendering of displaced-mapped models (§7.2).

7.1 Prefiltering at a Single Scale

Provided a high-resolution displacement map $h_{orig}$ and a downsampling scale, we first optimize for the downsampled displacement map using least-squares (§4.1) and compute the initial spatially varying multi-lobe BRDF $f'_{low}$ (§4.2). As in prior work [Han et al. 2007], we use six vMF lobes per patch NDF. Then, we seek the scaling function $R_{ir}$ so that the final SVBRDF $f_{low} = R_{ir} \cdot f'_{low}$ preserves the appearance of the input model (§4.3 and §5.2). This is achieved using our efficient factorization of $R_{ir}$ (§6).

To estimate the value of $T_{ir}$ at each tabulation bin, we use 5000 pairs of $(\omega_i, \omega_o)$ with both directions independently sampled from a cosine-weighted distribution. For evaluating the scaling function $T_{ir}$ via Eq. (20), we trace 2500 paths to estimate the effective BRDFs (17). Lastly, we normalize $T_{ir}$ such that it averages to one. To estimate the scaling function $S_{ir}$ of directions, we use 16 stratified samples for $p$ and 2500 light transport paths for $R_{ir}$. We find these sampling rates are sufficient to reproduce accurate appearances.

We implement our method based on the Mitsuba physically based renderer [Jakob 2010]. Running on a workstation equipped with a six-core Intel i7-5930K CPU, computing the factorized functions $T_{ir}$ and $S_{ir}$ takes 10–30 minutes for each input model.

**Shell mapping.** We use shell mapping [Porumbescu et al. 2005] to deform a flat displacement map to a general shape (Figure 11), instead of explicitly displacing a base mesh. In fact, objects in our scenes are perturbed by extreme high-resolution displacement maps (resolution up to $(200K)^2$, tiled by $(1K)^2$ base displacement maps) to describe very small geometric details. Creating an explicit displacement mesh is not practical.

**Importance sampling.** Our final prefiltered SVBRDF $f_{low}$ equals to the product of two functions $R_{ir}$ and $f'_{low}$. Due to the smoothness of the scaling function $T_{ir}$, we simply follow $f'_{low}$ when importance sampling $f_{low}$. That said, product importance sampling [Clarberg et al. 2005; Herholz et al. 2016] could be used to further improve the sampling quality.

7.2 Level of Detail

To enable anti-aliased rendering at multiple scales, we prefiltter the input model at several downsampling scales. The resulting models effectively form a mipmap. At each mipmap level, we compute and store the lobe parameters as well as the scaling functions $T_{ir}$ and $S_{ir}$, in addition to the downsampled displacement map. To balance computational and storage overhead and rendering performance, our mipmap uses a sparser set of levels compared to conventional texture mipmaps. In other words, the scales of two contiguous levels of our mipmaps usually differ by more than 2x. For example, the mipmap of the “Silk” model includes $(16x)^2$- and $(64x)^2$-downsampled versions (besides the original model), and the mipmap of “Twill” has $(8x)^2$ and $(64x)^2$-downsampled versions.

At render time, we use ray differentials [Igehy 1999] to compute each pixel’s footprint on the original displacement map and determine the desired mipmap levels. To interpolate between two levels, we trace paths on two different models and linearly interpolate the path contributions. Please refer to Figures 1, 12 and the accompanying video for results rendered with this method.

8 RESULTS

We now show results generated using our method: §8.1 contains experimental evaluations and justifications; §8.2 demonstrates the effectiveness of our technique via a few examples. Finally, we discuss limitations of our methods and future research directions (§8.3).

8.1 Evaluations and Justifications

**Energy Conservation.** We demonstrate that our prefiltered models conserve energy via white furnace tests (Figure 13). Given a
Fig. 8. **Using varying angular resolutions.** Increasing angular resolution improves rendering accuracy, but at the expense of larger computational and storage overhead (with a fixed spatial resolution of $M = 1$). We use a resolution of $N = 15$, which shows the best tradeoff between accuracy and storage/performance. The insets (and those in the next figures 9, 10(e, f)) show 10× squared error compared to the reference images.

Fig. 9. **Using varying spatial resolutions.** Increasing spatial resolutions results in more accurate rendering results, but the improvement is diminishing after $M = 4$ (errors may become slightly larger because of Monte Carlo noise). We find that $M = 4$ is adequate in most cases, and even using $M = 1$ yields high-quality results. Bottom-right insets visualize the 2D spatial scaling functions for different resolutions.

non-absorptive base BRDF and a constant environment light, our prefiltered models without interreflections (Figure 13(b)) lose a significant amount of energy, while those capturing interreflections (Figure 13(c)) preserve most of the energy. The effective albedos of our prefiltered models in Figure 13(c) are 0.988 (twill) and 0.999 (noise).

The ground-truth effective BRDFs are energy conserving as they are determined by explicitly simulating light transport within the micro-surfaces. Theoretically, our prefiltered models as approximations of the ground-truth effective BRDFs are not guaranteed to conserve energy. Limited energy loss is mainly due to neglecting distant interreflections. In practice, however, we do not observe any problem with energy conservation.

**Reciprocity.** Our effective BRDFs (17) are reciprocal when weighted by visible projected area (similar to the microflake reciprocity constraints proposed by Heitz et al. [2015]):

$$f_{\text{ir}}^{\text{eff}}(\omega_i, \omega_o; G, f) A_G(\omega_o) = f_{\text{ir}}^{\text{eff}}(\omega_o, \omega_i; G, f) A_G(\omega_i).$$

(25)

Since we aim to match the effective BRDFs given by the input models, our prefiltered models are approximately reciprocal.

**Correlated Surfaces.** Our prefiltering method can handle surfaces with correlations. Figure 14 shows a synthetic two-color sawtooth example where the surface normals and colors are strongly correlated. The baseline multi-lobe BRDFs $f'_{\text{low}}$ disregard this correlation and produce wrong appearance. Our result, on the other
Fig. 10. **Factorization comparison.** We compare our factorization method to rank-1 SVD in terms of scaling function reconstruction (a–c) and rendered images (d–f). We show 6D scaling functions (they are reorganized as scaling matrices) in (a). Even without explicitly computing the full 6D functions, the reconstruction error of our factorization (b) is similar to the error given by rank-1 SVD (c). Both errors are small because of the low-rank nature of the scaling matrices. The insets in (b, c) show the relative error maps of the reconstructed matrices compared to the reference matrices. The colored error bar is shown on the top-right. In terms of visual quality, using the factorized spatial scaling function $T$ and angular scaling function $S$ computed by our method reproduces the reference appearance accurately.

Fig. 11. **Shellmap illustration.** We deform a flat displacement map to a general shape using shell maps.

Fig. 12. **LoD rendering.** The reference (a) is rendered using a high-resolution displacement map and 5K spp. The baseline result (b) is rendered using the same displacement map but (10x)-lower spp. Our equal-time low-spp LoD rendering (c) closely matches the reference (a) at multiple scales, and suffers from less noise (measured using Liu et al. [2012], lower is better) compared to the baseline (b). The mipmap levels are shown in inset.

Hand, captures this correlation properly and preserves the reference appearance accurately.

8.2 Main Results

We use two sets of scenes to demonstrate the effectiveness of our method. The scene objects and their detailed parameters (before and after prefiltering) are summarized in Table 3. The first set contains two fabric examples (Figures 2, 16, 17). The “twill” and “silk” models (Figure 15(a, b)) are extracted from detailed volume data [Zhao et al. 2011]. The resulting surface micro-geometries are highly complex and can cause severe aliasing. The “plane” scene in Figure 2 contains a flat twill fabric. The “twill” scene in Figures 16, 17 (top row) uses the same fabric but shell-mapped. The “silk” scene in Figures 16, 17 (bottom row) has another shell-mapped silk fabric. Although we
use isotropic glossy BRDFs in these scenes, the rendered images still show anisotropic glossy highlights because of the complex structured micro-geometries.

Another set of scenes contains objects with more diverse materials. In Figure 18 (top row), we show the "ball" scene that has a bumpy ball on a rough floor. The ball is represented by the "bump" model (Figure 15(c)), while the floor is represented by the "noise" model (Figure 15(d)). Both models use diffuse base BRDFs. In Figure 18 (middle row), we show the "logo" scene that is represented by a bi-scale displacement map (Figure 15(e)). The large-scale displacement map describes the overall shape, while the small-scale displacement map (which is a tiled version of the large-scale one) accounts for the detailed micro-geometry. In Figure 18 (bottom row), we show the "relief" scene that is represented by a composite displacement map (Figure 15(f), which is resampled to keep the image aspect ratio). It is constructed by adding procedural Perlin noise on a height field converted from a gray scale image. In this case, we use a spatial scaling function with resolution $M = 1$ since it leads to sufficient accuracy.

**Direct Illumination.** Figure 16 contains rendered images generated through the intermediate steps of our method with the effective BRDFs capturing only direct illumination (i.e., neglecting micro-scale interreflections). Directly using original BRDFs $f_{\text{orig}}$ with downsampled micro-geometries (the third column) leads to significantly different appearances. Using the spatially varying multi-lobe BRDF $f'_{\text{low}}$ without our scaling function $R = ST$ (the fourth column) performs better, but the results are still notably brighter than the references as changes of shadowing and masking are neglected. By applying a further correction using $R = ST$, our models well preserve the input appearances. Additionally, we compare our prefiltered models with those generated using LEADR [Dupuy et al. 2013], which are derived from single-lobe Beckmann NDFs and Smith’s shadowing function. These models fail to reproduce the appearance of the input since the assumption of Gaussian surfaces does not hold (see the top insets in Figure 2(a, f)). Further, the single lobe representation used by LEADR also limits the result accuracy, especially for glossy base BRDFs.

**Global Illumination.** The results in Figures 17 and 18 are rendered with full global illumination. In this case, directly using $f_{\text{orig}}$ (the third column) still yields poor accuracy. Using multi-lobe BRDFs $f'_{\text{low}}$ improves the glossy highlights quality but cannot fully capture the change of interreflections. Our full model with $f'_{\text{low}}$ and the scaling functions $R_{\text{ir}} = S_{\text{ir}} T_{\text{ir}}$ resembles the input appearance accurately. To demonstrate the effectiveness and generality of our technique, we also show rendering results under different lighting and viewing configurations in Figure 17 and the accompanying video.

**Performance and Storage.** Table 3 summarizes the storage size of our prefiltered models and Table 4 shows equal-time mean squared error (MSE) comparisons. Our prefiltered models can offer significant storage reduction at a single downscaling scale. Breaking down individual components of our model, the multi-lobe SYNDFs are most storage-consuming (except "noise" and "logo" as they are greatly downsampled) while our scaling functions $T$ and $S$ only introduce minimal overhead.

Our model also benefits rendering performance by reducing ray tracing cost. We demonstrate this by rendering a few high-resolution scenes and their prefiltered counterparts (generated with our method) in equal-time and compare MSEs of the results (see Table 4). Using our prefiltered models leads to lower errors in all the scenes and anti-aliased renderings with better visual qualities (see Figure 2(c, e)). In the accompanying video, we demonstrate that our prefiltered models preserve input appearance accurately and reduce flickering significantly.

8.3 Limitations and Future Work

Figure 19 shows a failure example where our method does not accurately reproduce the original appearance caused by interreflections. This "statue" scene contains a $(4K)^3$ displacement map and a moderately glossy base BRDF. We downsample it by $(16\times)^3$ and use a spatial scaling function of resolution $M = 1$. Compared to the "relief" scene, the vertical displacements in this scene are much greater. In this case, distant interreflections can only be partially handled by explicit path tracing. Since our method neglects these interreflections (for enabling efficient computation of $S_{\text{ir}}$ and $T_{\text{ir}}$), the results have visible accuracy loss shown as darkening at the steep edges (see (b1) and (b2)). On the other hand, although our method produces inaccurate edge appearance, the overall surface reflectance (i.e. average image intensity) still matches the reference because the two effective BRDFs are matched by our scaling functions. For direct illumination results ((a1) and (a2)), the accuracy is not affected.

Since our method considers micro-geometries and base BRDFs jointly, which is necessary for accurately handling complex light transport effects such as shadowing-masking and interreflections, changing the micro-geometry or base BRDF requires recomputing our prefiltered models. In the future, our technique may be combined with material editing techniques such as Hasan and Ramamoorthi [2013] to allow changing the base BRDF with a single precomputation.

Interreflection is an effect that is well known to be challenging to analyze due to its high nonlinearity. A theoretical or data-driven analysis can be an interesting topic to explore. To balance accuracy and efficiency, we use relatively low resolutions for the spatial and
Our prefiltered model glossy (Tiling 2724 589 7 1) Resolution ∼ 200 km2 glossy 1915 7 2377 3502 24 7619 3954 484 (52x323) while LEADR does not perform well even for direct illumination and is not designed for interreflections (and is therefore not shown here).

We provide only parameters of the small-scale displacement map, since the large-scale one remains unchanged when downsampling.

We assume the base BRDFs to be isotropic because the anisotropic micro-geometries are capable of reproducing anisotropic appearance. Using anisotropic base BRDFs would increase the model expressiveness but at the price of more expensive precomputation. We leave this extension for future exploration.

Fig. 16. Direct illumination results. Our method accurately matches the reference even with significant downsampling, and is much more accurate than naive downsampling, multi-lobe BRDFs without taking shadowing-masking or interreflections into account, and LEADR. The insets (and those in the next figures 17, 18) show 10x squared errors compared to the reference images.

Fig. 17. Global illumination results. The four columns on the left are rendered under local area lights, while the four columns on the right are rendered under environmental lightings (environment maps are visualized as insets of the reference images) from a different view. Our method is able to capture interreflections well, closely matching the reference. Naive downsampling and the multi-lobe BRDFs do not model interreflections and have significant errors, while LEADR does not perform well even for direct illumination and is not designed for interreflections (and is therefore not shown here).

Table 3. Model Information. We list the detailed parameters used for prefiltering the input model at a single downsampling scale. We also provide the data size of our prefiltered models and their corresponding precomputation time.

<table>
<thead>
<tr>
<th>Name</th>
<th>Resolution</th>
<th>Tiling</th>
<th>Total size (KB)</th>
<th>Base BRDF</th>
<th>Downsampling scale</th>
<th>Total size (KB)</th>
<th>hlow (KB)</th>
<th>f′low (KB)</th>
<th>T/Tir (KB)</th>
<th>S/Sir (KB)</th>
<th>Total time (min)</th>
<th>hlow and f′low (min)</th>
<th>T/Tir (min)</th>
<th>S/Sir (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twill</td>
<td>(1K)2</td>
<td>1002</td>
<td>3502</td>
<td>glossy</td>
<td>(8x)2</td>
<td>2377</td>
<td>65</td>
<td>1915</td>
<td>1</td>
<td>396</td>
<td>22</td>
<td>1</td>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>Silk</td>
<td>(1K)2</td>
<td>2002</td>
<td>2724</td>
<td>glossy</td>
<td>(16x)2</td>
<td>1085</td>
<td>17</td>
<td>478</td>
<td>1</td>
<td>589</td>
<td>22</td>
<td>1</td>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>Bump</td>
<td>(1K)2</td>
<td>42</td>
<td>4015</td>
<td>diffuse</td>
<td>(16x)2</td>
<td>704</td>
<td>17</td>
<td>484</td>
<td>1</td>
<td>202</td>
<td>24</td>
<td>1</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>Noise</td>
<td>(1K)2</td>
<td>1002</td>
<td>3954</td>
<td>diffuse</td>
<td>(64x)2</td>
<td>239</td>
<td>2</td>
<td>34</td>
<td>1</td>
<td>202</td>
<td>22</td>
<td>1</td>
<td>4</td>
<td>17</td>
</tr>
<tr>
<td>Logo6</td>
<td>(1K)2</td>
<td>(1K)2</td>
<td>723</td>
<td>glossy</td>
<td>(1K)2</td>
<td>589</td>
<td>~0</td>
<td>2</td>
<td>1</td>
<td>586</td>
<td>36</td>
<td>1</td>
<td>12</td>
<td>23</td>
</tr>
<tr>
<td>Relief</td>
<td>(4K)2</td>
<td>1</td>
<td>33114</td>
<td>glossy</td>
<td>(16x)2</td>
<td>8367</td>
<td>238</td>
<td>7619</td>
<td>~0 (M=1)</td>
<td>510</td>
<td>26</td>
<td>1</td>
<td>0</td>
<td>25</td>
</tr>
</tbody>
</table>

angular scaling functions empirically. We leave how to determine the optimal resolutions as future work.

We assume the base BRDFs to be isotropic because the anisotropic micro-geometries are capable of reproducing anisotropic appearance. Using anisotropic base BRDFs would increase the model expressiveness but at the price of more expensive precomputation. We leave this extension for future exploration.
Table 4. Equal-time performance. Our prefiltered models not only reduce the storage size, but also bring benefits in rendering. Given equal time (around 60s for 50–100 spp), rendering using our models results in anti-aliased images, which have lower MSE than those rendered using the original models.

<table>
<thead>
<tr>
<th>Scene</th>
<th>Twill</th>
<th>Silk</th>
<th>Ball</th>
<th>Logo</th>
<th>Relief</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original model’s MSE (×10^{-3})</td>
<td>1.095</td>
<td>1.269</td>
<td>2.055</td>
<td>2.072</td>
<td>2.946</td>
</tr>
<tr>
<td>Our model’s MSE (×10^{-3})</td>
<td>0.571</td>
<td>0.622</td>
<td>1.850</td>
<td>0.498</td>
<td>1.692</td>
</tr>
</tbody>
</table>

Acknowledgments

We thank the anonymous reviewers for their constructive suggestions and comments. This work was supported in part by NSF grants 1451830 and 1703957, NSF IIS CHS-1813553, an NVIDIA fellowship, the Ronald L. Graham Chair, and the UC San Diego Center for Visual Computing.

References


ACM Trans. Graph., Vol. 38, No. 4, Article 1. Publication date: July 2019.
A  AVERAGE SLOPE OF A BILINEAR PATCH
Let $h_{00}, h_{10}, h_{01}, h_{11}$ be the height values at the four corners of a unit patch $[0, 1]^2$. Then, the height at any point $(u, v)$ within the patch can be obtained via a bilinear interpolation:

$$h(u, v) = (1 - u)(1 - v)h_{00} + u(1 - v)h_{10} + v(1 - u)h_{01} + uvh_{11}.$$  

The slope at each point can be computed as

$$x_3(u, v) = \frac{h(u + \Delta u, v) - h(u - \Delta u, v)}{2\Delta u} = h_{10} - h_{00} + v(h_{01} - h_{11} - h_{01} - h_{10}),$$

$$y_3(u, v) = \frac{h(u, v + \Delta v) - h(u, v - \Delta v)}{2\Delta v} = h_{01} - h_{00} + u(h_{01} + h_{11} - h_{01} - h_{10}).$$

So the average slope of the bilinear patch is

$$x_3 = \int_0^1 \int_0^1 x_3(u, v) \, du \, dv = \frac{1}{2}(h_{11} + h_{10} - h_{01} - h_{00}),$$

$$y_3 = \int_0^1 \int_0^1 y_3(u, v) \, du \, dv = \frac{1}{2}(h_{11} + h_{10} - h_{01} - h_{00}).$$

B  FACTORIZATION OF $R_{r_{ir}}$

Expanding $T_{ir}(x)$ as Eqs. (22, 23) and $S_{ir}(\omega_1, \omega_2)$ as Eq. (24) gives

$$T_{ir}(x) S_{ir}(\omega_1, \omega_2) = \int_{x_1}^x \int_{y_1}^y R_{ir}(x, \omega_1, \omega_2) \, dx_1 \, dy_1 \int_{\omega_1}^\omega \int_{\omega_2}^\omega \frac{dx_1 \, dy_1 \, d\omega_1 \, d\omega_2}{dp \int_{\omega_1}^\omega \int_{\omega_2}^\omega R_{ir}(x, \omega_1, \omega_2, \omega_3, \omega_4) \, d\omega_3 \, d\omega_4} \times \int_{\omega_3}^\omega \int_{\omega_4}^\omega \frac{dx_1 \, dy_1 \, d\omega_1 \, d\omega_2}{dp \\int_{\omega_1}^\omega \int_{\omega_2}^\omega S_{ir}(\omega_1, \omega_2, \omega_3, \omega_4) \, d\omega_3 \, d\omega_4} \, d\omega_3 \, d\omega_4.$$  

As $R_{ir}(x, \omega_1, \omega_2) \approx T_{ir}(x) \cdot S_{ir}(\omega_1, \omega_2)$, we can simplify this formula by factoring out $R_{ir}$ and regrouping the terms:

$$T_{ir}(x) S_{ir}(\omega_1, \omega_2) = \int_{x_1}^x \int_{y_1}^y T_{ir}(x) \, dx_1 \, dy_1 \int_{\omega_1}^\omega \int_{\omega_2}^\omega S_{ir}(\omega_1, \omega_2) \, d\omega_1 \, d\omega_2 \int_{\omega_1}^\omega \int_{\omega_2}^\omega \frac{dx_1 \, dy_1 \, d\omega_1 \, d\omega_2}{dp \int_{\omega_1}^\omega \int_{\omega_2}^\omega S_{ir}(\omega_1, \omega_2, \omega_3, \omega_4) \, d\omega_3 \, d\omega_4} \times \int_{\omega_3}^\omega \int_{\omega_4}^\omega \frac{dx_1 \, dy_1 \, d\omega_1 \, d\omega_2}{dp} \\int_{\omega_1}^\omega \int_{\omega_2}^\omega S_{ir}(\omega_1, \omega_2, \omega_3, \omega_4) \, d\omega_3 \, d\omega_4} \, d\omega_3 \, d\omega_4.$$  


