Eurographics Symposium on Rendering (2005)
Kavita Bala, Philip Dutré (Editors)

Multiresolution Reflectance Filtering

Ping Tan[1] † Stephen Lin[2] Long Quan[1] Baining Guo[2] Heung-Yeung Shum[2]

Hong Kong University of Science and Technology[1]
Microsoft Research, Asia[2]

Abstract

Physically-based reflectance models typically represent light scattering as a function of surface geometry at the pixel level. With changes in viewing resolution, the geometry imaged within a pixel can undergo significant variations that result in changing reflectance characteristics. To address these transformations, we present a multiresolution reflectance framework based on microfacet normal distributions within a pixel over different scales. Since these distributions must be efficiently determined with respect to resolution, they are recorded at multiple resolution levels in mipmaps. The main contribution of this work is a real-time mipmap filtering technique for these distribution-based parameters that not only provides smooth reflectance transitions in scale, but also minimizes aliasing. With this multiresolution reflectance technique, our system can rapidly and accurately incorporate fine reflectance detail that is customarily disregarded in multiresolution rendering methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Physically based modeling

1. Introduction

Various computer graphics techniques have been presented for rendering objects at different resolutions. For example, object geometry at different scales has been addressed by work in multiresolution meshes (e.g., [Hop96]), and color textures can be rendered at different levels of detail using mipmaps [Wil83]. These multiresolution representations serve the purpose of reducing run-time computation by approximating small-level details in a manner that maintains accurate appearance.

Although geometry and textures are often modelled at multiple scales, the reflectance properties of an object are customarily represented by a fixed bidirectional reflectance distribution function (BRDF). Reflectance characteristics such as shading and highlights, however, clearly depend upon the surface geometry imaged within a pixel, and typically the geometric characteristics at the pixel level change with scale, as shown in Fig. 1. As the viewer zooms out from a surface, fine-level geometric elements known as mesostructure reduce in scale into the pixel domain. Since changes in resolution can alter geometric characteristics at the pixel level as illustrated in Fig. 2, reflectance should also change accordingly.

To address this problem, we propose a technique for real-time rendering of multiresolution reflectance with respect to the geometric structure within each pixel. In representing pixel-level geometry, most physically-based reflectance models consider a surface to be composed of planar microfacets, and their surface normal directions are represented by a Gaussian distribution [CT81, APS00]. As seen in Fig. 2, a single Gaussian is often inadequate for modeling the more complex microfacet normal distributions of low resolution pixels which view a broader surface area, so a Gaussian mixture model (GMM) may be employed to more accurately convey multi-resolution variations in pixel geometry. Conventional microfacet-based reflectance models can be directly extended to compute reflectance from GMMs, and in our implementation, we utilize a combination of extended Lambertian and Cook-Torrance models [CT81] for the diffuse and specular components of reflectance, respectively.

The principal challenge of employing a multiresolution reflectance model is in how to efficiently and accurately determine the reflectance parameters at different scales. Since mesostructure geometry can vary over a surface, these reflectance parameters can be spatially variant as well. In obtaining GMM parameters at multiple resolutions, it is im-

† Correspondence email: ptan@cs.ust.hk

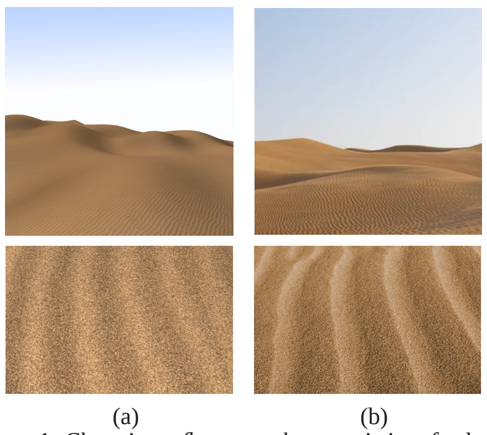

Figure 1: *Changing reflectance characteristics of a desert at different resolutions. (a) Rendering results with our method; (b) Real photographs captured at similar resolutions.*

practical to assemble and process the microfacets of each pixel at runtime, so we precompute and store them in mipmaps. However, since reflectance is a non-linear function of distribution-based GMM parameters, these GMM mipmaps cannot be trilinearly interpolated like color textures. Moreover, since reflectance is very sensitive to geometric information, filtering of GMM parameters must be carefully considered for aliasing to be minimized.

The primary contribution of our work is a method for constructing and filtering reflectance mipmaps that represent pixel geometry information. To promote accurate mipmap interpolation, we propose a Bayesian algorithm for prefiltering normal distributions such that a correspondence of distribution features is obtained. For corresponding features, we present a rapid filtering technique that capitalizes on hardware mipmapping functions to perform the non-linear filtering needed to optimally interpolate the distribution-based quantities. Though the formulation of the prefiltering algorithm is somewhat involved, the runtime mipmap interpolation and reflectance computation are simple and straightforward to implement. With this method, accurate scale-dependent reflectance effects with smooth transitions are rendered at high frame rates with negligible aliasing.

2. Previous Work

Numerous reflectance models have been proposed in computer graphics, and they have generally been designed for rendering BRDFs at a fixed resolution. Although some empirical models, such as spherical harmonics [WAT92], have the flexibility to represent multiresolution reflectance, we take a physically-based approach and focus on the reflectance effects of changing pixel geometry through multiple resolutions.

To account for the effect of pixel geometry on reflectance, Becker and Max [BM93] tabulate at rendering time the distribution of unoccluded normals from the projected displacement map in each pixel. For a substantial reduction

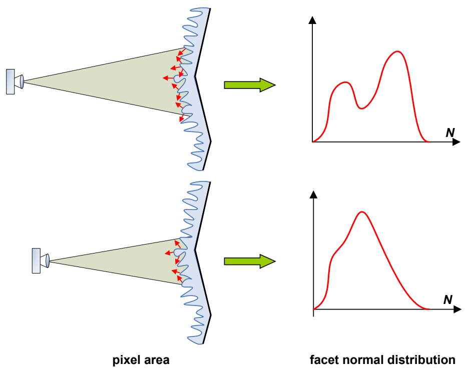

Figure 2: *Normal distributions at coarser resolution (top), and finer resolution (bottom). Distributions within a pixel change with respect to rendering resolution, which causes reflectance variations over different scales.*

in run-time computation, mipmaps of geometric attributes have been employed. For a mipmap of bump maps, Toksvig [Tok04] presents a method for adjusting surface roughness values over multiple scales based on a simple consistency measure of surface normal directions. In rendering complex volumetric structures at multiple scales, Neyret [Ney98] mipmapped intra-pixel geometry that was expressed in terms of an ellipsoid model of normal distributions.

At coarse resolutions, the geometry within a pixel can become so complicated that a more general model is needed for accurate representation. More important, the subsampling of normal distributions that effectively occurs with simplified models can lead to significant aliasing that is magnified in rendering due to the sensitivity of reflectance to surface normals, as illustrated in Fig. 4 (a). A high sampling rate is utilized in a method by Fournier [Fou92], which models reflectance as a set of seven Phong peaks per texel. In the mipmap filtering process, Phong peaks from interpolated texels are scaled and aggregated such that up to 56 peaks are used in rendering a pixel, which entails a non-trivial expense in computation. Similar to Fournier's work, we represent pixel geometry in terms of a GMM of normal directions. However, efficient filtering of these highly-sampled distribution-based parameters poses a challenging problem that was not addressed in [Fou92]. In this work, we successfully interpolate these reflectance mipmaps by using a proposed prefiltering technique for GMM mipmap construction and by non-linear filtering of the reflectance parameters in a manner that exploits linear hardware filters. With this approach, resolution dependencies of reflectance are modeled accurately and efficiently.

3. Multiresolution Reflectance Model

Microfacet normal distributions are typically represented by a single Gaussian lobe, which is suitable for micro-scale surface normal variations referred to as surface roughness [TS67]. At coarser resolutions where surface geometry perturbs the distribution of normals in a pixel, the single-Gaussian model becomes less valid, as illustrated in Fig. 2.

For a multiresolution representation, a more general descriptor is needed, so we employ a Gaussian mixture model (GMM) of normal distributions. GMMs have long been used for modeling general distributions, and the use of two Gaussians for representing normal distributions has been suggested previously [CT81]. Since a single Gaussian describes a micro-scale normal distribution at a single point, the distribution for a pixel that encompasses multiple such points can be expected to have a mixture of Gaussians form

$$G(\mathbf{n}) = \sum_{i=1}^{N} \alpha_i g(\mathbf{n}; \mu_i, \sigma_i),$$

where N is the number of Gaussians, $g(\cdot)$ is a Gaussian with mean normal vector $\mu_i = (\mu_i(x), \mu_i(y))$ and univariate standard deviation σ_i, and α_i are mixture weights. Considering surface geometry as a height field $z = f(x,y)$, a unit normal $\mathbf{n}$ can be represented as a 2D vector in x-y space.

For a GMM distribution of microfacet normals, the Lambertian and Cook-Torrance models can be directly extended by summing the reflectance values for each individual Gaussian. For the Cook-Torrance component, this sum can be expressed as

$$I_{CT} = \rho_s F(u) \sum_{i=1}^{N} \alpha_i \left[\frac{g(\mathbf{h}, \mu_i, \sigma_i)}{\mu_i \cdot \mathbf{v}} \right]$$

where ρ_s is the illumination color, $\mathbf{l}$ and $\mathbf{v}$ are lighting and viewing directions defined in the local coordinate frame, and $F(u)$ is the simplified Fresnel function of the material [Sch94] with respect to $u = max\{\mathbf{v} \cdot \mathbf{h}, 0\}$ where $\mathbf{h}$ denotes the bisector of $\mathbf{l}$ and $\mathbf{v}$. Similarly, the Lambertian component can be formulated for a sum of N Gaussians, whose mean normal vectors μ_i are used to represent surface normals:

$$I_{Lamb} = \rho_d (1 - F(u)) \sum_{i=1}^{N} \alpha_i [\mu_i \cdot \mathbf{l}] .$$

where ρ_d is the average albedo of the microfacets in a pixel as determined from a color mipmap. A scale dependent reflectance function that employs GMMs can then be expressed as a sum of these two components:

$$I = I_{CT} + I_{Lamb}.$$

4. Reflectance Mipmaps

For efficient rendering of the scale-dependent reflectance function, the GMM parameters need to be rapidly determined. To facilitate this computation, we precompute a reflectance mipmap for which the GMM parameters of each texel are computed according to normal samples drawn from the finest-resolution surface height field.

In filtering a reflectance mipmap, a GMM $\hat{\Theta} = \{\hat{g}_i := (\hat{\alpha}_i, \hat{\mu}_i, \hat{\sigma}_i); 1 \le i \le N\}$ needs to be accurately and efficiently interpolated from the K GMMs $\{\Theta_k; 1 \le k \le K\} = \{g_{ki} := (\alpha_{ki}, \mu_{ki}, \sigma_{ki}); 1 \le i \le N, 1 \le k \le K\}$ of the neighboring K texels. Although standard graphics hardware provides an efficient trilinear interpolation function, the distribution-based GMM parameters are non-linearly related to reflectance, and therefore cannot be accurately filtered by trilinear interpolation. As noted by [Fou92], K N-modal distributions may be interpolated by an NK-modal distribution, and all NK components should ideally be rendered and summed for accurate computation of reflectance. However, this accumulation of distribution components dramatically degrades rendering efficiency.

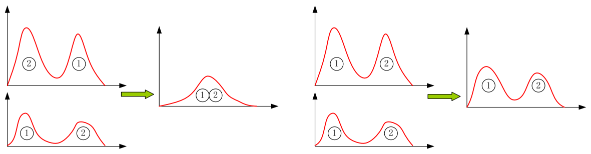

Figure 3: *Interpolation of unaligned (a) and aligned (b) GMMs. Gaussian components are numbered to show the correspondence. Interpolation takes place between Gaussian lobes with same index. While alignment maintains proper geometric characteristics after interpolation, misalignment results in lost detail and essentially a reduction in sampling rate.*

To address this problem, we propose a GMM preprocessing and filtering technique that accurately and efficiently computes interpolated GMMs that have the same number of Gaussian components as the GMMs of mipmap texels. For each Gaussian component $\hat{g}_i$ of the filtered GMM, we interpolate corresponding Gaussians $g_{ki}, 1 \le k \le K$, from the GMMs of the K neighboring texels. To elevate accuracy, we present an algorithm that computes mipmap GMMs in a manner that yields similar means for corresponding Gaussians g_{ki} among neighboring texels. For efficiently filtering this precomputed reflectance mipmap at rendering time, we describe a technique for optimal non-linear interpolation of corresponding GMM components that takes advantage of trilinear filters in standard graphics hardware.

4.1. GMM Alignment

Interpolating a set of K Gaussians by a single Gaussian can be accurate only when the centers of these Gaussians are closely located. To minimize interpolation error, it is therefore necessary for neighboring texels to have aligned GMMs such that corresponding Gaussian components have similar means. As illustrated in Fig. 3, if corresponding Gaussians have means that are distant from one another, the filtered GMM tends to be averaged towards a single lobe, effectively resulting in a downsampled representation that causes strong aliasing. To facilitate alignment of GMM components among neighboring mipmap texels, we propose a modification to the Expectation-Maximization (EM) algorithm [DLR77, Bil97] that solves for the texel GMMs in a manner that favors close alignment of neighboring GMM components.

With the traditional EM algorithm, the parameters Θ of an N-Gaussian GMM are computed according to normal samples $X = \{\mathbf{n}_j; 1 \le j \le M\}$ drawn from the original high-resolution height field, such that

$$\Theta = \arg\max P(X|\Theta) = \arg\max \Pi_{j=1}^{M} \left(\Sigma_{i=1}^{N} \alpha_i g(\mathbf{n}_j; \mu_i, \sigma_i) \right).$$

To globally align GMM components of neighboring texels, we introduce the following prior constraint on GMM parameters Θ with respect to the GMM parameters of neighboring texels $N(\Theta) = \{\Theta_k; 1 \le k \le K\}$:

$$P(\Theta|N(\Theta)) = \Pi_{k=1}^{K} P(\Theta|\Theta_k) = \Pi_{k=1}^{K}\Pi_{i=1}^{N} exp(-\alpha_{ki}||\mu_{ki}-\mu_i||^2)$$

$P(\Theta|\Theta_k)$ is at its largest when the corresponding means μ_i, μ_{ki} are identical, and maximizing it will favor neighboring GMMs with similar means for corresponding Gaussian components. Moreover, the penalty for discrepancies among Gaussian means $||\mu_{ki} - \mu_i||^2$ is scaled by α_{ki}, which allows a larger discrepancy for less important Gaussians with a smaller mixture weight. In some cases, the computed weight of a Gaussian component may be zero, $\alpha_{ki} = 0$, when the Gaussian does not contribute significantly to the GMM and does not correspond well with the neighboring GMMs. As such, the GMM parameters can be well defined by the following maximum a posterior (MAP) formula:

$$\begin{aligned} \Theta &= \arg\max P(\Theta|X) = \arg\max P(X|\Theta)P(\Theta|N(\Theta)) \\ &= \arg\max \left[log(P(X|\Theta)) + log(P(\Theta|N(\Theta)))\right] \end{aligned}$$

This posterior can be maximized by the Expectation-Maximization algorithm [Bil97]. For each texel in the mipmap, we alternately compute the expectation of sample $\mathbf{n}_j$ drawn from the i-th Gaussian component

$$Ez_{ij} = \alpha_i g(\mathbf{n}_j; \mu_i, \sigma_i) / \sum_{k=1}^{N} \alpha_k g(\mathbf{n}_j; \mu_k, \sigma_k),$$

and the maximization of the GMM parameters

$$\alpha_i = \frac{\sum_{j=1}^{M} Ez_{ij}}{M} \qquad \sigma_i^2 = \sum_{j=1}^{M} Ez_{ij} \frac{||\mathbf{n}_j - \mu_i||^2}{M \cdot d}$$

$$\mu_i \approx \frac{\sum_{j=1}^{M} Ez_{ij}\mathbf{n}_j + c\sum_{k=1}^{K}\alpha_{ki}\mu_{ki}}{\sum_{j=1}^{M} Ez_{ij} + c\sum_{k=1}^{K}\alpha_{ki}}$$

until the GMM parameters converge. Here, α_{ki}, μ_{ki} are the GMM parameters of neighboring texel k, $d = 2$ is the dimension of the unit normal vector, and c is a constant that modulates smoothness among corresponding Gaussians, which we set to 1 for all experiments.

To accelerate convergence, normal samples are clustered into N groups and each group is used to compute the initial values of $g_i = (\alpha_i, \mu_i, \sigma_i); 1 \le i \le N$. In a mipmap structure, each texel has 13 neighbors: eight at the same level, one from the immediate coarser level, and four at the finer level. Because of this network relationship, all the GMM parameters should ideally be estimated simultaneously, but this would result in a complex and expensive optimization. Instead, we compute the GMM of each texel one by one in scanline order, moving from coarser to finer levels, during which only those texels whose GMM parameters have been computed are counted as valid neighbors. Generally, this mipmap precomputation step converges quickly, in about 5 minutes for a 512×512 height field with a 2.8 GHz Pentium IV CPU.

4.2. Interpolation of Aligned GMM Components

With GMMs aligned by the described EM algorithm and stored in the reflectance mipmap, filtering is performed among corresponding GMM components. For the i-th component of a GMM, an interpolated Gaussian $g(\mathbf{n}, \hat{\mu}_i, \hat{\sigma}_i)$ is computed according to samples drawn from the K corresponding Gaussian distributions $g(\mathbf{n}; \mu_{ki}, \sigma_{ki}), k = 1, 2, \cdots, K$. An optimal interpolation can be expressed in closed-form as $\hat{\mu}_i = \frac{1}{K}\sum_{k=1}^{K}\mu_{ki}$ and $\hat{\sigma}_i{}^2 = \frac{1}{K}\sum_{k=1}^{K}\left[\sigma_{ki}{}^2 + ||\mu_{ki}||^2\right] - ||\hat{\mu_i}||^2$. Although the Gaussian mean $\hat{\mu}_i$ can be directly computed by trilinear interpolation, the variance $\hat{\sigma}^2$ cannot be interpolated linearly. However, the above closed-form solutions can be expressed using the the hardware trilinear interpolation function $T(p)$ for parameter p, such that optimal non-linear filtering of $\hat{\sigma}$ can nevertheless be efficiently computed in hardware as

$$\hat{\mu}_i = T(\mu_i) \qquad \hat{\sigma}_i{}^2 = T(\sigma_i^2 + ||\mu_i||^2) - ||\hat{\mu}_i||^2.$$

Taking the GMM mixture coefficients α_i into account, the above interpolation formulas can be extended to

$$\hat{\alpha}_i = T(\alpha_i) \qquad \hat{\mu}_i = T(\alpha_i\mu_i)/T(\alpha_i)$$

$$\hat{\sigma}_i{}^2 = T(\alpha_i(\sigma_i^2 + ||\mu_i||^2))/T(\alpha_i) - ||\hat{\mu}_i||^2$$

as derived in the Appendix. To facilitate this computation, the i-th Gaussian is stored in a floating-point texture map as $(\alpha_i, \alpha_i\mu_i(x), \alpha_i\mu_i(y), \alpha_i(\sigma_i^2 + ||\mu_i||^2))$, and hardware trilinear interpolation can directly be used to filter it. Since graphics hardware performs trilinear interpolations in real time, the distribution-based reflectance mipmaps are interpolated rapidly enough for real-time rendering. Our current implementation utilizes mixtures of four Gaussians, where each Gaussian component is stored as a texture.

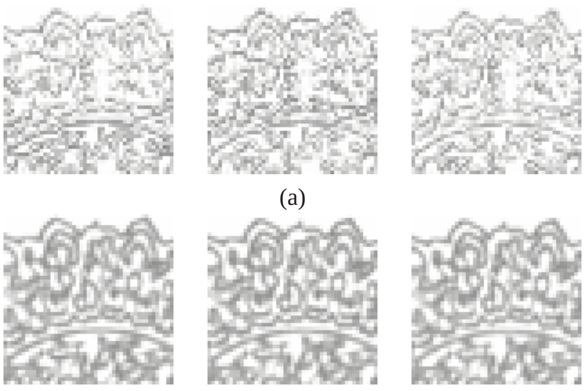

Figure 4: *A closeup section of the carved silver disk in Fig. 6 for three consecutive frames. (a) Rendered using the Toksvig method with a 16-bit floating-point normal mipmap; (b) Rendered with our method. Our results are more consistent between frames, while the Toksvig method has significant animation aliasing. In comparison to a ground truth image that traces 16 rays/pixel with our detailed reflectance model, the normalized Euclidean color error per pixel is 0.060 for our method and 0.165 for the Toksvig method.*

5. Results

We implemented our algorithm on a PC with a 2.8 GHz Pentium IV CPU and an nVIDIA 6800 GT graphics card. The

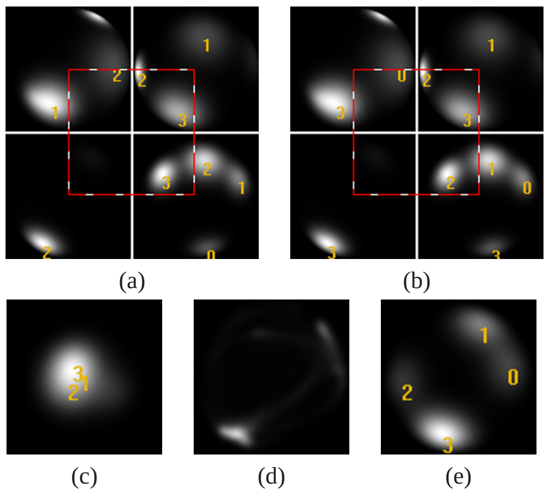

Figure 5: *Interpolation of independently computed GMMs in (a), and aligned GMMs from our method in (b). In (a) and (b), each black square represents a texel, and the dash square denotes a pixel area that overlaps parts of four texels. Gaussian components are numbered to show the correspondence, and interpolation takes place between components with the same index. Interpolation of (a) generates (c), which is far from the ground truth normal distribution (d). Interpolation of (b) generates a distribution (e) closer to (d).*

hardware-accelerated rendering is implemented in a single pass using Pixel Shader 3.0 and Vertex Shader 3.0 with DirectX. In total, 241 instructions are used in the Pixel Shader among which 4 are texture instructions and 45 instructions are used for the Vertex Shader.

Toksvig [Tok04] proposed a simple method that utilizes a single Gaussian with variant deviation to model microfacet normal distributions at different resolutions. However, subsampling a complex normal distribution by a single Gaussian can lead to significant aliasing. In Fig. 4, we compare aliasing of the method of Toksvig [Tok04] with that of our method for three consecutive frames from a camera zooming out. For a clearer contrast between the two methods, we refer the reader to the accompanying video, in which our method renders this example at 120 fps while our implementation of the Toksvig method renders at 125 fps.

In Fig. 5, the importance of GMM alignment is illustrated for the metal disk of Fig. 6. GMM interpolation results for aligned GMMs and unaligned, independently computed GMMs are displayed as microfacet normal histograms in x-y space. If GMM parameters are computed independently at each texel, the clarity of surface details is decreased and aliasing is introduced, as displayed in Fig. 6.

The need for multiresolution reflectance is exemplified by both the desert in Fig. 1 and the armor in Fig. 8, which consists of many small hemispherical metal scales. At coarser resolutions, the normal distribution within each pixel broadens, which effectively increases the surface roughness. In the

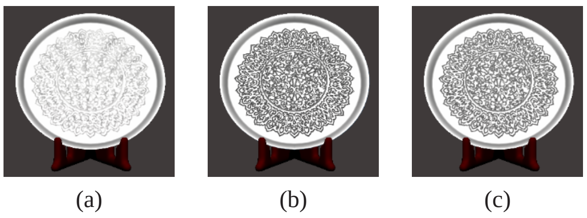

Figure 6: *GMM alignment for filtering. (a) With unaligned GMM mipmap; (b) Ground truth by multiple ray tracing; (c) With aligned GMM mipmap, formed by our method for normal distribution prefiltering. Interpolation of unaligned GMMs results in significant aliasing and loss of detail.*

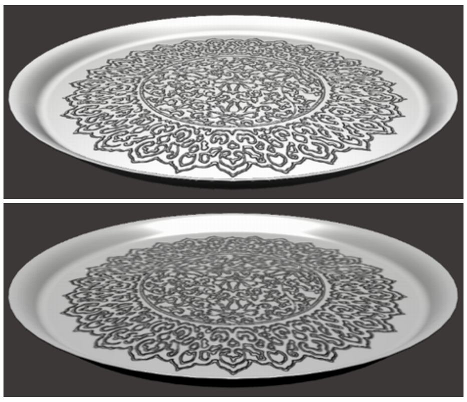

Figure 7: *Comparison of our result (top) and a result using Fournier's method (bottom). In comparison to multiple ray tracing, the color error in terms of normalized Euclidean color distance is 0.041 for our method and 0.123 for Fournier's method.*

figure, we compare our rendering results, results with a fixed BRDF represented by our reflectance model at highest resolution, and a ground truth image computed by multiple ray tracing (16/pixel). Since a fixed BRDF does not account for changes in pixel geometry at different scales, an incorrect amount of shine appears at coarse resolutions. Our frame rates for the desert and armor examples are 45 fps and 107 fps respectively at a 500×500 screen size, while multiple ray tracing renders at about 0.01 fps for the armor.

A comparison to Fournier's method with four Phong peaks is exhibited in Fig. 7, which displays a carved disk viewed from an oblique angle. A technical difficulty in using a combination of Phong peaks is its sensitivity in data fitting, which leads to some rendering artifacts. This example highlights the ability of our method to utilize additional hardware mipmap features, such as anisotropic filtering in this case, which is made possible by its use of hardware trilinear interpolation functions. Additionally, our method renders an order of magnitude faster than a hardware implementation of Fournier's method, at 120 fps vs. 13 fps for this disk, even though our method utilizes a more sophisticated reflectance model. For further comparisons, we refer the reader to the accompanying video.

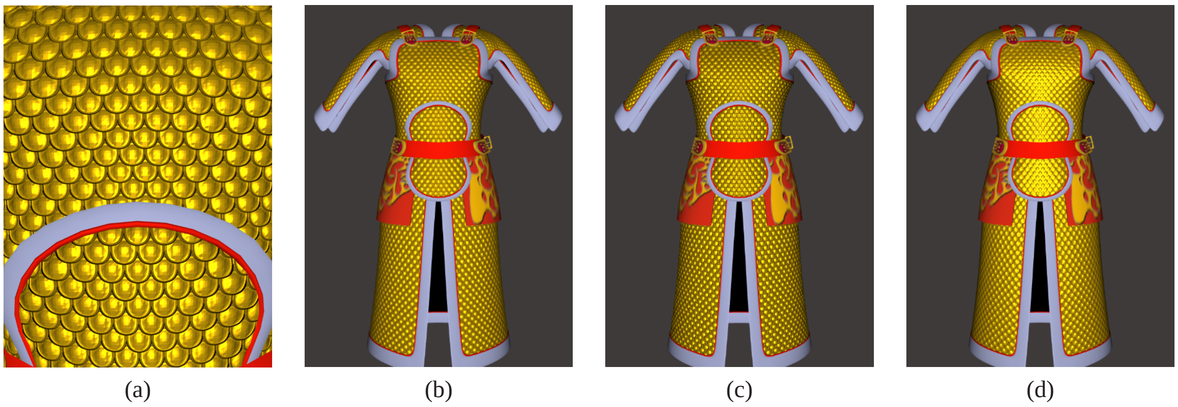

Figure 8: *Golden armor rendered at different resolutions. From a higher resolution (a) to a lower resolution (b), our method captures changes in reflectance properties, as seen in comparison to a ground truth image (c) rendered by multiple ray tracing. If reflectance variation is ignored and the armor is rendered with a fixed reflectance, overly shiny armor results at a coarse resolution (d). At the coarse resolution, the normalized Euclidean color error per pixel in comparison to multiple ray tracing is 0.058 for our method and 0.175 for the fixed BRDF.*

6. Conclusion

We have presented a real-time technique for multiresolution rendering of geometry-based reflectance. To handle distribution-based reflectance parameters needed for modeling multiresolution pixel geometry, we presented a method for mipmap precomputation that computes GMM parameters in a manner that yields accurate filtering and minimal aliasing. Although reflectance is represented by distribution-based quantities, we show how linear hardware filters can be exploited to attain real-time rendering.

Acknowledgement

This work is supported by the Hong Kong RGC grant HKUST 6182/04E and 6188/02E.

References

[APS00] Ashikhmin M., Premonze S., Shirley P.: A microfacet-based brdf generator. In *Proc. SIGGRAPH '00* (2000), pp. 65–74.

[Bil97] Bilmes J.: *A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*. Tech. Rep. ICSI-TR-97-021, Univ. of California-Berkeley, 1997.

[BM93] Becker B. G., Max N. L.: Smooth transitions between bump rendering algorithms. In *Proc. SIGGRAPH '93* (1993), pp. 183–190.

[CT81] Cook R. L., Torrance K. E.: A reflectance model for computer graphics. In *Proc. SIGGRAPH '81* (1981), pp. 307–316.

[DLR77] Dempster A. P., Laird N. M., Rubin D. B.: Maximum-likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society 39*, 1 (1977), 1–38.

[Fou92] Fournier A.: Normal distribution functions and multiple surfaces. In *Graphics Interface Workshop on Local Illumination* (1992), pp. 45–52.

[Hop96] Hoppe H.: Progressive meshes. In *Proc. SIGGRAPH '96* (1996), pp. 99–108.

[Ney98] Neyret F.: Modeling, animating, and rendering complex scenes using volumetric textures. *IEEE Trans. on Visualization and Computer Graphics 4*, 1 (1998), 55–70.

[Sch94] Schlick C.: A survey of shading and reflectance models. *Computer Graphics Forum 13*, 2 (1994), 121–132.

[Tok04] Toksvig M.: *Mipmapping Normal Maps*. Tech. Rep. TB-01256-0001, nVIDIA Corporation, 2004.

[TS67] Torrance K. E., Sparrow E. M.: Theory for off-specular reflection from roughened surfaces. *J. Optical Society of America 57* (1967), 1105–1114.

[WAT92] Westin S. H., Arvo J. R., Torrance K. E.: Predicting reflectance functions from complex surfaces. In *Proc. SIGGRAPH '92* (1992), pp. 255–264.

[Wil83] Williams L.: Pyramidal parametrics. In *Proc. SIGGRAPH '83* (1983), pp. 1–11.

Appendix

To linearly interpolate GMMs $\{\Theta_k; 1 \le k \le K\}$ according to texel weights $\{w_k; 1 \le k \le K\}$, we totally draw N samples from these GMMs, among which N_k samples come from GMM Θ_k. Here, $N = \sum_{k=1}^{K} N_k$ and $N_k/N = w_k$. The optimal interpolation $\hat{g}_i$ is fit to samples drawn from corresponding Gaussians $\{g_{ki}; 1 \le k \le K\}$. The number of samples drawn from g_{ki} is $N_k\alpha_{ki}$. The mean and variance of these samples are μ_{ki} and σ_{ki}^2 respectively. Therefore,

$$\begin{aligned}
\hat{\alpha}_i &= \frac{1}{N}\sum_{k=1}^{K} N_k\alpha_{ki} = \sum_{k=1}^{K}\frac{N_k}{N}\alpha_{ki} = \sum_{k=1}^{K} w_k\alpha_{ki} = T(\alpha_i) \\
\hat{\mu}_i &= \frac{\sum_{k=1}^{K}(N_k\alpha_{ki})\mu_{ki}}{\sum_{k=1}^{K} N_k\alpha_{ki}} = \frac{\sum_{k=1}^{K}\frac{N_k}{N}\alpha_{ki}\mu_{ki}}{\sum_{k=1}^{K}\frac{N_k}{N}\alpha_{ki}} = \frac{\sum_{k=1}^{K} w_k(\alpha_{ki}\mu_{ki})}{\sum_{k=1}^{K} w_k\alpha_{ki}} = \frac{T(\alpha_i\mu_i)}{T(\alpha_i)} \\
\|\hat{\mu}_i\|^2 + \hat{\sigma}_i{}^2 &= \frac{\sum_{k=1}^{K}(N_k\alpha_{ki})(\|\mu_{ki}\|^2+\sigma_{ki}^2)}{\sum_{k=1}^{K} N_k\alpha_{ki}} = \frac{\sum_{k=1}^{K}\frac{N_k}{N}\alpha_{ki}(\|\mu_{ki}\|^2+\sigma_{ki}^2)}{\sum_{k=1}^{K}\frac{N_k}{N}\alpha_{ki}} \\
&= \frac{\sum_{k=1}^{K} w_k\left(\alpha_{ki}(\|\mu_{ki}\|^2+\sigma_{ki}^2)\right)}{\sum_{k=1}^{K} w_k\alpha_{ki}} = \frac{T\left(\alpha_i(\|\mu_i\|^2+\sigma_i^2)\right)}{T(\alpha_i)}
\end{aligned}$$