Wavelet encoding of BRDFs for real-time rendering

Luc Claustres, Loïc Barthe, Mathias Paulin

To cite this version:

Luc Claustres, Loïc Barthe, Mathias Paulin. Wavelet encoding of BRDFs for real-time rendering, Graphics Interface, May 2007, Montréal, Canada. pp.169-176, 10.1145/1268517.1268546. hal-01518547

HAL Id: hal-01518547
https://hal.archives-ouvertes.fr/hal-01518547
Submitted on 4 May 2017

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

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Wavelet Encoding of BRDFs for Real-Time Rendering

Luc Claustres∗ Loic Barthe† Mathias Paulin‡

IRIT - University of Toulouse, France

ABSTRACT

Acquired data often provides the best knowledge of a material’s bidirectional reflectance distribution function (BRDF). Its integration into most real-time rendering systems requires both data compression and the implementation of the decompression and filtering stages on contemporary graphics processing units (GPUs). This paper improves the quality of real-time per-pixel lighting on GPUs using a wavelet decomposition of acquired BRDFs. Three-dimensional texture mapping with indexing allows us to efficiently compress the BRDF data by exploiting much of the coherency between hemispherical data. We apply built-in hardware filtering and pixel shader flexibility to perform filtering in the full 4D BRDF domain. Anti-aliasing of specular highlights is performed via a progressive level-of-detail technique built upon the multiresolution of the wavelet encoding. This technique increases rendering performance on distant surfaces while maintaining accurate appearance of close ones.


Keywords: brdf, wavelet, real time rendering

1 INTRODUCTION

The past few years have seen impressive improvements in the capabilities of graphics processing units (GPUs). Among many features, the most fascinating achievement is the realisation of GPU programmability for real-time high quality local illumination by directly evaluating physically-based reflection models as each fragment is shaded [25]. These models describe a bidirectional reflectance distribution function (BRDF) defined as the ratio of outgoing radiance to incoming irradiance at a surface point x [32]. The lighting configuration, i.e. the lighting or incoming direction \( \omega_i = (\theta_i, \phi_i) \) and the viewing or outgoing direction \( \omega_o = (\theta_o, \phi_o) \), is expressed with spherical coordinates relative to the local coordinate system of the surface (Figure 2). In real-time applications, the direct illumination usually comes from a finite set of \( N \) point sources and therefore the general rendering equation [13] reduces to:

\[
L_o(x, \omega_o) = \sum_{n=1}^{N} f_r(x, \omega_i^n, \omega_o) \frac{I_n \cos \theta_o}{||x - x_n||^2},
\]

where \( L_o \) is the reflected radiance from a surface point \( x \) in the viewing direction, \( I_n \) is the position (respectively the intensity) of the light source number \( n \), and \( f_r \) is the BRDF.

Although numerous reflection models have been created [2, 17,
41, 33], a recent experimental analysis [31] has shown that real materials often exhibit a complexity that exceeds the expressive power of current BRDF models. Moreover, the common method of fitting a measured BRDF dataset to analytical models generally requires robust nonlinear optimisation techniques (such as SQP or Levenberg-Marquardt) that are hard to implement and converge slowly. Indeed, low sampling rate and/or noise level make fitting very underconstrained and heavily dependent on the initial guesses. On the other hand, complex models are too computationally expensive for current GPU capabilities. Practical implementation requires to break up the model into separate functions that are sampled and stored in texture maps to avoid direct evaluation [11].

As a consequence, acquired data often provides the best knowledge of the reflectance of a real material. Ideally, on account of the available texture-mapping and computational capabilities of GPUs, we would like a numerical BRDF model that exhibits the following useful features:

- **generality** to handle all-frequency materials (from diffuse to specular),
- **compression** to manage potentially large datasets with controllable error,
- **efficiency** to be evaluated at per-pixel level,
- **filtering** to smoothly reconstruct the BRDF between acquired samples.

2 Related works

Recently, bidirectional texture functions (BTFs) [7], modelling realistic acquired materials at pixel scale, have been well studied and produce convincing results [27]. However, the principal component analysis (PCA) [29] used suffers from memory problems during computation and the reconstruction is fast and correct only for relatively simple materials.

*Factorisation* techniques represent 4D BRDFs using low-dimensional functions (factors) that are multiplied together [9]. Kautz [15] uses a numerical approach based on a singular-value decomposition (SVD) and builds a factorisation of 2D functions. These functions can be stored in texture maps and combined using graphics hardware to perform real-time per-pixel reconstruction. Several improvements have been proposed to optimise the factorisation for GPUs by suppressing negative terms [26], limiting dynamic range, or improving parameterisation [37]. Large data compression rate can be achieved with factorisation but without flexibility in weighting quality against space. More, these decompositions fail to reproduce fine details (high-frequencies) of a complex material and reconstruction of arbitrary fidelity can require a large number of factors, reducing performances.

*Spherical harmonics* (SH), which are the Fourier basis functions on the sphere, are often used to represent BRDFs [42]. Indeed, SH are very interesting for shading because their use reduces the lighting integral to a dot product. Nevertheless, SH have global support on the sphere and thus need numerous coefficients to encode general BRDFs. As a consequence, they are restricted to off-line rendering or low-order reconstruction, which can only approximate low-frequency lighting and shadowing effects [30]. On the contrary, Lalonde et al. [18] propose a BRDF representation based on the projection of the data onto a base of 4D wavelet functions. Then, a zerotree encoding is used to efficiently store and evaluate the BRDF. More recently, Claustres et al. introduce the generic wavelet transform [5]. Spectral BRDF datasets are independently projected onto each directional and wavelength dependence using dedicated wavelet transforms. This leads to a better compression ratio according to the reconstruction error.

*Precomputed radiance transfer* (PRT) [36] aims at enabling real-time illumination with arbitrary BRDFs represented by SH [16], wavelets [22] or factorisation [19]. Interactive rendering including environment lighting, shadowing and interreflections is achieved by pre-computing a sparse light transport matrix per vertex that reduces lighting computations at rendering time. However, because lighting is evaluated per vertex only and interpolated across triangles by graphics hardware, PRT is not well-suited to deal with the high-frequency local illumination induced by realistic materials. It also requires highly tessellated 3D models, prohibitive memory and costly pre-computations.

**Discussion:** Among these techniques, wavelets exhibit most of the required features. Indeed, an interesting property is the compact support of most wavelets, leading to a fast local reconstruction (logarithmic time according to the number of samples). The discrete wavelet transform produces the same number of coefficients as samples in the original dataset, but many of them are close to zero. Flexible lossy compression is obtained by zeroing those that are below a certain threshold. At last, wavelets can handle all-frequency lighting and shadowing effects [30]. For these reasons we propose a wavelet decomposition of the BRDF dataset and implement the reconstruction and filtering stages on the fragment processor, hence providing BRDF-based local illumination with both high-quality and real-time rendering (Figure 1). We also profit from the multiresolution (reconstruction at different levels of accuracy).
of the wavelet decomposition to improve filtering by performing anti-aliasing of specular highlights (Figure 3).

3 Overview of Our Contribution

Our main contribution is a complete numerical BRDF model designed for GPUs, i.e. a compressed representation providing reconstruction with magnification as well as minification filtering in the full 4D BRDF domain in real-time. At the rendering time, the number of texture accesses for each fragment depends on the resolution chosen on-the-fly for the BRDF reconstruction. If the surface is far away from the viewpoint, fewer levels are required to estimate the BRDF and the performance is enhanced. These results are achieved with built-in hardware filtering and by using the linearity and the multiresolution of the wavelet encoding. Moreover, a simple indirect map efficiently compresses BRDF data by removing much of the BRDF correlation between hemispheres corresponding to a set of close directions.

Figure 4 details our BRDF acquisition, encoding and rendering pipeline. First, the BRDF is measured using a gonioreflectometer. Before the wavelet encoding, a pre-process resamples the acquired data to match a regular sampling grid of the BRDF’s domain. We complete the sparse dataset using 4D nearest-neighbour queries on directions as done by Schregle [34]. Then, a moving least square (MLS) approximation is applied to smooth the resulting data, reducing measurement noise and discontinuities possibly introduced in the previous pre-processing pass. Next, the wavelet transform is applied in place on the BRDF data, then compression is performed (4). The remaining wavelet coefficients are quantised (5.1) and packed (5.2) into a 3D texture uploaded to the GPU before rendering time. For each pixel in the rendered image, the original BRDF is reconstructed (5.3) from the compressed data with the proper accuracy depending on the distance to the viewpoint (5.4.1). At last, the BRDF samples are filtered in order to produce a smooth lighting solution (5.4.2). This complete pipeline allows modelling and real-time rendering of acquired materials in common graphics solutions (5.4.2).

4 Wavelet Encoding

Wavelets have long been used for data compression, in particular of 2D images, and are the core of the recent JPEG2000 standard [39]. GPU implementations to speed up the transform, e.g. the JasPer codec [40], do exist but decompression occurs on the whole image and not on a per-pixel basis [12]. Although Candussi presents a novel representation of the wavelet coefficient tree amenable to current graphics hardware for 2D data sets [3], compression is limited because the sparse tree is represented as a costly index texture referencing a wavelet coefficient texture. This technique only offers point sampling, which results in color banding and is not visually acceptable for high-fidelity rendering.

4.1 Parameterisation

We use an implicit approach, which proposes a parameterisation of the sphere, i.e. a mapping between \( S^2 \) and \( \mathbb{R}^2 \), where the transform is more easily expressed. An explicit approach, which defines a wavelet transform over the spherical space described as a geometrical mesh [35], has also been explored. An additional indexing map must be used at run-time to make the correspondence between directions and triangles by encoding the mesh into textures. This technique suffers from aliasing, requires high-resolution textures to ensure high-quality reconstruction, and makes filtering schemes much more complex.

Lewis [21] recommends the use of Nusselt embedding to compromise on the matter of redundancy at the poles when parameterising the directional component of a BRDF. Christensen [4] prefers to use a combination of a gnomonic projection and stretch to map directions to the unit square. In a real-time context, the cartesian to spherical coordinates transform remains the most obvious and cheap mapping to use though. To limit visual artifacts in our implementation, the poles are aligned with the tangent instead of the normal direction in the local frame of the surface. Indeed, artifacts are less obvious at grazing angles than at front views. This leads to the following mapping: 

\[
(x = \sin(\theta) \cos(\phi), y = \cos(\theta), z = \sin(\theta) \sin(\phi)) \quad \text{with} \quad \theta \in [0, \pi] \quad \text{and} \quad \phi \in [0, \pi].
\]

Another advantage is that both angles are valued in a similar range. Thus, uniform sampling of \( S^2 \) results in a squared grid pattern that simplifies algorithms such as the wavelet transform or data filtering.

4.2 Transform

The transform encodes a set of BRDF samples (hereafter referred to as \( f_r \) values) into two parts: the scaling coefficients encoding the smooth approximation (low-frequency \( l \) values) and the wavelet coefficients encoding the details (high-frequency \( h \) values), i.e. the missing information to retrieve the original samples from the approximation. The process is recursively repeated on the \( l \) values providing a hierarchical dyadic decomposition of the samples, i.e. \( 2^N \) values lead to \( N \) levels or resolutions. We selected the Haar wavelet basis because of its narrow support, which ensures less computational requirements and matches our real-time constraint. Haar’s analysis and synthesis formula for a level number \( n \) are given in the following equations:

\[
\begin{align*}
\psi_{l+1} &= \frac{1}{\sqrt{2}} (f_{2l} + f_{2l+1}) \\
\psi_{h+1} &= \frac{1}{\sqrt{2}} (f_{2l} - f_{2l+1}) \\
\end{align*}
\]

When the data have multiple dimensions, the most efficient decomposition consists in applying the transform successively on each dimension at each level (non-standard approach). Hence local analy-
s and synthesis are performed in the longitude, then latitude angle, on the nested hemispherical grids (Figure 5).

Figure 5: 2D Wavelet transform applied on each hemisphere of the original BRDF data.

4.3 Compression
Claustres et al. have presented in [6] a wavelet compression scheme that generates high compression rates thanks to the computation of an adaptive (local) threshold for each dependence of the BRDF, which is reused in this work. The main idea consists in removing the hemispherical-to-hemispherical BRDF correlation between incoming directions in addition to standard wavelet compression. A set of hemispherical wavelet coefficients is viewed as a vector, which magnitude indicates the relative weight of corresponding hemispherical data in the acquired BRDF. If the vector magnitude lower than a given threshold, all hemispherical coefficients are removed from the wavelet encoding.

Efficient sparse representations of the coefficients usually requires low-level memory access, such as bit masks, and make intensive use of pointers, both techniques are not still available on modern GPUs. More simple indexing techniques, which are easily amenable to graphics processors, are usually not efficient because the memory size of an index (int) or a single coefficient (float) is quite similar. However, using this high-level compression scheme, an index is cheap with regard to a set of hemispherical wavelet coefficients. Thus, indexing becomes efficient as detailed in the next section.

5 WAVELET-BASED BRDF ON GRAPHICS HARDWARE

5.1 Data Quantisation
Precision and dynamic range are potential problems when using a limited precision per color component. Indeed, BRDFs can present arbitrary large dynamic ranges that result in contouring artifacts. To limit this problem, we replace \( f_r \) by \( f_r = \log(\frac{f_r}{\mu}) \) as suggested by McCool [26], where \( \mu \) is the average of all BRDF samples and \( \varepsilon \) a bias factor ensuring strictly positive values. Another advantage is that minimizing RMS error on \( f_r \) results in minimizing relative RMS error on \( f_r \). This is perceptually desirable since the eye is sensitive to ratios of intensity and not absolute intensity. Using this logarithmic encoding, 24-bits per-pixel textures are sufficient as modern GPUs support floating-point precision from end to end of the graphics pipeline. Indeed, it has been acknowledged that wavelet transforms can accommodate low-precision encoding of the coefficients if computations can be done with higher precision [23].

5.2 Data Storage
Since a BRDF is a 4D function, suitable storage would require 4D texture maps. Unfortunately, no commodity hardware currently supports textures with dimensions greater than three. To overcome this limit we organize the 4D BRDF data as a 3D texture where each 2D slice corresponds to a fixed outgoing direction. The normalized incoming latitude and longitude angles computed from the spherical coordinates of the lighting vector are related to the texture coordinates \((s, t)\) within each slice. The 2D non-standard wavelet transform is applied independently on each slice resulting in a multiresolution texture of wavelet coefficients.

Performing compression then results in suppressing a set of outgoing hemispheres, i.e. vectors of wavelet coefficients, corresponding to a given set of incoming directions. Actually, owing to BRDF reciprocity, this is equivalent to suppress the reciprocal incoming hemispheres, i.e. 2D slices in the 3D texture. The remaining slices are contiguously packed into a new 3D texture with lower depth. An additional compression map stores for each slice of the uncompressed texture its corresponding index in the compressed texture. We save this map as a 1D RGB texture encoding 16-bit indices, which is sufficient to handle hemispheres sampled up to \(2^8 \times 2^8\) in latitude and longitude angles.

In order to implement zeroing of all wavelet coefficients in the hemisphere, we should effectively remove the corresponding empty slice from the 3D texture. Consequently, texture look-up would require branching to test whether the data exists or not. However, branching is only supported on the most recent GPUs and it is only efficient with spatial coherence. Our experiments (restricted to NVIDIA 7x00 GPUs) show that it remains more efficient to avoid branching by referencing a zeroed slice (the same for all the empty hemispheres). This slice is stored as the first 2D slice of the 3D texture and it is referenced by the reserved index 0. The full data organisation is summarized in Figure 6.

Usually, the maximum size of each dimension for 3D textures cannot exceed \(2^3(512)\) pixels while 2D textures can often reach \(2^{12}(4096)\) pixels. This is problematic for outgoing directions that are stored sequentially in the 3D texture’s depth. Indeed, the best sampling allowed is then \(2^4 \times 2^4\) in latitude and longitude angles, which is too sparse for most datasets. Fortunately, to reach a better sampling rate, compression by limiting texture depth is worth having.

5.3 Data Reconstruction
For the desired lighting configuration, the synthesis of the BRDF value is done from the lowest resolution, adding more and more details until the highest resolution is reconstructed. At each level, the wavelet coefficients added to refine the current approximation are retrieved from the 3D BRDF texture. Once specified in the uncompressed texture, the texture coordinates are mapped to the compressed version using the compression map (Figure 6). While the \((s, t)\) coordinates of the BRDF texture are directly related to the spherical coordinates of the incoming direction, the third coordinate corresponding to the outgoing direction is computed using a 2D to 1D mapping from the normalised outgoing latitude and longitude angles.
Local BRDF reconstruction requires 2 texture look-ups (longitude then latitude coefficient read) at each level of the decomposition. This number must be doubled as the compressed BRDF texture is indirectly accessed through the compression map. Taking into account the final scaling coefficient read, the total number of look-ups is $4N + 1$ for a $N$ level decomposition. For instance, the local reconstruction of a typical 5-level BRDF requires 21 texture look-ups. Depending on the filtering, several samples have to be reconstructed per-pixel, hence increasing the total number of texture look-ups.

### 5.4 Data Filtering

#### 5.4.1 Data Minification Filtering

Although the BRDF is invariant with respect to the distance to the camera, sparkling noise may appear due to aliasing of localised high frequencies, i.e. specular highlights (Figure 10). Given the characteristics of an object, Amanatides [1] clamped the shininess of the Phong’s BRDF model to values that will not introduce aliasing. Tan [38] uses a Gaussian mixture model, which parameters are pre-computed at different scales and stored into mipmaps to be efficiently evaluated on the GPU at rendering time. Solutions were also proposed for normal maps [14, 8].

BRDF minification filtering theoretically depends on the derivative of the full lighting configuration, where the light/viewer distance and orientation play a role. However, due to singularities in the spherical coordinates system, this is practically difficult to compute. We rather propose to use a simple range-based LOD selection method [28] to obtain a continuous reconstruction level $l \in [0, N-1]$ on a per-pixel basis when performing rendering. More specifically: $l = \min (\max (\log_2 (ad), 0), N-1)$, where $d$ is the distance from the viewpoint to the rendered fragment and $\alpha$ is a scaling factor. The final BRDF value is linearly interpolated between the value reconstructed at levels $\lceil l \rceil$ and $\lceil l \rceil + 1$, according to the fractional part of $l$. This is attractively integrated into our multisolution encoding, without additional memory requirements. From the number of texture look-ups required for a complete BRDF synthesis, we find that performances are enhanced using the LOD reconstruction when $l < \frac{N-1}{2}$, otherwise it is more expensive. For instance, rendering becomes faster when $l < 2.375$ for a typical 5-level decomposed BRDF. As a consequence, LOD is interesting in term of performance when most of the pixels are covered by surfaces far away from the viewpoint.

The value of the scaling factor $\alpha$ is typically derived from the bounding volume of the scene and/or a perception-based criterion in order to make the filtering efficient in practical cases. In our experiment, a value of $\alpha = 0.15$ gives good results. Energy conservation, which is a fundamental issue when dealing with realistic reflectance and physically-based illumination, is ensured by the Haar transform. Indeed, for orthonormal wavelet families, e.g. Haar, the total energy in the coefficients is equal to the total energy in the original samples at each scale of the decomposition. For example, the median and mean luminance values computed for both images in Figure 3 are equal. However, as the energy of specular peaks is redistributed on the diffuse component of the BRDF, the scene seems to darken with the LOD used for rendering.

#### 5.4.2 Data Magnification Filtering

A crucial point for the visual realism is the quality of the smooth reconstruction of the BRDF for lighting configurations between acquired samples. Due to the computational complexity of 4D schemes, we share this magnification process between built-in hardware bilinear filtering (incoming directions) and software bilinear filtering in the pixel shader (outgoing directions). Indeed, the linearity of the wavelet transform allows us to directly interpolate the coefficients of the encoded function, without the need of reconstruction. Thus, the final step consists in blending the resulting hardware filtered values in the pixel shader.

### 6 Results

#### 6.1 BRDF Data

BRDFs are still difficult to acquire, however extensive research is being carried out to provide a better data availability in the future [31, 24, 41]. Up to now, we only have measured the RGB BRDF of different surfaces such as wood, cloth and velvet. We have also used synthetic datasets generated through analytical BRDF models in order to validate our approach. We selected the isotropic model of Lewis [20] and the anisotropic models of Ward [41] and Poulin-Fournier [33].

#### 6.2 Reconstruction Error

Table 1 presents the averaged RMS ($\varepsilon$) and relative ($\varepsilon_r$) errors for our set of acquired and synthetic BRDFs, with respect to the compression ratio $r_c$. Initial BRDF resolution of input data is $(32 \times 32)^2$, leading to a BRDF texture of 12MB stored in 32-bits floating point numbers. Evaluation error is always satisfying on the compressed dataset, even for radical thresholding. Results of Table 1 were achieved by measuring CPU reconstruction error. However, GPU-based BRDF matrices have an additional implicit compression ratio of 4:1 from the 32- to 8-bits data quantisation. The resulting additional error varies between 3% and 8% depending on the datasets. Usually, the final compressed texture used for our real-time applications has a size of less than 1MB, without noticeable visual artifacts.

<table>
<thead>
<tr>
<th>BRDF</th>
<th>Cloth</th>
<th>Wood</th>
<th>Velvet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean value</td>
<td>0.283</td>
<td>0.322</td>
<td>0.323</td>
</tr>
<tr>
<td>$r_c$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon_r$ (%)</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td>4:1</td>
<td>0.00537</td>
<td>1.64</td>
<td>0.00647</td>
</tr>
<tr>
<td>16:1</td>
<td>0.00930</td>
<td>2.707</td>
<td>0.0133</td>
</tr>
<tr>
<td>32:1</td>
<td>0.0125</td>
<td>3.23</td>
<td>0.0186</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BRDF</th>
<th>Lewis</th>
<th>Ward</th>
<th>Poulin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean value</td>
<td>0.282</td>
<td>0.278</td>
<td>0.394</td>
</tr>
<tr>
<td>$r_c$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon_r$ (%)</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td>4:1</td>
<td>0.0247</td>
<td>1.74</td>
<td>0.832</td>
</tr>
<tr>
<td>16:1</td>
<td>0.0417</td>
<td>2.90</td>
<td>0.840</td>
</tr>
<tr>
<td>32:1</td>
<td>0.0520</td>
<td>3.73</td>
<td>0.856</td>
</tr>
</tbody>
</table>

Table 1: Modelling errors for acquired (top) and synthetic (bottom) BRDFs.

#### 6.3 Examples

We have implemented the reconstruction algorithm on the NVIDIA GeForce 7800 GTX graphics processor. Vertex and fragment shaders are written in the NVIDIA Cg shading language managed through the OpenGL 2.0 API. The CPU programming has been implemented on an Athlon64 XP3500+ processor running Linux. For a 3-level decomposition, the GPU reconstruction is at least 152 times faster ($\sim 49M$ versus 321K BRDF evaluations per seconds). Even though the use of SIMD extensions of modern CPUs would show a substantial improvement in performance, our gain remains impressive.

Images are computed with a single (but not restricted to) point light source, and the rendering process is separated into two parts. First, a vertex shader program transforms the lighting and viewing
vectors into the local frame of the surface. Then, a fragment shader program computes the final image as follow:

1. determine the reconstruction level based on the distance to the viewpoint;
2. deduce the 3D texture coordinates to access the BRDF texture at required resolution;
3. synthesis of the BRDF value by appropriately weighting coefficients at the different scales;
4. evaluate the lighting equation 1.

Table 2: GPU performances (FPS) depending on the data resolution and filtering. Images are generated in a 512x512 floating point pixel buffer. The timing includes the first render pass that reconstructs per-pixel BRDF-based local illumination and the second pass that displays the pixel buffer.

<table>
<thead>
<tr>
<th>LOD level</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest</td>
<td>390</td>
<td>258</td>
<td>192</td>
<td>152</td>
<td>124</td>
</tr>
<tr>
<td>Bilinear</td>
<td>206</td>
<td>90</td>
<td>56</td>
<td>41</td>
<td>31</td>
</tr>
</tbody>
</table>

Figure 8: (a) Real-time per-pixel lighting using our wavelet encoding for different acquired BRDFs, from top to bottom: Cloth, Wood and Velvet. (b) Comparison between (top) our approach, (middle) per-pixel lighting using the Lewis analytical BRDF model, and (bottom) per-vertex lighting again using the Lewis model for the same plastic.

Images generated from the acquired BRDFs presented in Table 1 are shown in Figure 8a. Our approximation is also compared with the theoretical BRDF at per-pixel and per-vertex level in Figure 8b. On the fabric model (11K triangles) we have obtained the rates presented in Table 2. Our approach is mainly limited by the number of fragments generated and not by the geometry complexity of the scene. Thus, it can take special advantage of deferred shading to accommodate a large number of vertices or primitives with similar overall performances.

We also compare our BRDF reconstruction and per-pixel lighting on the GPU to a reference lighting solution computed by ray tracing on the CPU (Figure 9). The 3D model (16K triangles) is used with the acquired velvet BRDF, which is our most complex dataset. The mean per-pixel perceptual error (L*a*b* color space) on the resulting images is 1.3 and the corresponding standard deviation 2.29. This demonstrates the rendering accuracy in spite of the data quantisation since an error around 1 in the L*a*b* space is the threshold for non perceptible errors.

Figure 7 illustrates the high-quality local illumination provided by our BRDF representation, mixed with classical bump and environment mapping, for a scene composed of 11K triangles referencing four different BRDFs (velvet, cloth, wood and plastic).

In Figure 10b a terrain scene of 45K triangles is compared with the finest BRDF reconstruction and the LOD reconstruction. For a viewpoint far away from the surface, most specular highlight artifacts are removed by the LOD. The different reconstruction levels used for specular antialiasing are also shown in Figure 10a. The re-
ups when performing LOD. Also be exploited to avoid redundant computations and texture look-up. Coherence between reconstruction at different levels could highlights without additional memory cost. We provide a novel multiresolution representation that allows the real-time reconstruction using graphics hardware and per-pixel local wavelets for acquired data. This representation is suitable for real-time reconstruction using graphics hardware and per-pixel local illumination with general (anisotropic) BRDFs. BRDF data are stored into moderate-resolution 3D textures (typically $32 \times 32 \times 256$) and thus a scene may contain many different BRDFs. We provide a novel multiresolution representation that allows the real-time rendering of BRDF at different LODs and the filtering of specular highlights without additional memory cost.

We plan to improve the fine detail reconstruction by using other wavelets than Haars', certainly at the cost of slower reconstruction. Coherence between reconstruction at different levels could also be exploited to avoid redundant computations and texture look-ups when performing LOD.

7 Conclusion and Future Work

We have presented an efficient BRDF representation based on wavelets for acquired data. This representation is suitable for real-time reconstruction using graphics hardware and per-pixel local illumination with general (anisotropic) BRDFs. BRDF data are stored into moderate-resolution 3D textures (typically $32 \times 32 \times 256$) and thus a scene may contain many different BRDFs. We provide a novel multiresolution representation that allows the real-time rendering of BRDF at different LODs and the filtering of specular highlights without additional memory cost.

We plan to improve the fine detail reconstruction by using other wavelets than Haars’, certainly at the cost of slower reconstruction. Coherence between reconstruction at different levels could also be exploited to avoid redundant computations and texture look-ups when performing LOD.

REFERENCES


