Decoupling BRDFs from Surface Mesostructures

Jan Kautz[1] Mirko Sattler[2] Ralf Sarlette[2] Reinhard Klein[2] Hans-Peter Seidel[3]

MIT[1]
Cambridge, USA

Universität Bonn[2]
Bonn, Germany

MPI Informatik[3]
Saarbrücken, Germany

Abstract

We present a technique for the easy acquisition of realistic materials and mesostructures, without acquiring the actual BRDF. The method uses the observation that under certain circumstances the mesostructure of a surface can be acquired independently of the underlying BRDF.

The acquired data can be used directly for rendering with little preprocessing. Rendering is possible using an offline renderer but also using graphics hardware, where it achieves real-time frame rates. Compelling results are achieved for a wide variety of materials.

Key words: Reflectance, Texture Mapping, Graphics Hardware.

1 Introduction

One of the long sought goals in computer graphics is to create photorealistic images. To this end, not only physically correct global illumination algorithms are necessary but also realistic material properties. It is especially important to capture the fine spatial variation and mesostructure of materials since they convey a lot of information about the material.

Unfortunately, the acquisition of material properties is fairly tedious and time-consuming [3, 15, 19]. This is because material properties are specified by the four-dimensional *bidirectional reflectance distribution function* (BRDF) which depends on the local light and viewing directions. If spatial variation across a surface is to be captured, this function (usually called *bidirectional texture function* or *BTF* [3]) becomes six-dimensional. Obviously, the acquisition of such a high-dimensional function requires taking many samples, resulting in large acquisition times (and storage requirements), which is the reason why not many BTF measurements are easily available.

The benefit of capturing all this data is that BTFs generate very realistic renderings of materials, since they capture not only the local reflectance properties for each point on a surface, but also all non-local effects, such as self-shadowing, masking, and interreflections within a material.

In this paper, we present an empirical method that permits easy acquisition and realistic rendering of such spatially-varying realistic materials, e.g., see Figure 1. It is based on the following observation. Let us assume, we have captured a full BTF $b(\hat{\omega}_i, \hat{\omega}_o, \underline{x})$ for some discrete number of viewing and light directions $\hat{\omega}_o^m$ and $\hat{\omega}_i^n$. It can be converted into an average BRDF f_r (i.e. the BRDF for the material if viewed from a distance) by averaging the exit radiance over the captured area $\mathcal{S}$:

$$f_r(\hat{\omega}_i, \hat{\omega}_o) \cos\theta_i = \frac{1}{\mathcal{S}} \int_{\mathcal{S}} b(\hat{\omega}_i, \hat{\omega}_o, \underline{x})\, d\underline{x} =: \bar{b}(\hat{\omega}_i, \hat{\omega}_o)$$

We now assume that the captured material patch is isotropic and has only small-scale features (no large self-shadowing and masking effects). All the BTF slices with the same average exit radiance $\bar{b}$ will look roughly the same then. This means that we only need to store one representative for each average radiance and can use the average BRDF to lookup the appropriate BTF slice, since there is exactly one only.

Instead of acquiring a full BTF, we therefore propose to acquire only some slices of the BTF, so that we have

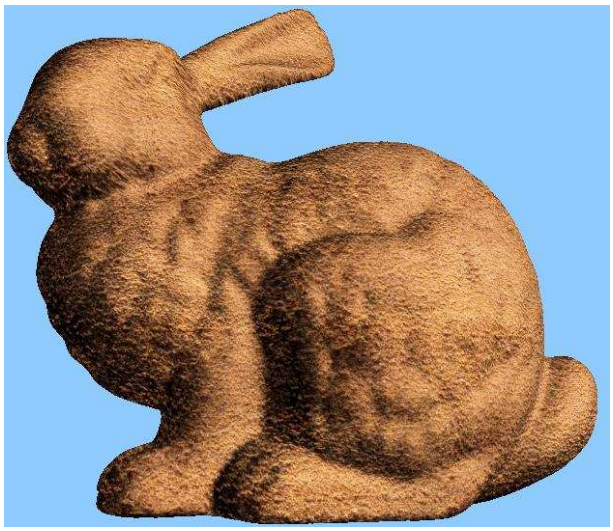

Figure 1: The Stanford bunny rendered with acquired suede. The acquisition time for the material was about 5 minutes.

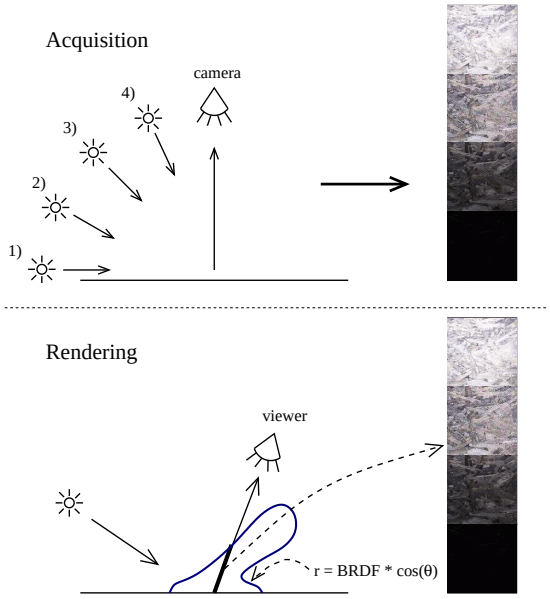

Figure 2: ***Overview****. We capture images for different light directions and for a fixed orthogonal view resulting in the shading map. For each image, we compute the average emitted radiance. During rendering, we compute the value r with some freely chosen BRDF and use this value to do a per-pixel lookup into our stack of images according to the average radiance. Finally, we scale the value with the intensity of the light source.*

representative samples for different average radiances; see Figure 2 (in a sense this corresponds to the tonal art maps introduced by Praun et al. [27]). We do *not* acquire the average BRDF for the lookup. Instead, we allow the user to use some arbitrary BRDF; see Figure 2. This decouples the BRDF from the mesostructure of the surface. If the user chooses a BRDF equal to the original average BRDF, the resulting material looks the same as the original material (given that above assumptions are met), but e.g. the user can also choose to make the material more or less specular.

For real-time rendering the necessary lookup can be implemented using 3D texture mapping; offline rendering is demonstrated using a RenderMan shader.

Futhermore, we show that the acquired (per-texel) data can be approximated and compressed by an exponential function. This results in six parameters per texel. Even real-time rendering is possible from the compressed data through the use of dependent texture lookups.

2 Prior Work

The acquisition, representation, and rendering of realistic materials are active areas of research in the computer graphics community. We will briefly review related work in these areas.

Light fields store the light incident from all directions and for every point on a two-dimensional surface. This surface can either be in space [8, 16] or coincide with an object's surface [1, 2, 24, 31]. Light fields do not allow to change the illumination, only the viewpoint can be changed. Rendering of light fields works in real-time as graphics hardware can be used [1, 2, 8, 16].

In contrast to light fields, a reflectance field does allow to change the illumination. Many different techniques have been proposed to acquire reflectance fields. All these methods require the acquisition of many images, since the reflectance field is six-dimensional. Dana et al. [3] show how to acquire bidirectional texture functions, which basically correspond to a reflectance field but constrained to a plane. A partial reflectance field (fixed view) can be stored efficiently in a polynomial texture map [18]; this representation also allows for fast rendering. We will apply a similar representation to compress our data. Debevec et al. [4] describe a method for acquiring the reflectance field of human faces. Recently, the reflectance fields of full and even partially transparent objects have been acquired [7, 19, 20]; rendering takes several seconds though. All reflectance field acquisition methods require an automated setup and produce a massive amount of data, which is usually alleviated by applying compression.

BRDFs have been acquired by various researchers. Early work [14, 30] was constrained to homogeneous BRDFs. Later on, spatially varying BRDFs were acquired [15, 21, 28, 32, 33]. All these methods fit the measured data to certain BRDF models, hence requiring sparser sampling. This is also the main difference to reflectance field acquisition, where the data is not fitted to a certain reflectance model but rather used directly. Nonetheless, BRDF acquisition is also a very time-consuming task, often taking several hours [15, 21].

Real-time rendering of realistic materials is fairly well-researched [11, 12, 13, 22, 23]. All these methods show how to render with interesting BRDFs. Most of these methods have certain restrictions, such as rendering homogeneous materials only. Recent work [13, 22] allows to render with spatially varying BRDFs, i.e., they can render directly from the acquired and fitted BRDF data. Our rendering algorithm differs significantly from these methods since the material's spatial variation is not encoded as a BRDF.

Also recently, BTF rendering was shown to work in real-time by applying compression first and directly rendering from the compressed data [29, 25]. Unlike these methods, our technique does not require a full BTF.

3 Acquisition

The acquisition and the preprocess is very simple, resulting in a fast acquisition procedure.

3.1 Setup and Acquisition

The setup is illustrated in the upper part of Figure 2. A camera is looking orthogonally at the material to be acquired (assuming a distant viewer). The material has to be flat. We then illuminate the surface with a parallel light source (we use a distant point light source) at different incident elevation angles. The light source is not rotated around the surface normal, i.e. the light source only moves along an arc.

The light source elevation angles are (roughly) spaced in 10° steps, i.e. we acquire 10 images for every material. If the resulting material is to be used in a global illumination rendering algorithm, one should capture high-dynamic-range images [5]. For real-time rendering, even low-dynamic-range images work fine, as can be seen in the results section.

In Figure 2, you can see the acquired images for a few different light source angles. We acquired all images manually but even then the acquisition of a material takes only a few minutes, a big advantage over other methods [3, 15, 19, 29].

3.2 Preprocess

During rendering, we will index into our stack of images using r (BRDF times cosine). The question is which of the acquired images corresponds to a certain r-value. To this end, the average radiance a_i of each image i is computed. We then resample the stack of images so that the average radiance increases linearly. This is done by first computing what the average radiance r_i for slice i should be:

$$r_i = \frac{i \cdot a_{\max}}{N-1},\ i = 0..N-1, \tag{1}$$

where $a_{\max}$ is the maximum acquired average radiance and N is the number of acquired slices. For each slice i we now take the two original images whose average radiance a_i is closest to the desired radiance r_i and linearly interpolate between those two images accordingly. The resulting stack of new images is called the *shading map*, see Figure 3. From now on, the values stored in the shading map are interpreted as BRDF times cosine values. This basically assumes that the material was illuminated with a light source of unit intensity.

4 Rendering

As stated before, rendering with our data is fairly simple. We will describe the necessary computation for one pixel only; obviously it has to be repeated for every visible pixel. We assume that for a pixel, we are given local

Figure 3: Shading Map. This set of images forms the shading map. In a shading map the average exit radiance is increasing linearly for every image.

view and light directions $(\hat{\omega}_o, \hat{\omega}_i)$, as well as a set of texture coordinates (s, t).

First, we compute the value r by evaluating a BRDF model f_r (it can be any BRDF model):

$$r(\hat{\omega}_o, \hat{\omega}_i) \quad := \quad \sigma f_r(\hat{\omega}_o, \hat{\omega}_i)\hat{\omega}_{i_z}. \tag{2}$$

The value σ is used to adjust the shade of the material. It is needed if the acquisition is not done with a unit light source or if only low-dynamic range images are captured.

Now we compute exit radiance by integrating over the incident light multiplied with the value $S(r)$ from our shading map:

$$L_o(\hat{\omega}_o) \quad = \quad \int_\Omega S\left(r(\hat{\omega}_o, \hat{\omega}_i)\right) L_i(\hat{\omega}_i)\, d\hat{\omega}_i, \tag{3}$$

where $L_i(\hat{\omega}_i)$ is the incident radiance from direction $\hat{\omega}_i$. Using the value $r(\hat{\omega}_o, \hat{\omega}_i)$, we do a lookup into our shading map using the operator $S()$. It performs the lookup with the coordinates $(s, t, r/a_{\max} \cdot (N-1))$, where (s, t) signifies the position in the map, and $r/a_{\max} \cdot (N-1)$ is the layer to be chosen (as before $a_{\max}$ is the maximum acquired value and N the number of slices). Trilinear filtering of the shading map samples should be used to avoid blockiness and contouring artifacts.

There is one special case that needs to be treated. If the required value r is higher than the maximum average value $a_{\max}$ in our shading map, we can either clamp the result at $a_{\max}$ or scale the brightest image in our shading map to the desired level r_i. The smallest average radiance r_0 is always zero (see Equation 1), and no special treatment is required.

4.1 Real-Time Rendering

The algorithm for real-time rendering using graphics hardware is not much different. We load the shading map as a 3D volume onto the graphics hardware. In a vertex shader we compute $r(\hat{\omega}_o, \hat{\omega}_i)$ (for a single point light source). The result of the vertex shader is a set of texture

coordinates $(s, t, r/a_{\max})$ which is then used to lookup into the shading map. A pixel shader performs the multiplication with the intensity of the light source.

Newer graphics hardware such as the ATI Radeon 8500 and 9700, as well as the NVIDIA GeForce FX support dependent texture lookups in the pixel shaders. With this feature, it is possible to compute the BRDF (times cosine) on a per-pixel basis and then do a dependent texture lookup into the shading map, which avoids undersampling artifacts that may occur when using a vertex shader.

Unfortunately, not all graphics hardware supports mipmapping of 3D textures, which can lead to visible aliasing.

5 Compression

The storage cost of the acquired data is fairly high since in our experience at least 8 slices are necessary in the shading map. Fortunately, the data can be compressed quite well.

Since most reflectance models are based on power functions, we also use a power function to approximate the change of radiance of a single texel in the shading map (unlike, e.g., polynomial texture maps [18], which use a fixed polynomial). Given all the radiance samples of a single texel s_i with $i = 0..N-1$, we approximate them with:

$$f(r) = k_s \cdot r^K, \tag{4}$$

where $k_s = s_{(N-1)}$. The exponent K is simply tested in a brute force manner. We first check different K with a large step-size and then do a second pass in the neighborhood of the minimum with a small step-size. For a $512 \times 512 \times 8$ data set, this fitting procedure only takes about 40 seconds on a P4 1.7Ghz, so we decided not to use a more elaborate fitting scheme.

It should be noted, that we do this fitting separately for each color channel, which drastically increases the quality of the fit. So for every texel we get two three-channel parameters k_s and K resulting in a total of six parameters.

Rendering from the compressed data is obviously simple for offline rendering. Hardware accelerated rendering can be also implemented if either exponentiation is supported in the pixel shader (both NVIDIA's GeForce FX and ATI's Radeon 9700 do so) or if dependent texture lookups are supported, with which one can implement arbitrary functions, including exponentiation, as a table lookup [13, 26].

6 Results

First we would like to show several renderings with acquired materials. All materials were acquired using ten different light source positions. The images were cropped to 1024×1024 and then downscaled to 512×512. In order to be compatible with graphics hardware, which usually requires power-of-two texture sizes, we resampled these ten images down to eight images during linearization. The linearization takes about 5 seconds for each set of images. For all renderings we have used the modified Phong model [17].

In Figure 6, you can see offline renderings computed with a RenderMan shader [10] and the Blue Moon Rendering Toolkit [9]. One image takes about 3 minutes to compute on a P4 1.7 Ghz at a resolution of 800×800 with 4 samples per pixel. Renderings were done with different techniques. The left-most images are just using a single slice of the shading map scaled by $r(\hat{\omega}_o, \hat{\omega}_i) L_i(\hat{\omega}_i)$, i.e., the slice serves as a texture which is multiplied with the BRDF. The second column uses the final $L_o(\hat{\omega}_i)$ instead of $r(\hat{\omega}_o, \hat{\omega}_i)$ for the lookup, performing the lookup outside the integral (see Equation 3). This makes rendering faster because of fewer lookups. The third column uses the angle between incident light direction and the normal for the lookup into the shading map; multiplication with the incident radiance and the BRDF is performed afterwards. The fourth column uses our proposed model (lookup with $r(\hat{\omega}_o, \hat{\omega}_i)$ and multiplication with light intensity).

As can be seen, the scaled texture map performs badly for specular materials, like plastic. The second technique is highly dependent on the light source intensity and does not work well for highly specular materials. But also more diffuse materials, such as the cloth, tend to look odd. The third and fourth techniques produce fairly similar results. But at grazing angles (note the walls) the simple angular based lookup fails to reproduce highlights, unlike our method.

The top two rows compare the same material rendered with two different BRDFs. The first image uses a Phong exponent of 5, whereas the second image uses a Phong exponent of 60. Both rows use the same input data. This shows how the appearance of a material can be adapted later on.

In Figure 7 you can see different renderings done with our real-time implementation. The shown materials are jeans, suede, leather, pressed wood, wax, nylon, cloth, and wool. All renderings are done with the enhanced Phong BRDF.

Figure 4 shows a rendering of a sweater made out of wool. The left image was done with a full BTF (6500 images, compressed using PCA with 16 components). The right image was done with our technique. Differences are mainly visible at grazing angles, where masking effects become important, which cannot be reproduced by our

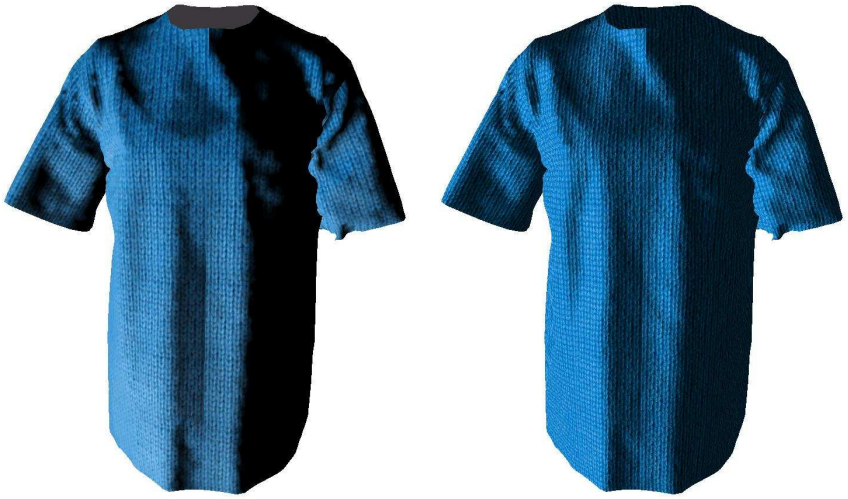

Figure 4: Comparison between BTF rendering (left) and our method (right).

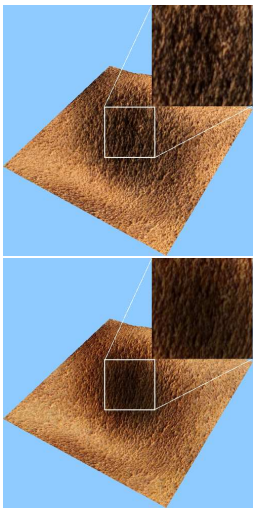

Figure 5: Comparison between original data (top) and compressed data (bottom).

technique. This also results in slight color differences (although some color difference is also caused by the compression). Considering that wool is one of the materials which violates our initial assumptions (it has a larger-scale structure and is anisotropic), our method does fairly well.

In Table 1, some RMS approximation errors are listed for our compression technique. The error is always in the same order of magnitude. In Figure 5 you can see a visual comparison between original data (suede) and approximated data. Some differences are visible, but this example shows the worst case we found. In general, the visual difference is rather small.

Our tests and results show that our acquisition technique works well for many kinds of materials, even though the actual BRDF is not captured.

	suede	plastic	wood	stone
RMS	0.064878	0.049011	0.046052	0.040058

Table 1: RMS errors for our compression technique.

6.1 Discussion

The main question is, why is it sufficient to acquire images for a fixed view and several incident illumination directions only? As derived in the introduction, it is not sufficient if you want to acquire a real BRDF. However, we want to acquire only the change in illumination due to a material's surface structure and for this purpose it is sufficient. The actual BRDF is supplied by the user.

We believe that this technique works so well because a human observer expects a certain kind of variation in materials. Judging from the results, this variation is captured by our technique and gives a good impression of the actual materials.

Generally speaking, materials with small-scale variations, such as the shown cloth, suede, jeans, and so on, work best. Materials with spatially varying reflectance properties (different specularites, etc.) are captured well too, as can be seen in the candle wax and pressed wood example.

For certain materials though, this technique works less well. E.g., the acquisition of highly specular materials is problematic, mainly because the use of a point light source as an approximation to a parallel light source. As stated in the introduction, our method assumes materials with a fine-scale structure. This is because our technique only models variation with the incidence angle not with the azimuth. Therefore, shadows from larger structures cannot be captured/represented well, as they always appear in the same direction. Furthermore, the materials are assumed to be isotropic.

Even for materials violating these assumptions, our technique might be still suitable for a fast preview, since it is much better than normal texturing.

7 Conclusions and Future Work

We have presented a method that enables the acquisition of complex spatially varying materials using a few images only. This empirical method does not capture the actual BRDF but only how illumination changes due to fine surface structure; the BRDF can be adapted later.

Real-time and offline rendering can be easily implemented. Storage cost can be minimized by a lossy com-

pression technique based on power functions.

The technique has been validated with many materials achieving good results. Again it should be noted, that this method is empirical. It works well on all the materials we have tried, but the results are not an accurate reproduction of reality.

The acquired data is easily amenable to texture synthesis [6], i.e., small acquired surface patches can be enlarged using these synthesis methods — just with higher-dimensional sample data. Some of our shown materials were made tileable this way.

This work shows that the perception of materials depends on multiple factors. The actual BRDFs are only part of this. The fine surface structure also conveys information about the material. Future work should be concerned with the perception of BRDFs and materials.

Acknowledgements

We would like to thank the University of Tübingen for the sweater model, Christian Rössl for the parameterization of the models, and Stanford University for the bunny model. Furthermore, we would like to thank the anonymous reviewers for their valuable comments.

References

[1] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured Lumigraph Rendering. In *Proceedings SIGGRAPH*, pages 425–432, August 2001.

[2] W.-C. Chen, J.-Y. Bouguet, M. H. Chu, and R. Grzeszczuk. Light Field Mapping: Efficient Representation and Hardware Rendering of Surface Light Fields. In *Proceedings SIGGRAPH*, pages 447–456, July 2002.

[3] K. Dana, B. van Ginneken, S. Nayar, and J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics*, 18(1):1–34, January 1999.

[4] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar. Acquiring the Reflectance Field of a Human Face. In *Proceedings SIGGRAPH*, pages 145–156, July 2000.

[5] P. Debevec and J. Malik. Recovering High Dynamic Range Radiance Maps from Photographs. In *Proceedings SIGGRAPH*, pages 369–378, August 1997.

[6] A. Efros and T. Leung. Texture Synthesis by Non-Parametric Sampling. In *IEEE International Conference on Computer Vision*, pages 1033–1038, September 1999.

[7] R. Furukawa, H. Kawasaki, K. Ikeuchi, and M. Sakauchi. Appearance based object modeling using texture database: Acquisition compression and rendering. In *Thirteenth Eurographics Workshop on Rendering*, pages 267–276, June 2002.

[8] S. Gortler, R. Grzeszczuk, R. Szelinski, and M. Cohen. The Lumigraph. In *Proceedings SIGGRAPH*, pages 43–54, August 1996.

[9] L. Gritz and J. Hahn. BMRT: A Global Illumination Implementation of the RenderMan Standard. *Journal of Graphics Tools*, 1(3):29–47, 1996.

[10] P. Hanrahan and J. Lawson. A Language for Shading and Lighting Calculations. In *Proc. SIGGRAPH*, pages 289–298, August 1990.

[11] W. Heidrich and H.-P. Seidel. Realistic, Hardware-accelerated Shading and Lighting. In *Proceedings SIGGRAPH*, pages 171–178, August 1999.

[12] J. Kautz and M. McCool. Interactive Rendering with Arbitrary BRDFs using Separable Approximations. In *Tenth Eurographics Workshop on Rendering*, pages 281–292, June 1999.

[13] J. Kautz and H.-P. Seidel. Towards Interactive Bump Mapping with Anisotropic Shift-Variant BRDFs. In *Eurographics/SIGGRAPH Hardware Workshop*, pages 51–58, August 2000.

[14] E. Lafortune, S. Foo, K. Torrance, and D. Greenberg. Non-Linear Approximation of Reflectance Functions. In *Proceedings SIGGRAPH*, pages 117–126, August 1997.

[15] H. Lensch, J. Kautz, M. Goesele, W. Heidrich, and H.-P. Seidel. Image-Based Reconstruction of Spatially Varying Materials. In *12th Eurographics Workshop on Rendering*, pages 103–114, June 2001.

[16] M. Levoy and P. Hanrahan. Light Field Rendering. In *Proceedings SIGGRAPH*, pages 31–42, August 1996.

[17] R. Lewis. Making shaders more physically plausible. In *4th Eurographics Workshop on Rendering*, pages 47–62, June 1993.

[18] T. Malzbender, D. Gelb, and H. Wolters. Polynomial Texture Maps. In *Proceedings SIGGRAPH*, pages 519–528, August 2001.

[19] W. Matusik, H. Pfister, A. Ngan, P. Beardsley, and L. McMillan. Image-Based 3D Photography Using Opacity Hulls. In *Proceedings SIGGRAPH*, pages 427–437, July 2002.

[20] W. Matusik, H. Pfister, R. Ziegler, A. Ngan, and L. McMillan. Acquisition and Rendering of Transparent and Refractive Objects. In *Thirteenth Eurographics Workshop on Rendering*, pages 277–288, June 2002.

[21] D. McAllister. *A Generalized Representation of Surface Appearance*. PhD thesis, University of North Carolina, 2002.

[22] D. McAllister, A. Lastra, and W. Heidrich. Efficient Rendering of Spatial Bi-directional Reflectance Distribution Functions. In *Proceedings Graphics Hardware*, September 2002.

[23] M. McCool, J. Ang, and A. Ahmad. A Homomorphic Factorization of BRDFs for High-Performance Rendering. In *Proceedings SIGGRAPH*, pages 171–178, August 2001.

[24] G. Miller, S. Rubin, and D. Ponceleon. Lazy Decompression of Surface Light Fields for Precomputed Global Illumination. In *9th Eurographics Workshop on Rendering*, pages 281–292, June 1998.

[25] G. Müller, J. Meseth, and R. Klein. Compression and Real-Time Rendering of Measured BTFs Using Local PCA. In *Proceedings of Vision, Modeling and Visualisation 2003*, November 2003.

[26] M. Peercy, M. Olano, J. Airey, and P. J. Ungar. Interactive Multi-Pass Programmable Shading. In *Proceedings SIGGRAPH*, pages 425–432, July 2000.

[27] E. Praun, H. Hoppe, M. Webb, and A. Finkelstein. Real-Time Hatching. In *Proc. SIGGRAPH*, pages 579–584, August 2001.

[28] Y. Sato, M. Wheeler, and K. Ikeuchi. Object Shape and Reflectance Modeling from Observation. In *Proceedings SIGGRAPH*, pages 379–388, August 1997.

[29] M. Sattler, R. Sarlette, and R. Klein. Efficient and Realistic Visualization of Cloth. In *Eurographics Symposium on Rendering 2003*, pages 167–178, June 2003.

[30] G. Ward Larson. Measuring and Modeling Anisotropic Reflection. In *Proceedings SIGGRAPH*, pages 265–272, July 1992.

[31] D. Wood, D. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. Salesin, and W. Stuetzle. Surface Light Fields for 3D Photography. In *Proceedings SIGGRAPH*, pages 287–296, July 2000.

[32] Y. Yu, P. Debevec, J. Malik, and T. Hawkins. Inverse Global Illumination: Recovering Reflectance Models of Real Scenes From Photographs. In *Proc. SIGGRAPH*, pages 215–224, August 1999.

[33] Y. Yu and J. Malik. Recovering Photometric Properties of Architectural Scenes from Photographs. In *Proceedings SIGGRAPH*, pages 207–218, July 1998.

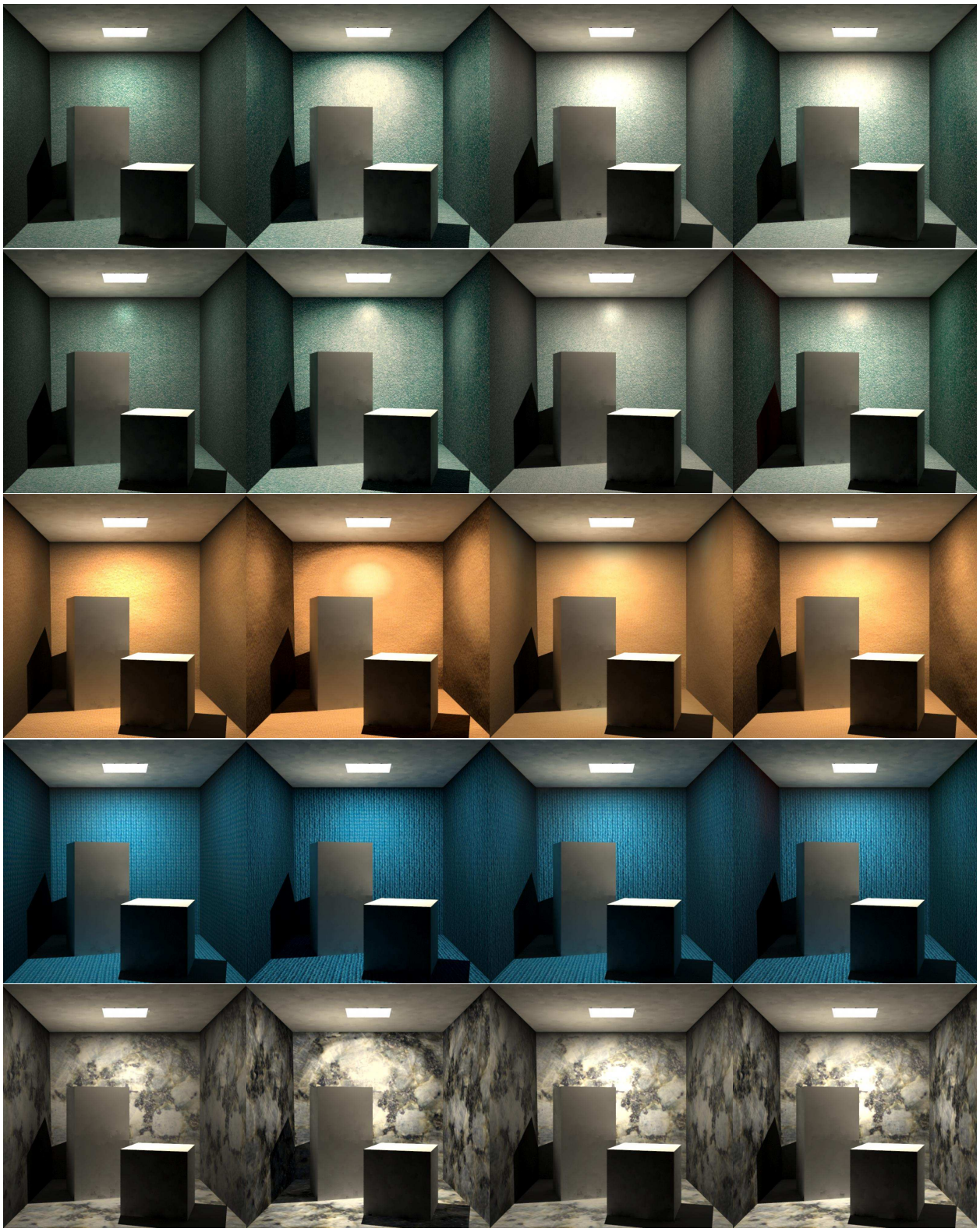

Figure 6: Left to right: lit texture, lookup with L_i, angular lookup, lookup with r. Top to bottom: plastic with $N = 5$, plastic with $N = 60$, cloth, wool, stone.

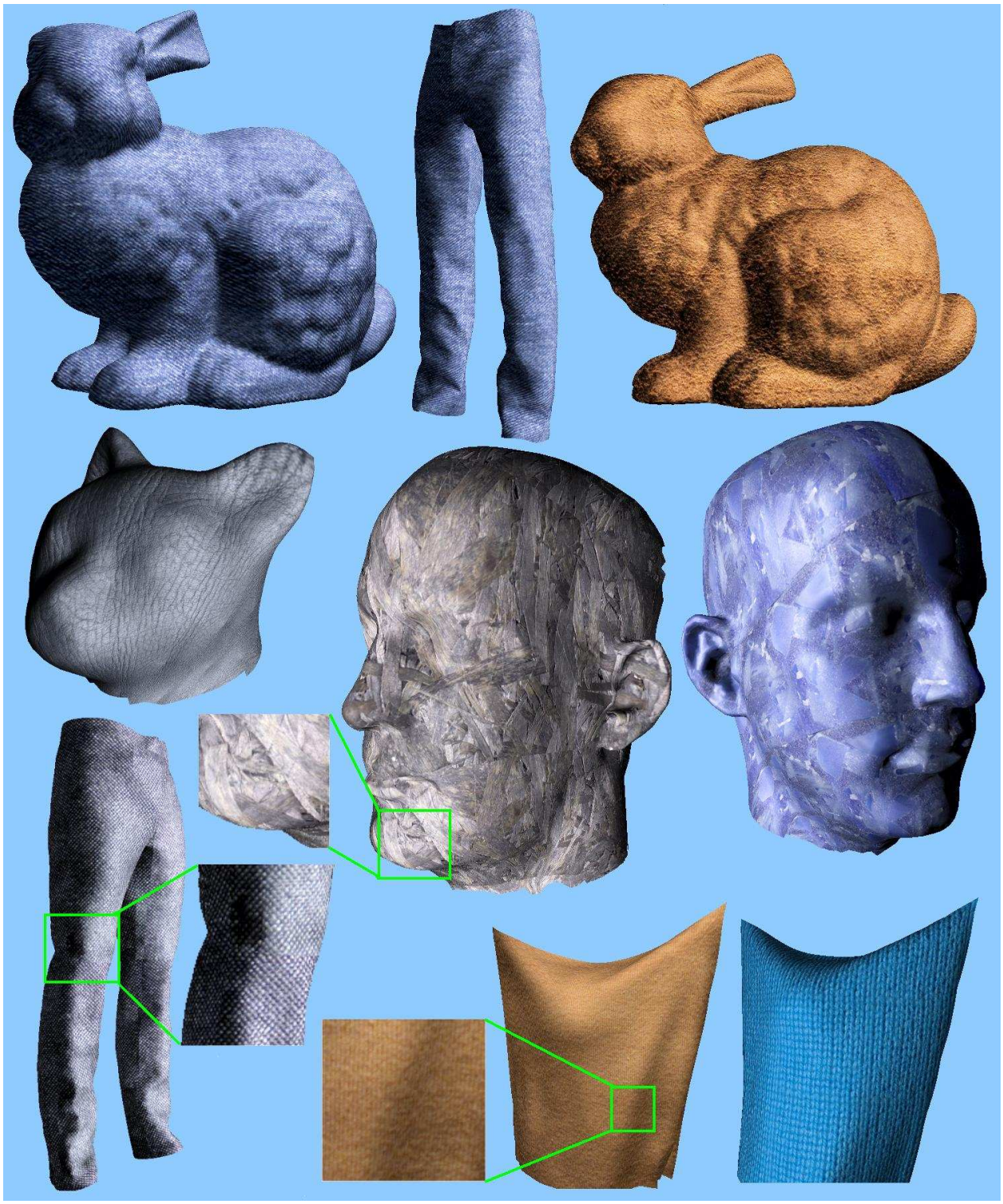

Figure 7: Different models and materials. The shown materials are jeans, suede, leather, pressed wood, wax, nylon, cloth, and wool. All models are rendered in real-time.