Synthesizing Bidirectional Texture Functions for Real-World Surfaces

Xinguo Liu [†‡] **Yizhou Yu** [††] **Heung-Yeung Shum** [†]

[†] Microsoft Research, China
[††] University of Illinois at Urbana-Champaign
[‡] State Key Lab. of CAD&CG, Zhejiang University

Abstract

In this paper, we present a novel approach to synthetically generating bidirectional texture functions (BTFs) of real-world surfaces. Unlike a conventional two-dimensional texture, a BTF is a six-dimensional function that describes the appearance of texture as a function of illumination and viewing directions. The BTF captures the appearance change caused by visible small-scale geometric details on surfaces. From a sparse set of images under different viewing/lighting settings, our approach generates BTFs in three steps. First, it recovers approximate 3D geometry of surface details using a shape-from-shading method. Then, it generates a novel version of the geometric details that has the same statistical properties as the sample surface with a non-parametric sampling method. Finally, it employs an appearance preserving procedure to synthesize novel images for the recovered or generated geometric details under various viewing/lighting settings, which then define a BTF. Our experimental results demonstrate the effectiveness of our approach.

CR Categories: I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding—modeling and recovery of physical attributes I.3.7 [**Computer Graphics**]: Three-dimensional Graphics and Realism—color, shading, shadowing, and texture I.4.8 [**Image Processing**]: Scene Analysis—color, photometry, shading

Keywords: Bidirectional Texture Functions, Reflectance and Shading Models, Texture Synthesis, Shape-from-Shading, Photometric Stereo, Image-Based Rendering.

1 Introduction

Surface appearance modeling has drawn much attention from researchers since the dawn of computer graphics [6, 2, 3]. Appearance models are closely related to geometry. There are three levels of scales in geometry, namely, the macrostructure level, the mesostructure level [20] and the microstructure level. A geometric model usually refers to the macrostructure level, and is often specified as a set of polygonal and/or curved surfaces. The mesostructure level includes geometric details that are relatively small but still visible such as bumps and dents on a concrete surface. The microstructure level involves surface microfacets that are visually indistinguishable by human eyes. The last two levels of geometry contribute to surface appearance properties. For instance, bump maps are used to model the mesostructure level while BRDFs model the microstructure level.

We are interested in modeling appearance at the mesostructure level for real-world surfaces such as concrete surfaces, crumpled papers, pebbles and carpets. The presence of such small-scale details gives rise to a rich set of visual effects, including mutual shadowing, interreflection, occlusion and foreshortening, in addition to varying surface normal orientations. Without properly modeling such effects, surfaces would look too smooth to be real. Bump and normal mapping techniques can model the effects caused by changing normal orientations but not others. However, all the above visual effects for bumpy surfaces (as well as spatially varying reflectance) can be captured by bidirectional texture functions (BTFs).

A BTF is defined as a six dimensional function with a 2D texture associated with each possible combination of lighting and viewing directions which account for the other four dimensions. Thus, a BTF has two additional dimensions for textures compared to a 4D BRDF. The pioneering work by Dana et. al. [7, 9] on BTFs took an experimental approach that acquired images of material samples under various combinations of lighting and viewing directions. Their work led to the CUReT database that has a sparse set of images partially covering the lighting and viewing hemispheres for each material sample. Such a sparse sampling is not adequate to faithfully represent material appearances for graphical rendering purposes. On the other hand, acquiring a dense set of samples for BTFs is prohibitive in practice because a BTF has six dimensions.

In this paper, we study the following problem: given discrete samples of the BTF of a real-world surface with mesostructure details, can we synthesize the continuous BTF ? Specifically, from a sparse set of sample textures, can we synthesize a new texture at any given lighting/viewing setting ? Moreover, can novel BTFs be synthesized from a given BTF to emulate the stochastic properties of texture images of the given BTF under all lighting/viewing settings, similar to 2D texture synthesis ?

We present an algorithm to solve the above problems, by exploiting both geometric and photometric properties of material samples and effectively integrating them together. In our work, we use sample textures from the CUReT database. We first recover the height field on a material from a small number of images and synthesize novel 3D structures for the same material. The recovered or synthesized height fields are used for rendering synthetic images of the material under different combinations of lighting and viewing

[†] hshum@microsoft.com
[††] yizhouy@acm.org, www-faculty.cs.uiuc.edu/∼yyz
[‡] Xinguo Liu participated in this work when he was visiting Microsoft Research, China.

directions. The rendered images are then fed into an appearance preserving texture synthesis procedure, along with the set of acquired sample images, to synthesize high-quality sample images of the corresponding BTF.

In summary, this paper has the following three major contributions.

- First of all, we propose a novel hybrid approach for studying appearance models, which can be a useful idea for bridging other geometry-based and image-based techniques.
- Second, we introduce an algorithm that synthesizes complete BTFs, including the statistical structure and statistical texture, from a sparse set of sample images.
- Third, we develop a method to recover bump maps from photographs of real world materials by adapting an existing shape-from-shading algorithm.

The remainder of this paper is organized as follows. The next section provides the necessary background and related work. Section 3 gives an overview of our algorithm. The details of our algorithm will be discussed in Section 4 (geometry recovery), Section 5 (geometry synthesis) and Section 6 (BTF synthesis). And Section 7 presents our results.

2 Background and Related Work

2.1 BTFs

A BTF can be regarded as a mapping from the 4D space of lighting and viewing directions to the space of all 2D images:

$$\tau : L \times V \rightarrow I \tag{1}$$

where L and V are lighting and viewing directions parameterized by a pair of tilt and azimuth angles (θ, ϕ), I is a mapping itself from R^2 to the RGB color space. This definition basically views a BTF as a collection of images, and favors texture analysis and synthesis.

We assume every image in a BTF observes a homogeneous Markov Random Field (MRF) model, which is a common assumption in the texture synthesis literature [12, 38, 41, 45]. Because of the visual effects caused by varying lighting and viewing directions, each image in a BTF has a distinct MRF model. MRF methods model an image as a realization of a local and stationary random process. That is, each pixel of a texture image is characterized by a small set of spatially neighboring pixels, and this characterization is the same for all pixels. A MRF model allows us to view every BTF image as a collection of local neighborhoods. Each realization of the MRF model can be viewed as a random rearrangement of these local neighborhoods in the image plane. We also assume that the height field inducing the BTF on a material sample observes a homogeneous MRF model which enables us to synthesize novel instances of the height field using existing texture synthesis algorithms [12, 38].

Alternatively, we can follow the line of previous work on light fields and plenoptic functions [1, 23, 14], and consider a BTF as a specific 6D reflectance field if ignoring wavelength and fixing time:

$$T = T(\theta_i, \phi_i, x, y, \theta_r, \phi_r) \tag{2}$$

which provides the connection between reflected flux in a direction (θ_r, ϕ_r) and incident flux in another direction (θ_i, ϕ_i) at the same point (x, y) on a material sample. This is a simplified version of a more general 8D reflectance field in [10] for a general 3D object enclosed by a convex hull by assuming parallel light sources.

Since a BTF includes images of a material under all possible lighting directions, it essentially provides lighting-independent appearance properties of the material. Novel images of the material under an arbitrary lighting condition can be synthesized from its BTF by properly integrating the contribution from each individual BTF image.

2.2 Related Work

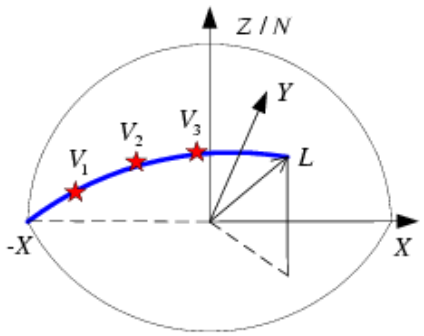


Figure 1: The sampled lighting and viewing directions of the BTF images in the CUReT database. Three typical viewpoints (V_1, V_2 V_3) are shown in between the lighting direction L and the negative X-axis.

Previous work on BTFs aims to capture appearance data for natural materials and represent them efficiently [7, 8, 9, 22]. However, each material in the CUReT database [7] has a sparse 4D sampling of only 205 images under 205 different viewing and lighting conditions. Specifically for each lighting direction L on the half hemisphere with $Z > 0$ and $Y < 0$, seven images were captured at viewpoints along the short spherical arc between the lighting direction and the negative X-axis. Fig. 1 illustrates a few such viewpoints.

Although [18] introduces a technique to precompute some of the visual effects for regular synthetic bump structures, stochastic details on real-world materials are not modeled.

Part of our work is inspired by the recent success of 2D texture synthesis [17, 4, 36, 12, 38, 41, 45]. In 2D texture synthesis, from a texture sample, a new texture is synthesized such that, when perceived by a human observer, it appears to be generated by the same underlying stochastic process. To the best of our knowledge, however, there has been no previous effort on synthesizing 6 dimensional BTFs where textures are under different stochastic processes with different viewing/lighting settings.

Our work also shares similarities with previous work on image-based rendering in that novel images of a real scene are generated from acquired image samples, without [25, 23, 14, 37, 34, 35] or with [11, 33, 31, 32, 44, 43, 42] the knowledge of the scene geometry. In particular, our 6D BTF synthesis problem has a similar spirit as plenoptic modeling where a continuous 5D plenoptic function is synthesized from discrete samples. Note that BTFs are different from surface light fields [39] and view-dependent textures [11] since the latter two only capture surface appearance under fixed lighting conditions. Appearance models from real images have become an active research topic in graphics [31, 32, 24, 44, 43, 10].

3 Overview

For the rest of the paper, a *viewing/lighting setting* means a combination of viewing and lighting directions. The set of input images to our method are called the *sample images* of a real material.

Two factors affect the appearance of bumpy surfaces: the 3D structure of the bumps and spatial reflectance variations. The geo-

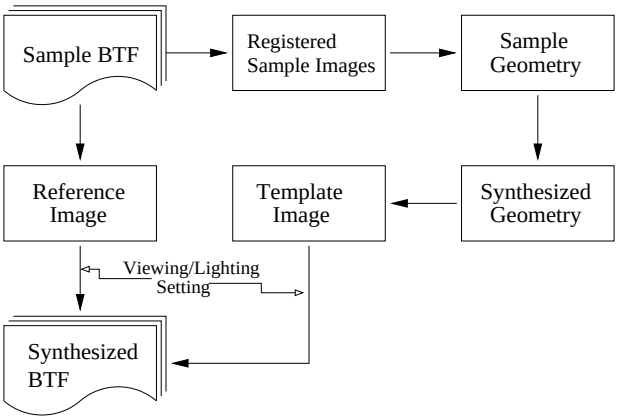


Figure 2: The flow chart of the overall BTF synthesis algorithm.

metrical structure produces shadows and occlusions. Both factors together generate texture and shading effects including highlights. The first step in our approach is to recover the surface geometry. However, the statistical properties of the 3D structure are more important than the exact geometry itself because of the random spatial distribution of the 3D features.

We represent the surface geometry using a height field on top of a supporting plane of the surface. From a collection of images taken under different viewing/lighting settings, we derive the geometry and its approximate albedo map using a modified version of the height from shading method in [21]. The modified method can recover depth variations by incorporating multiple input images in the formulation. More importantly, we explicitly handle shadows and possible highlights in the images. These input images are first globally registered using the video mosaicing technique developed in [37].

From the recovered height fields we can synthesize other statistically equivalent height fields that can then be used to generate new BTFs. By considering the height field as a sample (gray-scale) image, we can apply previously developed 2D texture synthesis algorithms to synthesize novel height fields. To synthesize an accurate BTF image from a novel viewing/lighting setting, one possibility could be texture mapping the recovered height field from the sample images followed by viewing/lighting dependent interpolation. Unfortunately, there are two issues here. First, for the recovered height field, the sample images may be irregularly distributed so that there is no neighboring image for certain viewing/lighting settings. Second, for the synthesized novel height field, we do not have any sample images at all. So simply interpolating from neighboring images becomes infeasible. Instead, we propose a local appearance preserving texture synthesis procedure.

Our BTF synthesis algorithm can synthesize a BTF image from two input images. One is a *template image* which is synthetically rendered with shadows from the input height field. The other is a *reference image* which is selected from the set of real sample images of the material. Both images should have the same viewing/lighting setting as that of the BTF image being synthesized. Either a constant albedo or the recovered albedo map can be used for rendering the template image. This template image exhibits correct shadows, occlusions for the synthesized height field and approximate shading effects associated with the material. But it does not have very accurate color, shading and mutual illumination at each point since we do not have point-wise 4D reflectance functions on the material surface at this stage. On the other hand, the reference image has the correct color and shading information. But its underlying geometry may be different from our input height field. Our method can combine the useful information from these two images. The flow chart of our BTF synthesis algorithm is illustrated in Fig. 2.

The technique of copying texture from the reference image is partially inspired by the block-copy synthesis method [41]. Compared with the pixel-wise synthesis scheme adopted in some previous 2D texture synthesis algorithms, the block-copy synthesis scheme is much faster while maintaining feature integrity. However, one must be careful about choosing the right places and order to paste blocks. Those regions with prominent features, such as corners and edges, should be synthesized first with higher priority.

Ideally, the reference image for our texture synthesis should have the same viewing and lighting directions as the template image. Such a reference image may not be available because of the limited size of the input image collection. Our solution is to find the "nearest" image, and then make it consistent with the desired viewing/lighting setting by warping. A more serious problem is that the sample image collection may not be distributed uniformly in the 4D space for lighting and viewing directions. A subspace may not have any corresponding images at all. When this happens, we exploit a certain level of isotropy exhibited by the material samples. A material sample has a large number of tiny bumps. Being "isotropic" means that for any bump B_i and any rotation angle between 0 and 360 degrees, there always exists another bump B_j whose shape and reflectance are rotated versions of B_i by the given angle. Most natural materials approximately satisfy this condition except for very elongated structures such as straw. Although a single "nearest" reference image is enough to synthesize a BTF image, the resulting image may be noisy. More reference images can be used to further improve the synthesis quality.

With the above introduction to our method, we can summarize the *requirements* we need to impose on the set of input sample images: a) The images should cover a reasonable number of random bumpy structures of a material for the MRF model to work well; b) for anisotropic materials, we need a sparse set of images covering the 4D space of viewing and lighting directions; c) for isotropic materials, we only need images covering a 3D subspace of the viewing/lighting settings since the azimuth angle of the lighting direction is no longer important.

4 Geometry Recovery

Geometry recovery is a classic problem in computer vision. Many kinds of geometry recovery algorithms have been proposed using different visual cues such as disparity, shading, focus and defocus [13]. We decided to use shading as the major cue to account for shading variations, which are present in the set of input sample images because of a changing illumination direction.

4.1 Shape from Shading

We adopt a shape-from-shading technique called height from shading [21]. Unlike most shape-from-shading and photometric stereo methods [19, 5, 40, 27], this technique computes a height field directly rather than through surface normals. This technique is efficient and robust. The geometry recovery is formulated to minimize the following energy functional [21]:

$$E = \sum_{i,j} [\alpha(\rho R(p_{ij}, q_{ij}) - I(i,j))^2 + \lambda(u_{ij}^2 + v_{ij}^2)] \quad (3)$$

where ρ is the surface albedo, I is the observed image intensity, p_{ij}, q_{ij}, u_{ij}, v_{ij} are the symmetric first and second finite differences of the surface height field $\{z_{ij}\}$, α and λ are two constant coefficients, and R is the Lambertian reflectance model:

$$R(p_{ij}, q_{ij}) = \vec{n_{ij}} \cdot \vec{L} = \frac{x_L p_{ij} + y_L q_{ij} - z_L}{\sqrt{p_{ij}^2 + q_{ij}^2 + 1}} \quad (4)$$

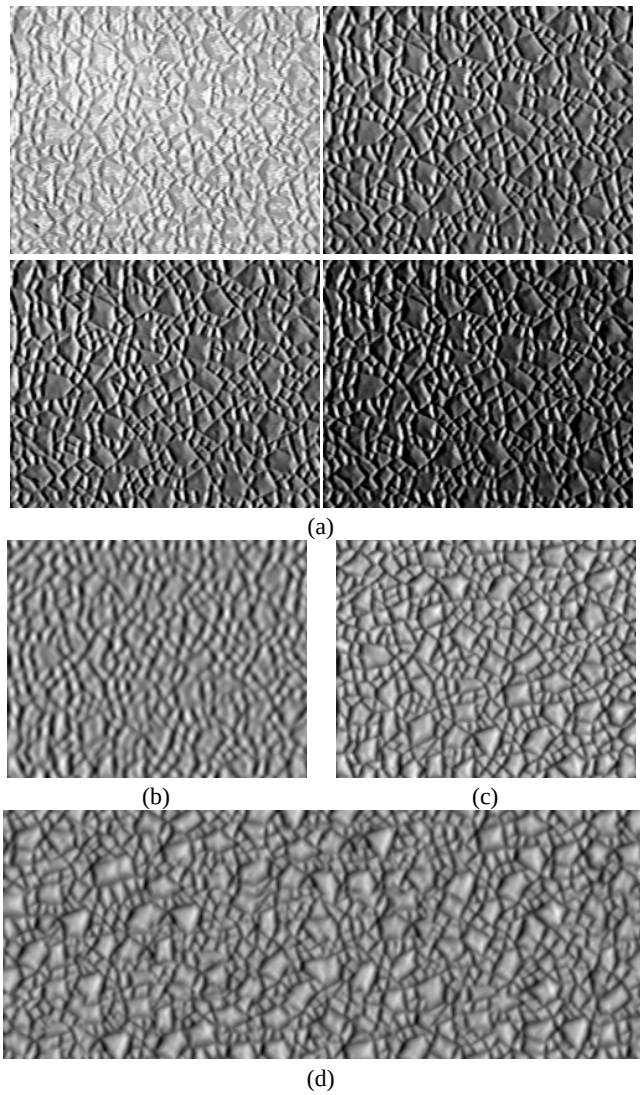


Figure 3: (a) Four calibrated gray-scale images used for recovering a height field; (b) a recovered height field from the height-from-shading algorithm in [21]; (c) a recovered height field from our revised algorithm; (d) a synthesized height field generated from (c). Height fields are visualized as gray-scale images in (b)-(d).

where $L = (x_L, y_L, z_L)$ is the unit vector of the light source direction.

The first term in Eq. 3 corresponds to the photometric error term. And the second is a regularization term on the smoothness of the surface. These two terms are balanced by two weights α and λ.

4.2 Modifications

It has been shown [21] that the above technique generates good quality height fields for smooth objects such as human faces. To deal with shadows, occlusion, or specular highlights which exist commonly on mesostructure surfaces, we make the following three significant modifications.

- **Albedo function.** An albedo function $\{\rho_{ij}\}$ is defined over the surface instead of a constant value. Accordingly, a regularization term for albedo variation is added to the original energy function:
$$\sum_{i,j}(s_{ij}^2 + t_{ij}^2),$$
where s_{ij} and t_{ij} are the symmetric second finite differences of the surface albedo function $\{\rho_{ij}\}$. This albedo regularization term is also weighted by another coefficient γ.

- **Classification of pixels.** Unlike [21] where each pixel from a single image has equal weight for calculating photometric error, we use a small number of registered images $\{I_k\}$ and set a different weight $\eta_k(i,j)$ for each pixel's contribution according to its type: normal pixel, shadow pixel and highlight pixel. Normal pixels are weighted more than shadow and highlight pixels which are treated as outliers. Following a Lambertian reflectance model, we detect outliers using robust statistics [15]. Although [40] proposed an approach to use multiple images, it does not have pixel-wise adaptive weights and regularization terms which are crucial to deal with shadows and specular highlights.

- **Geometry smoothness.** The discontinuity features on a surface (e.g., sharp creases, ridges and grooves) call for varying weights $\mu(i,j)$ for all surface points in the regularization term, such that they are not smoothed too much in the recovered geometry. In our implementation, we first determine the degree of smoothness at all points of the surface by detecting intensity edges of input images. Then points with larger edge responses are assigned smaller weights of regularization. Smaller weights does not necessarily lead to geometric discontinuities. Note that intensity edges may be caused by shadow boundaries or discontinuous reflectance on the surface as well, and might have been accounted for in the albedo function and shadow classification.

All of the above modifications lead to the following form of objective energy function:

$$\begin{aligned} E \quad = \quad & \sum_{i,j}\{\alpha[\sum_{k}(\rho_{ij}R_k(p_{ij},q_{ij}) - I_k(i,j))^2\eta_k(i,j)] \\ & +\lambda(u_{ij}^2 + v_{ij}^2)\mu(i,j) + \gamma(s_{ij}^2 + t_{ij}^2)\} \end{aligned}$$

where

$$R_k(p_{ij},q_{ij}) = \vec{n_{ij}} \cdot \vec{L_k} = \frac{x_{L_k}p_{ij} + y_{L_k}q_{ij} - z_{L_k}}{\sqrt{p_{ij}^2 + q_{ij}^2 + 1}} \qquad (5)$$

Finally, the non-linear minimization problem is solved numerically using the conjugate gradient algorithm [30]. An example of a recovered height field is shown in Fig. 3(c). It is recovered from the four images shown in Fig. 3(a). For comparison, Fig. 3(b) shows the recovered height field from the original algorithm in [21].

5 Generating New Geometry

If the recovered height field is regarded as a gray scale image by converting height values into pixel intensities, we can apply 2D texture synthesis algorithms to generate new surface geometry. The height field image indeed exhibits the stochastic properties which make texture synthesis algorithms work well. Our synthesis algorithm is an accelerated version of the non-parametric sampling method [12], much similar to the multi-resolution algorithm in [38]. It is based on the MRF texture model, which assumes that pixel values in a texture are determined probabilistically by their surrounding patches [12]. An optimized K-D tree based searching algorithm [26] is applied to accelerate the patch matching process. An example of synthesized geometry is shown in Fig. 3(d).

6 BTF Synthesis

Given a natural material and a sparse set of sample images of its BTF, the goal of our synthesis procedure is to generate a complete BTF for a height field that has statistically equivalent mesosturctures as the considered material surface. Our approach works in the same way for both recovered and synthesized height fields. Basically we need to synthesize images for all viewing/lighting settings. This task cannot be done by simply running a general 2D texture synthesis algorithm on each image separately, since consistent underlying geometry for a changing viewing/lighting setting isn't guaranteed. As shown in Fig. 4, all the synthesized images are perceived to have the same statistical features as the original material under corresponding viewing/lighting settings. But they cannot be images of the same BTF because the perceived mesostructure details change from image to image. The reason is that all images are synthesized independently, and no constraints on the underlying mesostructure details are imposed.

BTF images arise not only from spatial variations of surface reflectance, but also from spatial variations of surface geometry, which lead to local shading, highlights, inter- reflection, shadowing and occlusion of local surface elements by neighboring elements. Note that geometry plays an important role in the generation of texture appearance. We take advantage of geometry to render a template image, and then use it as a constraint during texture synthesis.

6.1 Local appearance preserving texture synthesis

One of the most critical things in synthesizing BTF is to generate, under a varying viewing/lighting setting, consistent changes of features caused by the underlying geometry. Therefore, we generate a synthesized gray scale image of the geometry with features, such as shadows, occlusions and highlights, under each given viewing/lighting setting, and use it as a template texture during texture synthesis. We can tolerate minor errors in the recovered or synthesized geometry because the geometry is never used directly for producing the final images, and is only used for rendering the intermediate template images. We would like to make sure that every pixel in the final images is from somewhere in the input sample images to preserve the appearance of non-geometric features as well. Obviously, we should take pixels from the reference image, i.e., the sample image which was taken under the same viewing/lighting setting as the BTF image being synthesized.

With a real reference image and a synthetic template image, we can synthesize a final BTF texture efficiently. We do it block-by-block, rather than pixel-by-pixel. The main idea of our block-wise texture synthesis is: for each pixel of the template image, a block of appropriate size in the reference image is found, which best matches the corresponding neighboring patch in the template image, and then copied to the corresponding region centered at the pixel. This process is repeated until the synthesized image is filled. Since the reference image is taken from a camera and the template image is synthetically rendered, similar features of the material sample may have different intensity and color contrasts in the two images due to different image formation pipelines. Therefore, the reference image is converted into a gray scale image and the histograms of the gray scale template image and reference image are equalized first [17]. Of course, the copied blocks are taken from the original colored reference image.

Our synthesis algorithm consists of the following three main steps:

- Feature ordering
- Feature matching
- Block copying

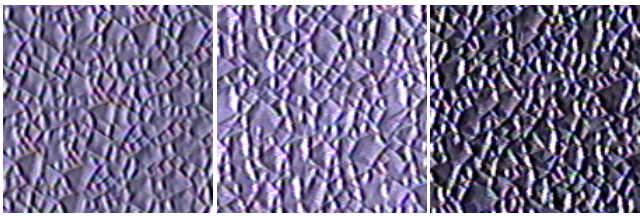


Figure 4: Three independently synthesized textures are generated from three sample images taken at three viewing/lighting settings. A 2D texture synthesis algorithm is used without knowledge of the underlying mesostructure details. However, putting together these synthesized textures does not give us a BTF because they can not be perceived to have the same geometry. Look, for example, at the upright corners of these three images. Clearly they are different geometrically.

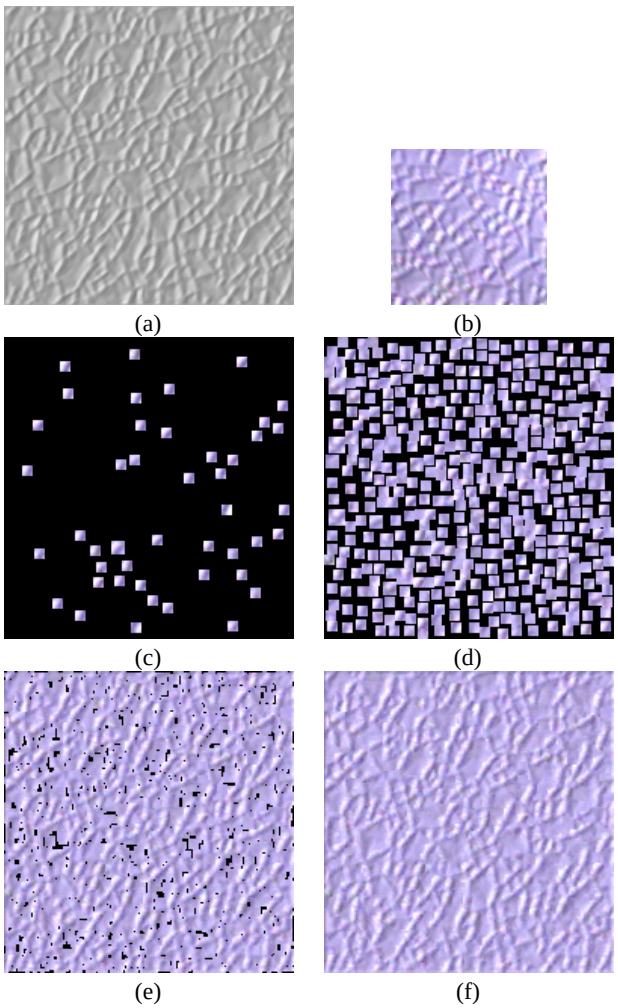


Figure 5: Input images and different stages of feature preserving BTF synthesis. (a) The synthetic gray scale template image; (b) The real reference image; (c) An initial stage during block copying when only pixels with high priority are considered; (d) towards the end of block copying, most pixels are covered by nonoverlapping blocks while gaps among blocks remain as holes; (e) holes are being filled up with tiny blocks that are allowed to overlap with existing blocks; (f) the final result of a synthesized BTF image after hole filling. (f) is synthesized from (a) and (b).

First, we run the Harris feature detector [16] to prioritize all the pixels of the template image according to the gradient variation in the image, such that significant features such as corners will be considered first for synthesis. Then starting from the highest prioritized pixel, an appropriate surrounding block centered at this pixel is built and used to find a block with similar features in the reference image. At last, the found blocks are copied from the reference image to the corresponding positions in the template image.

Optimally the size of the blocks should be set adaptively, which is never an easy task. In practice, our method tests blocks with a few different predefined sizes and picks the best one. The predefined blocks consist of a series of $N \times N$ squares centered at the pixels. There are still two issues that we need to address. The first one is how to measure the similarity of two blocks, one of which is from the template image, and the other is from the reference image. A criterion can be established on any texture model. For simplicity and efficiency, we use the summation of squared differences (SSD), as in some 2D texture synthesis algorithms. The second one is how to compare the matching quality among blocks with different sizes. Since it is unfair to directly compare SSDs between blocks of different sizes, we normalize the SSDs by the number of pixels in each block.

For efficiency consideration, we do not allow blocks to be overlapped at first, so as to prevent pixels from being copied repeatedly. However this leads to some unfilled pixels after block copying. Therefore we need to do hole filling at the end. In fact hole filling is very similar to block copying except that only a smaller size for the blocks is used and the small hole-filling blocks are allowed to overlap with other copied blocks. Fig. 5 illustrates the block copying and hole filling processes. The block matching problem is equivalent to finding the nearest neighbor in a high dimensional space. This has been extensively studied, and many acceleration techniques have been put forward. In this paper, we take a K-D tree based searching algorithm [26] to accelerate our matching process.

The key to our algorithm, or why we can simply copy feature blocks at different locations from the reference image, is that we assume a Markov Random Field model for each BTF image, which enables us to view an image as a realization of the underlying stochastic process which randomly rearranges the collection of local neighborhoods in the image plane, as mentioned in Section 2.1. Note that the height field is not directly involved in the block matching process. But for recovered height fields, we could register all input images with the height field and find the best matching block by running block matching on the height field. This can probably generate more consistent shadowing and occlusion effects in the synthesized images with different viewing/lighting settings. However, registration is hard and interpolation of BTF is not obvious. In practice, we have found that block matching without a height field can generate very good results.

6.2 Reference image generation

In the above synthesis algorithm, a reference image captured from the real world with the same viewing/lighting setting as that of the template image is assumed. However that can hardly be achieved in most situations since dense sampling of the 4D space of viewing/lighting settings is prohibitive, and we can only capture a limited collection of images. The CUReT database mentioned in Section 2.2 is such an example. However, the appearance of BTF images heavily depends on their viewing/lighting settings. In practice, for those viewing/lighting settings not sampled, we can find a sample image with the "nearest" viewing/lighting setting using a distance metric between two viewing/lighting settings.

Let $C_i =< V_i, L_i >=< (\theta_{V_i}, \phi_{V_i}), (\theta_{L_i}, \phi_{L_i}) >, i = 1, 2$, be two viewing/lighting settings. The distance metric is defined to be

$$dist(C_1, C_2) = \sqrt{\|V_1 - V_2\|^2 + \lambda \|L_1 - L_2\|^2} \qquad (6)$$

where λ is the relative weight. A large λ value places more emphasis on the lighting condition. We measure the distance using the polar and azimuth angle of the viewing and lighting directions.

For materials with the isotropy we defined in Section 3, the azimuth angle is not important, but the difference between the viewing and the lighting azimuth angles is. Therefore, a more complicated distance metric is adopted for isotropic materials:

$$dist_{iso}(C_1, C_2) = \min_r \{dist(C_1, C_2(r)), dist(\widehat{C_1}, C_2(r))\} \quad (7)$$

where $C(r)$ is a rotation of C by the angle r around the normal of the surface, and $\widehat{C}$ is the reflected version of C about the lighting direction of C. This definition of distance between two viewing/lighting settings for isotropic materials enables us to make use of the images for isotropic materials in the CUReT database.

In the following discussion, we assume orthographic projection, and that the parallax introduced by the height field on the material surface is minimal. If the viewing/lighting setting of the "nearest" reference image is not the same as that of the synthesized texture, it needs to be morphed. Two important factors must be considered: the foreshortening effect caused by the tilt angle of the viewing direction, and the azimuth of the lighting direction. The first one affects the aspect ratio of mesostructure details, and the second one gives rise to lighting effects such as highlights and shadow patterns.

Our algorithm can be summarized in three steps (Fig. 6).

- First, we back-project the sample image (I_0) onto the surface plane to obtain an intermediate image (I_1), according to the camera parameters used to capture the image.
- Second, we rotate the projected image (I_1) around the surface normal such that the azimuth of its original lighting direction coincides with the azimuth of the desired lighting direction of the synthesized texture. We can perform this transformation because we assume that the mesostructure distribution of the geometry is isotropic. The resulting image is called I_2.
- Third, the final reference image (I_3) is obtained by reprojecting I_2, the rotated version of the back-projected image I_1, onto the target view to maintain correct foreshortening.

6.3 Using multiple reference images

In the above synthesis algorithm, only one closest reference image is used for each template image. Better results can be obtained by using more nearby reference images. And the respective results are weighted and averaged to obtain a final synthesized image. We use distance-based weights. Let C_T be the viewing/lighting setting of the synthesized image, and I_{R_i} be the reference images with viewing/lighting setting C_{R_i} , $i = 1, 2, ..., M$. The weight for each reference image I_{R_i} is set up as:

$$\frac{\exp(-\sigma \cdot dist(C_T, C_{R_i}))}{\sum_{k=1}^{M} \exp(-\sigma \cdot dist(C_T, C_{R_k}))} \qquad (8)$$

where σ is a constant coefficient, such that the weight is close to 1 for the nearest reference image, and almost 0 for the furthest one in the set of chosen nearby images. The scheme enables a smooth transition when the furthest image is removed from the set and a new reference image is added. It is an interpolation scheme for irregularly scattered data, so it works well even when the input image collection does not uniformly sample the viewing and lighting directions. For sample images with regularly distributed lighting and viewing directions, quadrilinear interpolation in the 4D space of viewing/lighting settings would be a more appropriate choice.

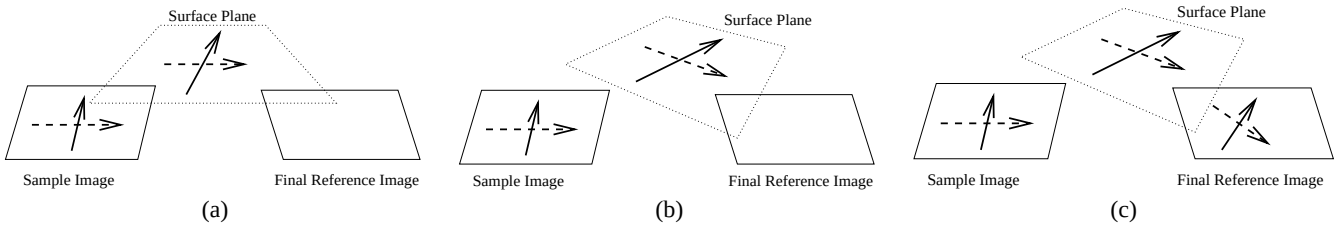


Figure 6: Three steps to generate a reference image from a sample image with variations in lighting and viewing directions: (a) back-project a sample image onto the surface plane; (b) rotate the surface plane to account for change in lighting direction; (c) re-project the rotated surface plane onto the target image plane with different viewing direction.

6.4 Compression

Compression is needed to reduce the amount of disk space for a discrete BTF. However, it is not the focus of this paper since there are many existing mature techniques. For example, synthesized BTFs can be compressed using the clustering technique in [22]. It is also quite straightforward to extend the compression schemes for surface light fields in [28, 39] to work for BTFs.

7 Results

We have successfully tested our algorithms on a few materials from the CUReT database, including rough plastic, plaster, pebbles and terrycloth. For each material, we recovered a 200x200 patch of its height field from four images using our revised height-from-shading algorithm. The four images are chosen to have different viewing/lighting settings and a relatively small number of shadowed pixels. The recovered height fields are then used to synthesize novel height fields at 512x512 resolution, which is much larger than the size of the recovered patches. The BTF synthesis algorithm can run on both recovered and synthesized height fields and generate complete BTFs from all possible viewing and lighting directions. Our program is able to synthesize a 256x256 BTF image in five seconds on a Pentium III 800MHz processor and generate all the images for a 5x12x5x12 grid in the 4D viewing/lighting space in 5 hours. As a result, the materials can be illuminated from all directions and viewed from all directions. Since BTFs belong to lighting-independent surface appearance properties, BTF mapped objects can be easily rendered together with other objects under novel illumination in a ray-tracing or global illumination system.

7.1 Comparison with Ground Truth

Fig. 7 compares three types of materials between the synthesized BTF images and corresponding real reference images. Three pairs of comparisons are shown for each material. Each of the pairs has a distinct viewing/lighting setting. The synthesized images were generated from synthesized height fields. Note that there is a different reference image for each synthesized image. The reference images for the same material are not registered with one another. So they may have different underlying height fields. Nonetheless, the synthesized images can be perceived to have consistent underlying height fields.

7.2 Example images of a synthesized BTF

Fig. 8 shows a plate of synthesized BTF images for rough plastic. The images cover a wide range of lighting directions. The azimuth angle of the lighting direction varies between -90 and 90 degrees. Its tilt angle varies between 45 and 75 degrees. Each column of images have the same viewing direction, but different lighting directions. The left column has a tilt angle of 25 degrees for the viewing direction, and the right column 45 degrees. Each row of images have the same lighting direction, but different viewing directions. The 1st, 3rd, 5th and 7th rows have a tilt angle of 45 degrees for the lighting direction, and the remaining have a tilt angle of 75 degrees. The azimuth angle for the lighting direction increases every other row from -90 to 90 degrees from top to bottom. The appearance of the material varies from image to image because of different locations of highlights and shadows as well as different intensity levels.

7.3 BTF mapping

BTFs can be easily mapped onto objects whose surfaces are parameterized on a rectangular region since texture coordinates for BTF mapping can be set up in the same way as regular 2D texture mapping. Locally shading BTF mapped surfaces from a point light source (instead of a parallel light source) can be carried out as follows. Given the pair of viewing and lighting directions at a certain point on the surface, we can find the corresponding BTF image. From the texture coordinates of that point, we can figure out which pixel value in the found BTF image should be used as the reflectance value for the point. To exploit existing texture mapping functions from graphics libraries such as OpenGL and RenderMan, we need to explicitly extract a 2D texture map for each surface using the above procedure. Note that the extracted texture map is only correct regarding the given viewpoint and light source position. Therefore, we need a distinct texture map for each light source. And the texture maps need to be updated from frame to frame on the fly for an animation with a changing viewpoint or moving objects.

BTF mapping can be implemented as a shader and integrated into any ray-tracing software. We have implemented a shader for BTF mapping in RenderMan BMRT to accumulate the contribution from multiple light source dependent texture maps. A comparison between bump mapping and BTF mapping is shown in Fig. 9. We can see that BTF mapping can deliver more prominent shadowing, occlusion and foreshortening effects as well as spatially varying reflectance. Therefore, the bumps in the BTF mapped image look more protruding and realistic. A scene with multiple objects rendered from RenderMan using ray-tracing is shown in Fig. 10. Some of the objects are BTF mapped. From these examples, we can see that BTF mapping can be considered as a basic rendering function to improve surface appearance.

8 Conclusions and Future Work

In the paper, we presented a novel approach to synthetically generate bidirectional texture functions. Our approach consists of three steps. First, it recovers the approximate 3D geometry of surface details using a shape-from-shading approach. Then, it generates a novel version of the geometric details that has the same statistical properties as the sample surface with a non-parametric sampling method. Finally, it exploits an appearance preserving procedure to synthesize novel images for the recovered or synthesized geometric details under various viewing/lighting settings, which then define a novel BTF. Our experimental results demonstrate that our approach generates BTFs effectively and efficiently.

There are some limitations of our approach in recovering mesostructure details. For example, we impose regularization terms for both geometry and reflectance. Although they have been made spatially-adaptive to account for discontinuities, it is still difficult to recover geometry for natural objects such as grass and straw. And the algorithm for recovering height fields needs a dominant local diffuse component. We would like to address these problems in our future work. Another future direction is to recover both the height field and point-wise non-diffuse reflectance functions simultaneously so that we could generate a BTF with all visual effects without texture synthesis. Note that the global BRDF of a material viewed from distance is most likely more complicated than the local point-wise BRDFs on the material surface to account for interactions among local geometric features. To enforce the global BRDF of a material, we can scale the average intensities of the synthesized BTF images according to the global BRDF. Obviously, mapping BTFs onto arbitrary free-form objects [29] would be a desirable operation.

Acknowledgments

The authors wish to thank Xin Tong for helpful discussion on BTF synthesis, Johnny Chang for implementing the RenderMan BMRT shader for BTF mapping, Yingqing Xu for discussion on patch-based texture synthesis, and the anonymous reviewers for their valuable comments. Timely proofreading by Steve Lin is greatly appreciated. This research was supported in part by the Faculty Startup Funds from University of Illinois at Urbana-Champaign.

References

[1] E.H. Adelson and J.R. Bergen. The plenoptic function and the elements of early vision. In *Computational Models of Visual Processing*, pages 3–20. MIT Press, Cambridge, Mass., 1991.

[2] J.F. Blinn. Models of light reflection for computer synthesized pictures. In *Computer Graphics, SIGGRAPH'77, Vol.11*, pages 192–198, 1977.

[3] J.F. Blinn. Simulation of wrinkled surfaces. In *SIGGRAPH'78*, pages 286–292, 1978.

[4] J. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proc. of Siggraph*, pages 361–368, 1997.

[5] M.J. Brooks and B.K.P. Horn. Shape and source from shading. In *Proc. Intern. Joint Conf. Art. Int.*, pages 932–936, 1988.

[6] E. Catmull. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. PhD thesis, Univ. of Utah, 1974. Report UTEC-CSc-74-133.

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

[8] K.J. Dana and S.K. Nayar. Histogram model for 3d textures. In *Proc. of IEEE Conf. on Comp. Vision and Patt. Recog.*, 1998.

[9] K.J. Dana and S.K. Nayar. Correlation model for 3d textures. In *International Conference Computer Vision*, 1999.

[10] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar. Acquiring the reflectance field of a human face. In *Proceedings of SIGGRAPH*, pages 145–156, 2000.

[11] P.E. Debevec, C.J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *SIGGRAPH '96*, pages 11–20, 1996.

[12] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *International Conference Computer Vision*, 1999.

[13] O. Faugeras. *Three-Dimensional Computer Vision*. The MIT Press, Cambridge, Massachusetts, 1993.

[14] S.J. Gortler, R. Grzeszczuk, R. Szeliski, and M.F. Cohen. The lumigraph. In *Computer Graphics Proceedings, Annual Conference Series*, pages 43–54, 1996.

[15] F.R. Hampel, P.J. Rousseeuw, E.M. Ronchetti, and W.A. Stahel. *Robust Statistics*. John Wiley & Sons, New York, 1986.

[16] C. Harris and M.J. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 147–152, 1988.

[17] D.J. Heeger and J.R. Bergen. Pyramid-based texture analysis/synthesis. In *Proc. of SIGGRAPH*, pages 229–238, 1995.

[18] W. Heidrich, K. Daubert, J. Kautz, and H.-P. Seidel. Illuminating micro geometry based on precomputed visibility. In *SIGGRAPH'2000*, pages 455–464, 2000.

[19] B.K.P. Horn and M.J. Brooks. The variational approach to shape from shading. *Computer Vision, Graphics & Image Processing*, 33:174–208, 1986.

[20] J.J. Koenderink and A.J. van Doorn. Illuminance texture due to surface mesostructure. *J. Opt. Soc. Am.A*, 13(3):452–463, 1996.

[21] Y.G. Leclerc and A.F. Bobick. The direct computation of height from shading. In *Proc. of IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 552–558, 1991.

[22] T. Leung and J. Malik. Recognizing surfaces using three dimensional textons. In *International Conference Computer Vision*, 1999.

[23] M. Levoy and P. Hanrahan. Light field rendering. In *Computer Graphics Proceedings, Annual Conference Series*, pages 31–42, 1996.

[24] S. R. Marschner, S. H. Westin, E. P. F. Lafortune, K. E. Torrance, and D. P. Greenberg. Image-based brdf measurement including human skin. In *10th Eurographics Workshop on Rendering*, pages 139–152, 1999.

[25] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. In *Computer Graphics Proceedings, Annual Conference Series*, pages 39–46, 1995.

[26] D.M. Mount. *ANN Programming Manual*. Dept. Comput. Sci., Univ. of Maryland, College Park, Maryland, 1998.

[27] S.K. Nayar, K. Ikeuchi, and T. Kanade. Determining shape and reflectance of hybrid surfaces by photometric sampling. *IEEE Trans. Robotics and Automation*, 6(4):418–431, 1990.

[28] K. Nishino, Y. Sato, and K. Ikeuchi. Eigen-texture method: appearance compression based on 3d model. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'99)*, pages 618–624, 1999.

[29] E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. In *Siggraph'00*, pages 465–470, 2000.

[30] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C*. Cambridge Univ. Press, New York, 1988.

[31] H. Rushmeier, G. Taubin, and A. Gue'ziec. Applying shape from lighting variation to bump map capture. In *Proceedings of the Eighth Eurographics Workshop on Rendering*, pages 35–44, 1997.

[32] Y. Sato, M.D. Wheeler, and K. Ikeuchi. Object shape and reflectance modeling from observation. In *Computer Graphics Proceedings, Annual Conference Series*, pages 379–388, 1997.

[33] S.M. Seitz and C.R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proc. of IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 1067–1073, 1997.

[34] J. Shade, S. Gortler, L.-W. He, and R. Szeliski. Layered depth images. In *Proc. of SIGGRAPH*, pages 231–242, 1998.

[35] H.-Y. Shum and L.-W. He. Rendering with concentric mosaics. In *Proc. of SIGGRAPH*, pages 299–306, 1999.

[36] E. Simoncelli and J. Portilla. Texture characterization via joint statistics of wavelet coefficient magnitudes. In *Fifth International Conference on Image Processing, Vol.1*, pages 62–66, 1998.

[37] R. Szeliski and H. Shum. Creating full view panoramic mosaics and environment maps. In *Computer Graphics Proceedings, Annual Conference Series*, pages 251–258, 1997.

[38] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of Siggraph*, pages 479–488, 2000.

[39] D.N. Wood, D.I. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. Salesin, and W. Stuetzle. Surface light fields for 3d photography. In *SIGGRAPH'00*, pages pp.287–296, 2000.

[40] R.J. Woodham. Photometric method for determining surface orientation from multiple images. In B.K.P. Horn and M.J. Brooks, editors, *Shape from Shading*, pages 513–532. MIT Press, 1989.

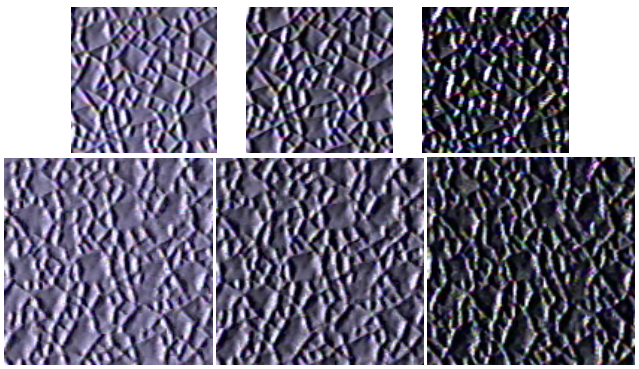
[41] Y. Xu, B. Guo, and H.-Y. Shum. Chaos mosaic: Fast and memory efficient texture synthesis. Technical Report MSR-TR-2000-32, Microsoft Research, 2000.

[42] Y. Yu. *Modeling and Editing Real Scenes with Image-Based Techniques*. PhD thesis, Computer Science Division, UC Berkeley, 2000.

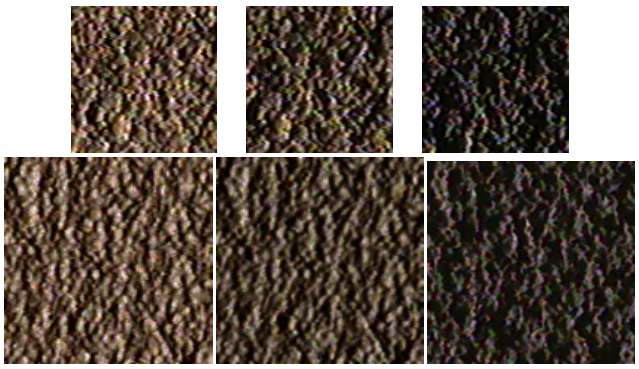
[43] Y. Yu, P. Debevec, J. Malik, and T. Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *Proc. of SIGGRAPH*, pages 215–224, 1999.

[44] Y. Yu and J. Malik. Recovering photometric properties of architectural scenes from photographs. In *Proceedings of SIGGRAPH*, pages 207–217, 1998.

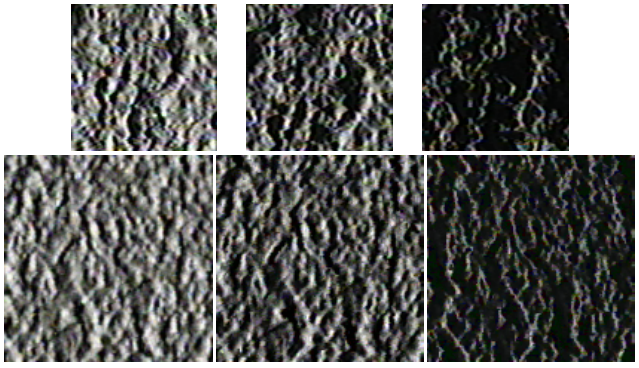
[45] S. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (frame)-towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998.



(a)



(b)



(c)

Figure 7: A comparison on three different materials between synthesized BTF images(large ones) and their real reference images(small ones). Three pairs of comparisons are shown for each material. The reference images are on top of their corresponding synthesized images. (a) Rough plastic, (b) pebbles, (c) plaster. The synthesized images were generated from synthesized height fields. Note that there is a reference image for each synthesized one. The reference images for the same material are not registered to one another. They may have different underlying height fields. Nonetheless, different synthesized images for the same material have consistent underlying height fields.

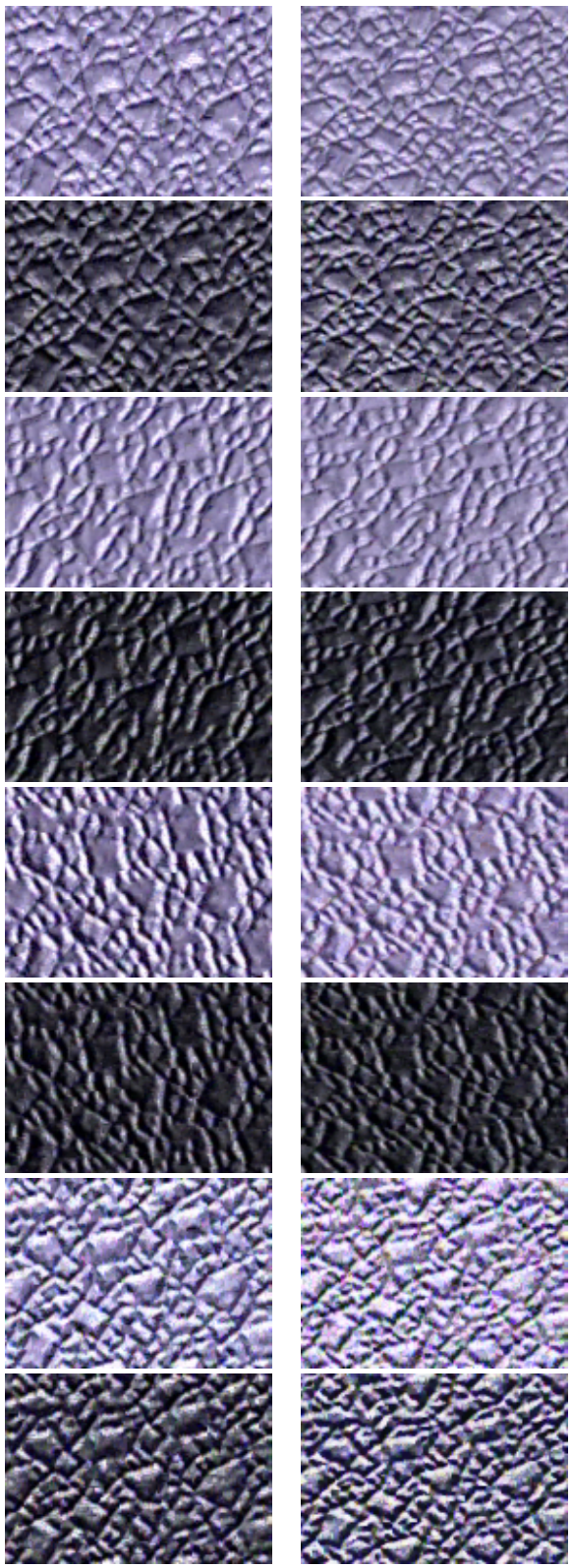


Figure 8: A plate of synthesized BTF images for rough plastic. The images cover a wide range of lighting directions. The azimuth angle varies in a range between -90 and 90 degrees, and the tilt angle varies between 45 and 75 degrees.

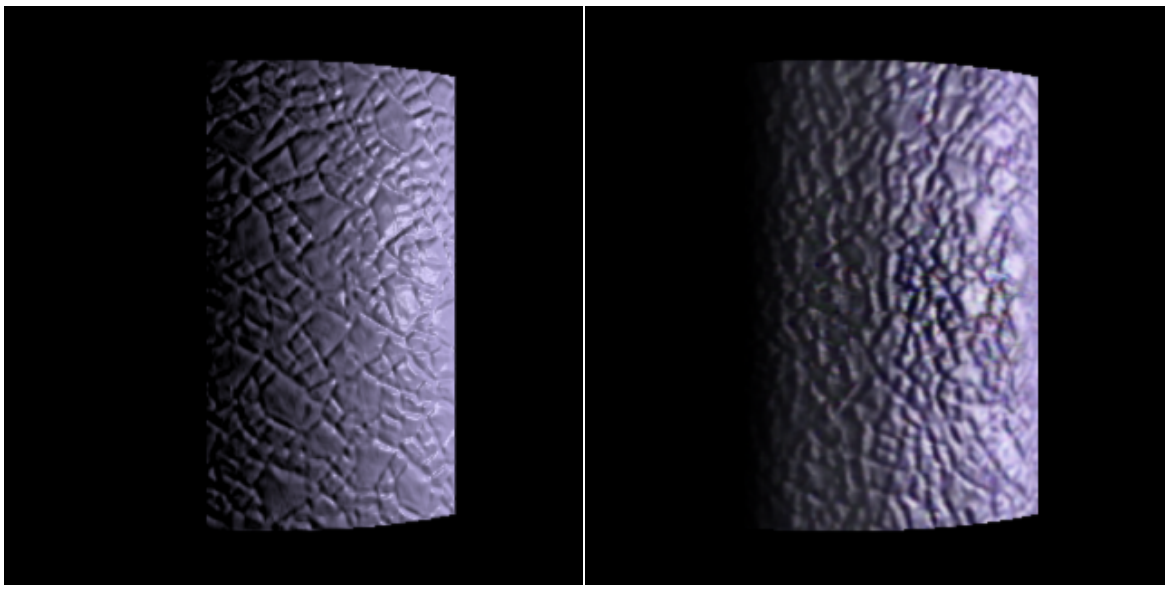


Figure 9: A comparison between bump mapping and BTF mapping. The left image shows a cylindrical surface with bump mapping under the illumination of a point light source from the right hand side. The right image shows the same surface with BTF mapping with the same viewing/lighting setting. We can see BTF mapping has more prominent shadowing, occlusion and foreshortening effects as well as spatially varying reflectance. Therefore, the bumps in the right image look more protruding and realistic.

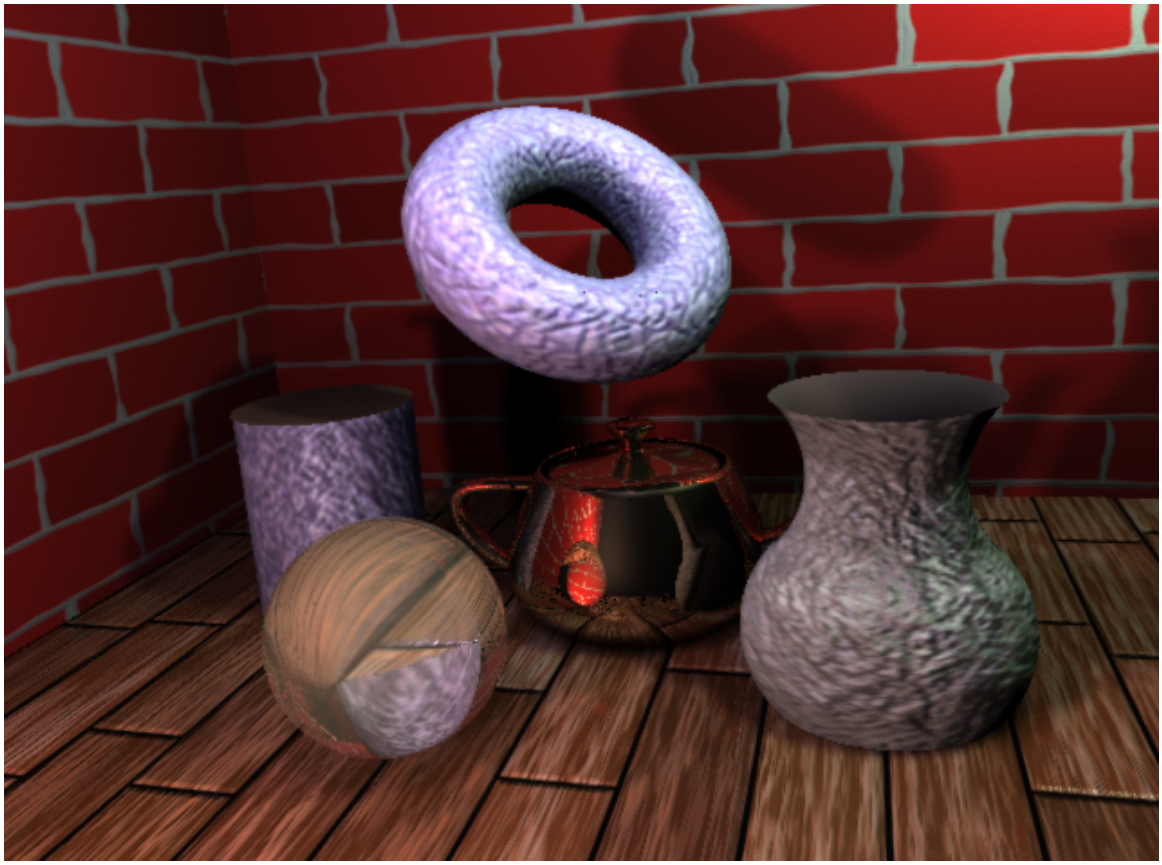


Figure 10: A scene with multiple objects rendered from RenderMan. The torus is mapped with a BTF for rough plastic. The vase is mapped with a BTF for plaster. And the cylinder is mapped with a BTF for terrycloth. We can see small scale shadows among bumps on BTF mapped objects as well as large scale shadows from ray-tracing. The bottom part of the torus has some reddish color coming from interreflection among the objects. The walls are texture mapped. The floor is bump mapped. The teapot has a metallic BRDF and the sphere is half transparent. The BTF mapped objects look more realistic than the texture or bump mapped surfaces.