Coherent Texture Synthesis for Photograph Relighting and Texture Transfer

Yuan-Chung Shen Wan-Chun Ma Yung-Yu Chuang Bing-Yu Chen Ming Ouhyoung

Communication and Multimedia Laboratory

National Taiwan University

Abstract

This paper presents a texture synthesis method for relighting and texture transfer. This method is based on the observation that many points on the surface of a textured object share the same material but have different normals. Hence, once we identify these points, we collect many samples of a material under different illuminations. Hence, we can relight the object by interpolating similar lighting conditions. Furthermore, we can transfer the texture to other objects under different illuminations. We acquire the source textured object by taking a few photographs for a fixed viewpoint under controlled lighting. Stereo is used to estimate the normal map and the unshaded material map of the source object. Then, points of the same material are recognized by neighborhood matching of pixel values of unshaded material map. Finally, we can transfer the texture to another object and render it under different illumination conditions using these clustered texels.

1 Introduction

Rendering objects realistically has been one of the major goals in graphics research and commercial use for years. Computer graphics applications often use textures to represent complex surface appearance without modeling geometric details. However, acquisition or manual painting of textures is still a difficult and tedious task. A number of techniques have been developed to address such a problem. There are algorithms for synthesizing a wide variety of textures over surfaces. However, they are often limited to render the target objects under the same lighting condition of the source texture. To render effects with complex reflectance properties typically requires knowledge of accurate light-transport models, such as *Bidirectional Reflectance Distribution Functions* (BRDFs) or *Bidirectional Texture Functions* (BTFs). Unfortunately, full coverage of all viewing and lighting directions is difficult to acquire or estimate. Our goal is to propose a procedure to acquire textures from real-world objects so that these textures can be transferred to other objects and relit.

The basic idea is based on the observation of Haro and Essa [5]. They use a photograph of a sphere-like object as the source and transfer appearance of this object to an input 3D surface by copying colors from the pixels with similar normals in the source photograph. Their method is very simple but produces realistic results. However, it can only render the target model with the same illumination as the input photograph. If the input illumination changes, the target rendering requires synthesis from another photograph and may cause the inconsistency of the textured appearance. Our work extends their approach and tries to keep the consistency of the texture under different illuminations at hte expense of taking multiple photographs. Thus, Our goal is to relight the source object using several photographs and to transfer the relightable texture of this object to another object using texture synthesis.

Figure 1 is the overview of our method. We capture several photographs of the same object as our input images. During acquisition, we fix the viewpoint and change the lighting directions. Then, we use *Photometric Stereo* to estimate the normal map and the unshaded material map of the source object. With the unshaded material map, we can estimate the distribution of the underlying texture points by our clustering method and generate an index map for the object. Such an index map can generate shading maps from input images for real-time rendering. Besides, we can also use this index map to texture on other objects according to

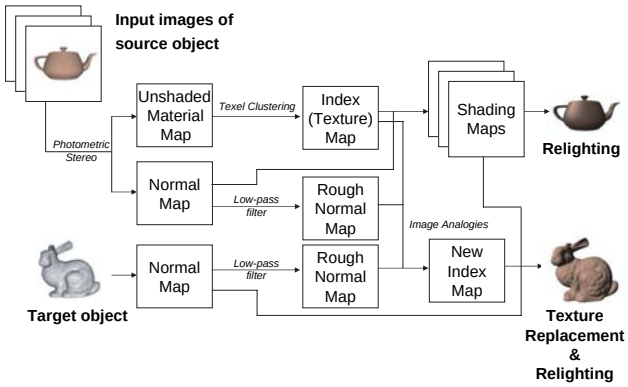


Figure 1 System Overview

the combined consideration of the texture coherence and the geometry surface, using *Image Analogies*.

2 Related work

Our work is built on previous work on reflectance, photometric stereo and texture synthesis, which we summarize below.

BRDFs and BTFs. Kautz *et al.*[8] proposed a technique for the easy acquisition of realistic materials without acquiring the actual BRDFs. They captured several images for different light directions in a fixed orthogonal viewing direction. These acquired data form shading maps, in which the average radiance is increasing linearly for every image. Therefore, real-time rendering can be performed by simple table lookup of this shading map according to the chosen BRDFs. Dana *et al.*[1] proposed a new texture representation called Bidirectional Texture Function (BTF) to model visual appearance of both reflectance and meso-structure. It extends conventional 2D texture with varying lighting and viewing directions, which captures many visual effects such as translucency, inter-reflection, shadow, and occlusion effects of real world material. Unfortunately, BTF acquisition needs more dense samples to recover the reflection properties with different viewing and light directions. Liu *et al.*[11] introduced a novel approach for synthesizing large-scale BTFs.

Photometric stereo. Photometric Stereo was first introduced by Woodham[14]. It however only works for diffuse objects. Hertzman *et al.*propose example-based stereo for arbitrary BRDFs [7]. Goldman *et al.*[4] extend the work of Hertzmann *et al.*[7] and propose a method to extract per-pixel BRDFs along with 3D shape. Their captured photographs are with the same viewpoint and different illumination. They observed that the materials can be depicted as a combination of different fundamental materials. Therefore, they can reconstruct the shape and the spatially-varying BRDFs by optimizing BRDF parameters of an objective function.

Texture synthesis. Texture synthesis has been widely researched for years. Here, we only discuss those most related to ours. Lefebvre *et al.*[10] introduced a novel texture synthesis scheme based on neighborhood matching. For each exemplar, they use coordinates instead of pixel values to synthesize texture into a coarse-to-fine pyramid. We borrow such an idea from their work in synthesis. Wei and Levoy [13] and Turk [12] each proposed an algorithm for synthesizing textures over surfaces around the same time. With a similar idea, used the vector field specified by users to indicate the orientation of the texture for synthesizing. Considering a general type of texture, Ying [15] synthesize textures on surfaces from examples.

Recently, several related photograph editing techniques are proposed to easily manipulate the appearance from images. Haro *et al.*[5] presented a computationally inexpensive method for transferring texture to an object using a single photograph. Their concept is the inverse of the work by Hertzmann [7]. With a single photograph, Fang *et al.*[3] developed a sophisticated tool of shape-from-shading and distorted graphcut textures to conveniently and robustly synthesize textures on objects in photographs. Khan *et al.*[9] proposed a method to change the reflectance properties of the object such as BRDF or translucency in a photograph. Their method is based on the observation that human vision are very tolerant to effects such as shadow, translucency, and transparency.

3 Acquisition

This section describes the acquisition procedure of our system. The procedure and the pre-process are very simple. We capture several (at least three) photographs of the source ob-

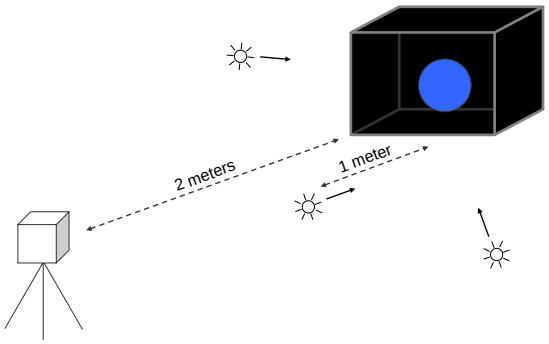


Figure 2 Acquisition setup. The acquired blue ball is inside the black box, and the camera is located 2 meters away. There are three point light source, and the distance between the object and the light is about 1 meter.

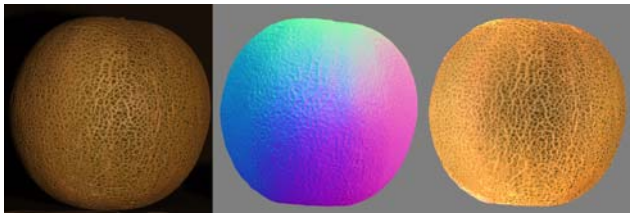


Figure 3 Photometric stereo. The left image is one of the captured "melon" photographs, the middle one is the recovered normal map, and the right one is the unshaded image.

ject at the same viewpoint with different lighting directions. After capturing these photographs, we can use photometric stereo to recover the normal map and the unshaded material map during the pre-process.

The acquisition setup is shown in Figure 2. The acquired object is illuminated by a distant point light source at different incident directions and captured by an orthographic camera. We locate this object inside a black box to avoid the ambient light. A mirror ball is then put beside the acquired object for light calibration. The distance between the object and the camera is about 2 meters, and the distance between the object and the light is about 1 meters. We assume that the source object follows the Lambertian model. Thus, it only requires us to consider the relationship between the normal and the lighting direction. During the acquisition, we capture one image per light direction, and there are totally 9 photographs for robustness.

For light calibration, we locate the brightest pixel on the mirror ball and trace the lighting direction from that pixel. We assume that the mirror ball is a perfectly reflected sphere. Hence, the normal at a point on the surfaces is simply in the same direction as the vector from the center of the sphere. We can thereby infer the lighting direction.

Because we assume that the acquired object follows the Lambertian model, the image intensity is only affected by the angle between the normal of the pixel and the lighting direction. Thus, for normal recovery and unshading, we simply follow the photometric stereo approach [14]. The results for a melon are shown in Figure 3.

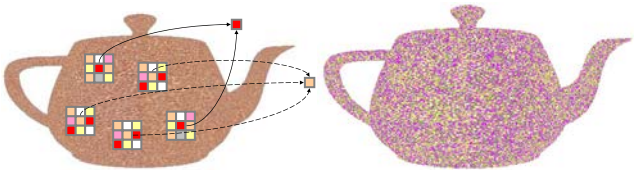
4 Photograph relighting

After acquisition, we have several photographs of the source textured object at a fixed viewpoint under different lighting directions. We first cluster all pixels according to their neighborhood intensities. Therefore, each cluster form a shading map using the samples from the photographs. This shading map describes how the corresponding material behaves under different lighting conditions. We can then illuminate the object under other lighting conditions or apply any isotropic BRDFs on it by simple table lookups of these shading maps.

4.1 Texel clustering

In this step, we try to group pixels which likely correspond to the same texel of the texture map. For real objects, there is no such a texture map. However, due to the nature of textures, the above observation still holds conceptually.

Our approach is based on clustering. The input to our texel clustering method is the unshaded material map of the source object, albedos for Lambertian objects. We use unshaded material map to eliminate the shading effect on the pixel color. We assume that there should be certain patterns laid on its surface. If we can reconstruct the underlying texture by analyzing its texels, we can retrieve such a texture for further rendering or texture transfer. In our observation, we found that different texels have different albedos and have coherence with their neighbors on its appearance. Here, we perform two steps for clustering: (1) cluster pixels by their albedos and (2) further cluster all pixels in each



(a) (b)

Figure 4 (a) Texel clustering. (b) The visualization of indices of the teapot. There are totally 256 clusters.

group by neighborhood matching. Figure 4(a) illustrates the basic idea.

We first classify pixels by albedos. For acceleration, we use *k-means* algorithm to minimize a total intra-cluster variance:

$$V = \sum_{i=1}^{k} \sum_{j \in C_i} \mid x_j - u_i \mid^2,$$

where there are k groups C_i, $i = 1, 2, \ldots, k$, u_i is the centroid point of the i-th cluster, and x_j is the albedo of the pixel $j \in C_i$. As a result, we will have k groups of pixels, and pixels in each group will have approximately the same albedo.

However, classifying pixels only by albedos is not enough to form texels. We think pixels from the same texel not only have the same albedo but also have a coherent texture pattern within its neighborhoods. Hence, after classifying pixels into several groups, we will further cluster them in each group by neighborhood matching, which will make texels more consistent. As the same as in the first step, for each group g, we also use k-means algorithm to cluster pixels, but here the matching vector is not only the albedo of the pixel but albedos within the neighborhoods of the pixel, that is,

$$V'_g = \sum_{i=1}^{k'} \sum_{j \in C_i} \mid \vec{x_j} - \vec{u_i} \mid^2,$$

where there are k' different texels in group g, and $\vec{x_j}$ is a vector of albedos of pixels within the neighborhood of j [1]. As a result, we further divide each group into sub-groups. Each group is then given a unique label, i.e. index, which

[1] In our experiments, we set $k = 8 \sim 64$ and $k' = 8 \sim 32$ according the variance of the underlying texture.

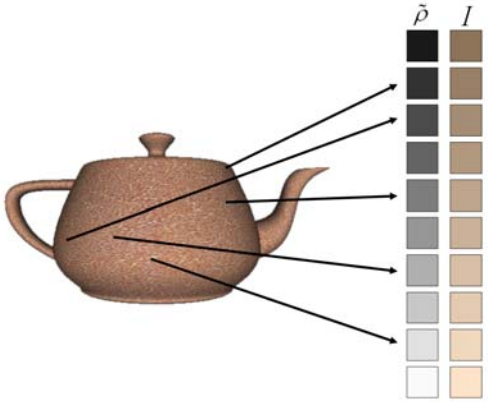


Figure 5 Shading maps. For each texel, we will construct a list of the pixel values from dark to bright.

means an individual texel. Figure 4(b) shows an example of index map.

4.2 Shading maps

Next, we generate *shading maps* $S(k, r)$, where k is the index of a certain texel and r corresponds to the rendering coefficient, e.g. $N \cdot L$. Shading maps behave like a stack of images of the underlying texture under different lighting conditions. Once we want to render an object with such a texture by a specific rendering coefficient r-value, rendering can be easily achieved by a simple table lookup into these shading maps. The goal of shading maps construction is to create a list of the pixel variance due to the light reflection, i.e. r, for each texel. Such a list records the pixel values of each texel from darkest to brightest.

For each group, we have a few pixels of a texel with different normals and lighting directions. We then construct a sorted list from this collection by $N \cdot L$. Re-sampling from this list uniformly can generate a series variation of such a texel from dark to bright. It is done by querying different but uniformly distributed r_i with linear interpolation, extrapolation, or even scaling from this sorted list. Figure 5 shows an illustration for construction of such a list for a texel.

4.3 Relighting

After estimating the surface normals for each pixel and constructing shading maps, we are able to render the object with

varied lighting directions and with different reflection models. For single light source,

$$I(p) = \tilde{\rho}'(N(p), \omega_i)L(\omega_i),$$

where p is one of the pixels, $\tilde{\rho}'$ is the reflection model, ω_i is the incident lighting direction, and $N(p)$ is the normal of p. Although the source object is assumed to follow the Lambertian model and the shading maps construction is also based on the same model, the reflection function $\tilde{\rho}'$ here can be any isotropic BRDFs, which can be seen as a mapping from $N \cdot L$ to BRDFs through shading maps. Lambertian model concerns about the ratio of the reflected light on the diffuse surfaces, and this ratio only depends on the angle between N and L. Other BRDF models have different distribution of these ratios, and the distribution is associated not only with the angle between N and L but also with more parameters. We simply map these reflection models by the values of the ratios. Although such a mapping may not be physically correct, yet the difference is not very sensitive to human vision system due to the tolerance of our eyes. This of course can only approximate isotropic BRDFs.

At first, we will compute rendering coefficient r for each pixel p. The rendering coefficient r means the ratio of the reflected light. Again, although we assume diffuse objects during shading maps construction, we can use any other BRDFs for r instead, such as *Phong model*.As stated above, after deciding rendering coefficient r, we do a table lookup into our shading maps $S(index(p), r)$, and copy the pixel values to the target object. It also performs linear interpolation of different r for smoothing under different illumination. If the desired value r is beyond the maximum or under the minimum in the shading maps, we can either clamp r_k to the maximum or minimum or scale the pixel value according to the desired r.

5 Texture transfer

In this section, we will discuss how to transfer the texture from the source object to the target object, using the texture (index) map generated by our texel clustering method. Different from other texture synthesis methods, we synthesize the index map for the target object instead of the pixel values directly. In this way, we can change the illumination after synthesis as the same as in the previous section.

After clustering pixels into each texel, we can almost retrieve the underlying texture of the source object. For texture transfer, we use the index map of the source object to generate that of the target object by traditional 2D texture synthesis algorithms. In order to match the surface geometry with the texture, here we use normals as well as the texture coherence (albedos) for synthesis since we assume that the source object provides enough coverage of the complete samples of the normals. Besides, for fluctuant surfaces, we will blur the normal map in pre-process, using common low-pass filters, such as box filter, to eliminate the influence of the high-frequency details and maintain the coherent structure because high-frequency will produce a little defects of uneven results.

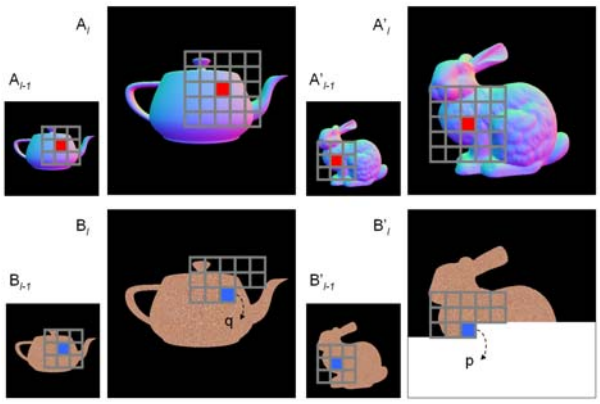


Figure 6 Image analogies. A and A' are the normal maps of the source and target objects, while B and B' are the unshaded images of the source and target object.

In order to preserve both small and large scale for fine-detailed surfaces, we use neighborhood matching in multi-resolution pyramids. As the same as Haro *et al.*approach [5], we use image analogies [6] as our matching algorithm shown in Figure 6. Our texture transfer algorithm takes a set of four images as input: I, A, A', and B, and it will produce B' as well as I' as output:

- A : the normal map of the source object
- A' : the normal map of the target object
- B : the unshaded material map of the source object
- B' : the unshaded material map of the target object
- I : the texture index map of the source object
- I' : the texture index map of the target object

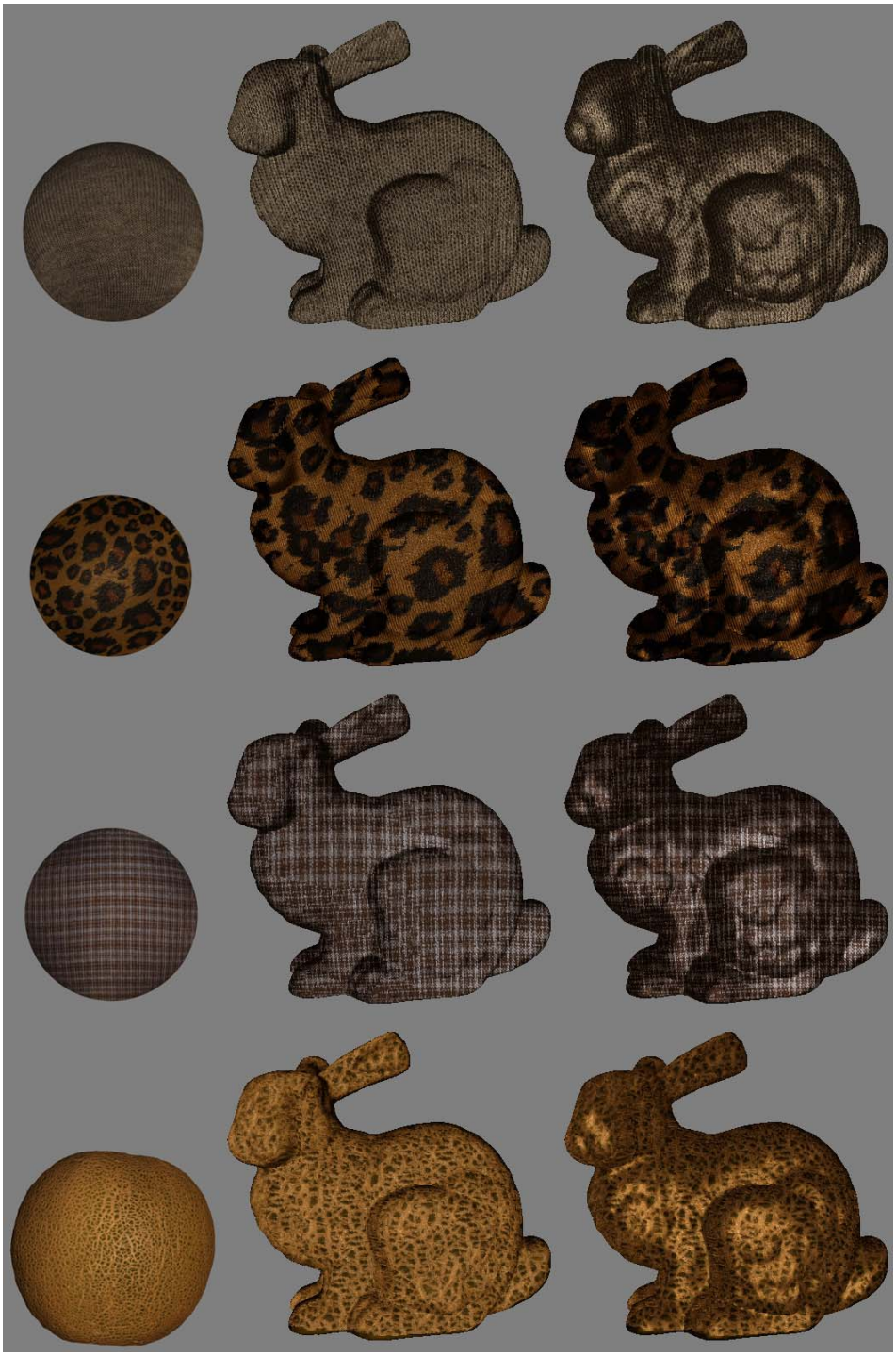


Figure 7 Texture transfer and relighting for real objects. The images in the left column are the source objects. The images in the middle and right columns are generated by our method and rendered in Lambertian Model and in Phong Model respectively.

At first, in the initialization step, three image pyramids (multi-scale representation) are constructed for A, A', and B. In our experiments, we construct 3 levels of the image pyramides. The synthesis algorithm then proceeds from the coarsest level to the finest level, one level at a time, computing each level of B'. At each level l, the feature vectors $F(B_q)$ is constructed for each pixel q on the source object before synthesis. Each pixel p to e synthesized on the target object B' will then be compared against for each q by these feature vectors in a scan-line order. At the last step in the synthesis procedure, we copy the corresponding indices I of the source to the target as well as the albedos itself, and it will produce a new index map I' of the target object with the original texture for more rendering effects.

6 Results

We have implemented our algorithm on an Intel P4 3.2 GHz personal computer. Our method takes about 5 minutes for clustering texels and 3$\sim$5 hours for transferring textures with images of resolution 512x512 pixels. The rendering computation is close to real-time performance (30 FPS) with 10 samples of shading maps under illumination of single point light source.

texture transfer and relighting. Figure 7 shows results of relighting and texture transfer from real objects to synthetic objects. The objects on the left are the source objects. The first three objects are plastic balls wrapped with different kinds of clothes. In the last example, we use a melon as our source object. We choose sphere-like objects because spheres provide better coverage of normals. Figure 8 shows results of transferring to other synthetic objects. In Figure 9, We transfer textures from an real object to another real one and relight it.

Image-based lighting In addition to rendering with the single point light source, we can also apply image-based lighting on our target objects. We use light probe images from Debevec's webpage [2] as light sources. These probes record the lighting conditions of the surrounding environment. For each pixel to be rendered, we sum the contributions of around 1,200 samples rom this probe image. As shown in Figure 10, rendering the target object with complex environment lighting produce more vivid results.

Figure 8 More results.

7 Conclusion and future work

In this paper, we present a texture synthesis algorithm for image-based relighting and texture transfer. The main contribution is to maintain texture consistency during synthesis and make it relightable. Although we make some assumptions about the input photographs, the artifacts are often unnoticeable even if the recovered reflectance properties are not physically correct.

There are still many interesting problems for making this work more practical. We are interested in using this approach to produce more vivid results and apply other effects to real objects. Currently, we can only extract texture for reflectance properties. We would like to explore transferring more shading properties, such as bump maps or displacement maps. As shown in the experimental results, our method works well for stochastic textures. However, for more regular textures, a better synthesis algorithm is necessary to maintain the structured pattern of the texture.

Acknowledgments

This work is partially supported by National Science Council of Taiwan under NSC 94-2213-E-002-096.

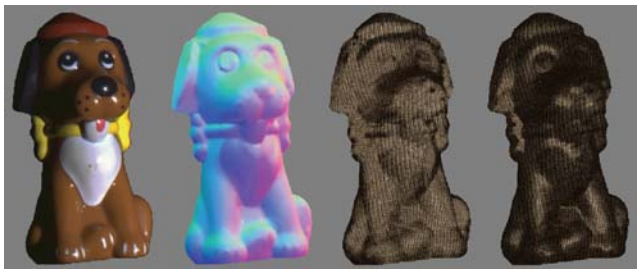


Figure 9 Texture transfer from real object to another real one. The normal map is recovered in pre-process. The right two images are the results rendered in Lambertian model and Phong model respectively.

Figure 10 Rendering with environment lighting.

References

[1] 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.

[2] P. Debevec. Image-based lighting, 2001.

[3] H. Fang and J. C. Hart. Textureshop: Texture synthesis as a photograph editing tool. In *Proc. SIGGRAPH 2004*, 2004.

[4] D. B. Goldman, B. Curless, A. Hertzmann, and S. Seitz. Shape and spatially-varying brdfs from photometric stereo. Technical report, University of Washington, 2004.

[5] A. Haro and I. Essa. Exemplar based surface texture. In *Proc. Vision, Modeling, and Visualization 2003*, 2003.

[6] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proc. SIGGRAPH 2001*, pages 327–340, 2001.

[7] A. Hertzmann and S. M. Seitz. Shape and materials by example: A photometric stereo approach. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 533–540, 2003.

[8] J. Kautz, M. Sattler, R. Sarlette, R. Klein, and H.-P. Seidel. Decoupling brdfs from surface mesostructures. In *Proc. Graphics Interface 2004*, pages 177–184, 2004.

[9] E. A. Khan, E. Reinhard, R. Fleming, and H. Buelthoff. Image-based material editing. In *SIGGRAPH*, 2006.

[10] S. Lefebvre and H. Hoppe. Parallel controllable texture synthesis. *ACM Trans. Graph.*, 24(3):777–786, 2005.

[11] X. Liu, Y. Yu, and H.-Y. Shum. Synthesizing bidirectional texture functions for real-world surfaces. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 97–106. ACM Press, 2001.

[12] G. Turk. Texture synthesis on surfaces. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 347–354. ACM Press, 2001.

[13] L.-Y. Wei and M. Levoy. Texture synthesis over arbitrary manifold surfaces. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 355–360, 2001.

[14] R. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19(1):139–144, January 1980.

[15] L. Ying, A. Hertzmann, H. Biermann, and D. Zorin. Texture and shape synthesis on surfaces. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 301–312, London, UK, 2001. Springer-Verlag.