An Efficient Representation of Complex Materials for Real-Time Rendering

Wan-Chun Ma[1] Sung-Hsiang Chao[1] Bing-Yu Chen[1] Chun-Fa Chang[2]

Ming Ouhyoung[1] Tomoyuki Nishita[3]

[1]National Taiwan University, Taipei, Taiwan
[2]National Tsing Hua University, Hsin-Chu, Taiwan
[3]The University of Tokyo, Tokyo, Japan

ABSTRACT

To make the computer generated imagery (CGI) vivid, analyzing the appearance of materials becomes the first priority. Real world materials usually exhibit complex appearance and cannot be faithfully represented simply by analytical or parametric reflectance functions, such as the combination of bump, glossy, specular and diffuse maps which is widely used in current real-time rendering programs.

In this paper, we propose an appearance representation for general complex materials which can be applied in real-time rendering framework. By combining a single parametric shading function (such as the Phong model) and the proposed spatial-varying residual function (SRF), this representation can recover the appearance of complex materials without much loss of visual fidelity. The difference between the real data and the parametric shading is directly fitted by a specific function for easy reconstruction. It is simple, flexible and easy to be implemented on programmable graphics hardware. We capture real photographs of different materials with a homemade lighting platform and use them to obtain the the data of SRF representation. Experiments show that the mean square error (MSE) between the reconstructed appearance and real photographs is less than 5%.

1. INTRODUCTION

"If it looks like computer graphics, it is not good computer graphics" quoted from Jeremy Birn's book [2], and which touches every computer graphics scientist who is pursuing photorealistic rendering of CGI. Indeed, photorealistic rendering is always an important issue in computer graphics. Applications such as special visual effects in films, training-oriented simulations, or computer games all rely on this technology.

When mentioned about today's real-time rendering applications, computer (video) games probably are good candidates. By surveying the computer games one may acknowledge that, although great improvement have been done in only few years, it is still hard to say that the visual quality is almost photorealistic (although not every computer game needs this characteristic). Fortunately, the realization of programmable graphics hardware and high-level shading language [16] demonstrates the possibility of cinematic real-time rendering. The latest generation of graphics processing units (GPUs), such as NVIDIA GeForce FX or ATI Radeon series, support following similar functionalities:

1. More texture units: up to 16 textures per rendering pass.
2. Longer shader programs with flow control: up to 65,536 instructions per vertex shader, and 1,024 texture and color instructions per pixel shader. ATI even introduces F-buffer technology [17] to virtually increase the program length.
3. Higher floating point precision: the support of the 32-bit floating point format brings true 128-bit color as the same level of precision used in the film industry today.

By taking these benefits, the developers gain much control and flexibility in shading programming. More and more real-time rendering methods, even those that are usually software-based techniques such as ray tracing [23], are proposed.

To make the CGI vivid, analyzing the appearance of materials becomes the first priority. Real world materials usually exhibit complex appearance and cannot be accurately represented simply by ordinary methods, such as basic shading functions or the combination of bump, glossy, specular and diffuse maps. Homogenous materials may demonstrate simple appearance, where plastics and metals are the most typical examples. However, real world materials, even we may regard them homogenous semantically, are complexes and can exhibit complicated appearance. To render this kind of materials in real-time,we need a method that is not just the present well known methods, such as combination of

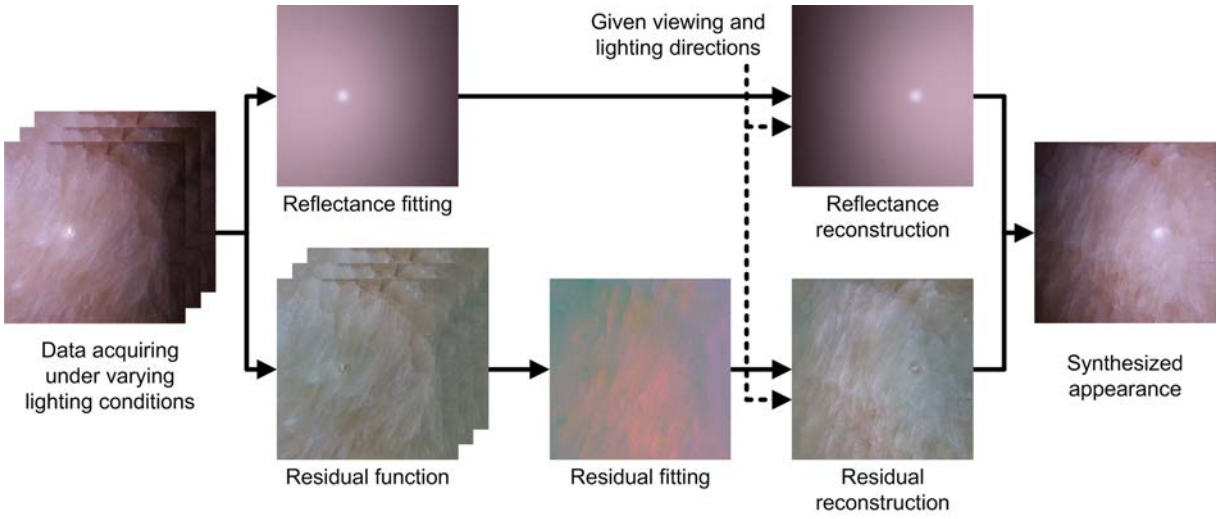


Figure 1: Overview of the framework. First we use a parametric shading function to fit the captured read data, and then the difference (residual) between the real data and the parametric shading is directly fitted by a specific function. Given viewing and lighting directions, the appearance can be synthesized with the shading function and the reconstructed residual. The value of the residual function is shifted so that it can be displayed.

bump, glossy, specular and diffuse maps. Before describing our ideas, we need to define what a complex material would be. The appearance of a complex material should include the following three main properties:

1. Inhomogeneity: local variations caused by contaminants, irregular interior structure, such as mineral crystallization or mix of different materials.
2. Geometry: local variations caused by surface mesostructures, such as self-shadowing, inter-reflection.
3. Transparency: the transparent effect of materials which is captured should also be considered.

Under this definition, materials such as minerals, skins and clothes all belong to complex materials.

The parametric shading function approximately represents the appearance of an object. The rendering of shading function may look alike, but missing many details if we compare it to the photographs of a material. Hence, it is intuitive that the key of what makes people feel the material looks realistic should exist in the difference between the real data and its approximation based on a parametric shading function.

Following the above proposition, in this paper we propose an appearance representation for general complex materials which can be applied in present real-time rendering framework. By combining a single parametric shading function (the Phong model) and the proposed spatial-varying residual function (SRF), this representation can recover the appearance of complex materials without much loss of visual fidelity. The SRF, which is retrieved by subtracting the real data to the parametric shading, is directly fitted by a specific function. By using the restored SRF, we may render the complex materials under different lighting and viewing conditions faithfully. The representation itself is simple, flexible and easy to be implemented on programmable graphics hardware. We capture real photographs of different materials with a homemade lighting platform, and use them to obtain the the data of SRF representation for each materials. Several bi-directional texture functions (BTFs) are also used to generate SRFs. Experiments show that the average error between the reconstruction appearance and real photographs is less than 5%. Figure 1 helps to introduce the overall framework.

2. RELATED WORK

Many successful studies have been done in order to render virtual objects in high quality. By comprehending these algorithms, there are mainly two representations for photo-realistic rendering: image-based and parametric-based.

2.1 Image-based methods

Image-based methods create vivid imagery without explicit knowledge of geometry or reflectance properties. Classic image-based rendering (IBR) [3, 26] uses a large amount of 2D images of different views to generate the illusion of 3D scenes (object movies or panoramas). One may traverse the scenes by directly changing, interpolating, or warping [24] between these images. Most of the early stage object movies are based on fixed lighting, which means it is impossible to change lighting conditions. In contrast to object movies, many attempts have been made, such as [6, 15]. Although they may produce rendering under varying lighting condition, the viewpoint remains fixed. It is possible, though exhausting, to acquire an object movie under various lighting conditions. However, the accompanying tremendous storage need and management problems make it impractical.

In the last decade, more and more representations are proposed for improvement of IBR. The surface light field (SLF)

Figure 2: Our data acquisition platform.

[29, 4] is a function that outputs appearance color to each viewing direction from a specific surface location. The SLF can well represent the object appearance under complex (but fixed) lighting conditions. The polynomial texture map (PTM) [15] is a special case of image-based representation. A PTM approximates the sequence of input images which are captured under varying lighting condition using biquadratic polynomials, so only the fitted polynomial parameters are stored in PTM.

The BTF which is proposed by Dana et al. [5] is a pioneering work in representing complex surface appearance under various lighting conditions and viewpoints in a manner similar to traditional texture map. A BTF is defined as a 6D reflectance function, which has two additional dimensions for the surface position compared to the bi-directional reflectance distribution functions (BRDFs). Due to its high dimensionality, the BTF requires huge memory space for storage. Therefore, how to efficiently manipulate BTFs becomes an issue. Similar to all the image-based approaches, data compression is always a critical issue. Methods such as principle component analysis (PCA) [21], factorization [25] or vector quantization [27, 14] are frequently adopted to preprocess the data for better run-time efficiency.

2.2 Parametric-based methods

In contrast to image-based, parametric-based methods emphasize the use of physically-based parametric reflectance models, which are effective abstraction for describing how the light is reflected from a surface. By statistically fitting the measured data (by either dense or sparse sampling), these models generally provide fair approximations of surface appearance of arbitrary homogeneous materials. From another point of view, fitted models compress the measured data to an extremely small size (only a few of parameters for each color channel). Nowadays, many parametric reflectance models are explored and widely adopted. These parametric BRDFs can be either physically-based, or empirical.

However, a single parametric model cannot handle appearance contributed by complicated surface properties such as transparency or inhomogeneity. Thus, different methods such as bi-directional surface scattering distribution functions (BSSRDFs) [9] or spatially-varying BRDFs (SBRDF) [13, 20, 19] are proposed to solve these problems. Recently, photometric stereo is adopted by parametric-based methods for surface reflectance recovering [8]. These methods can handle SBRDFs and allow for rendering under different viewing and lighting conditions. Impressive synthesis images were shown in their results. Compared to image-based approaches, which needed up to hundreds of images, parametric-based methods require much less (mainly for the fitting process).

We propose a method by combining the advantages of above two approaches: using a reflectance model to represent the approximated appearance of an arbitrary material and a residual function analogous to image-based methods.

3. DATA SOURCES

In this section, we describe two data sources that are used for analyzing the appearance of materials under different viewing and lighting conditions: the first one is acquired by our homemade platform, and the second one is obtained directly from a BTF database.

3.1 Self Acquisition

To analyze the appearance, we need to take photographs under varying lighting conditions. A small-scale homemade platform is built to measure the reflectance of materials under varying lighting conditions. It consists of a rotation arm which orbits the measured target horizontally and a light source mounted on it. Figure 2 shows the data acquisition platform. The light source here is an array of light emitting diode (LED). The color temperature of the LED we use is 6500K. There are several reasons why we choose the LED:

1. The LED is a small light source which concentrates its emission in a narrow range (about only 15 degrees). Its behavior is similar to a spotlight. By arranging several LEDs into a square array, a directional light source may be approximated within a short distance.

2. The LED is inexpensive, and it dissipates only small amount of heat, does not need warm-up, and can be driven by low voltage.

The system requires spherical material samples as measurement targets. The radius of the spheres are 5cm in average. For capturing soft materials (such as clothes), we spread them on the spherical surface. The camera we use is Konica Minolta DiMAGE 7i and the resolution of camera we set is 2560×1920. The camera is initially white-balanced and geometrically calibrated [30] for color and lens distortion correction. Then we station the camera, which is zoomed in high magnification (7x), at a distance away from the sample sphere to simulate orthogonal view (the focal length is set at 1.1m). For each lighting direction, we capture 5 photographs with exposure time ranges from 1/30 to 2 seconds to increase the dynamic range.

The light source rotation arm is manually adjusted to change lighting directions. Several indicative marks are carved on the platform to facilitate this operation. We assume that the direction from the target object to the camera is zero azimuthal angle, then we measure lighting directions from

azimuthal angle -60 to 60 degrees with a increase of 10 degrees. The direction of zero azimuthal angle is avoided because the light source blocks the view of the target object. However, because the rotation arm moves only horizontally, the appearance of materials cannot be sampled under vertical lighting conditions.

The acquired photographs are then transformed into reflectance maps [28], which is frequently used to solve shape from shading problems. A reflectance map $R(p,q)$ is a function of reflectivity in terms of orientations of a parametric surface $S(x,y,z) = (x,y,f(x,y))$ in gradient space:

$$p = \frac{\partial f}{\partial x},\ q = \frac{\partial f}{\partial y}.$$

Since we assume that the captured images are orthogonally-projected, the transformation is simply in the following form:

$$R(p,q) = I(r \times \frac{p}{\|n\|} + c_x,\ r \times \frac{q}{\|n\|} + c_y) \qquad (1)$$

where $n = (p,\ q,\ -1)$ is the surface normal. $I(s,t)$ is the acquired image, r and $(c_x,\ c_y)$ are the radius and center of the projected sphere. The boundary of the reflectance map is manually chosen (here, both the ranges of p and q are between $[-1,1]$). The resolution of reflectance map is set at 256×256.

3.2 BTF Database

In order to get accurate results, we also use BTFs of different materials as the input. The BTF database is provided by University of Bonn[1]. For each material, 6561 images are taken (81 lighting and 81 viewing directions) to sample the appearance variation.

4. DATA REPRESENTATION

BTF is a fundamental representation of complex appearance under varying lighting and viewing directions. We try to decompose the BTF into the following two terms, a parametric reflectance function f_r and a residual function δ:

$$f_{BTF}(P,V,L) = f_r(V,L) + \delta(P,V,L), \qquad (2)$$

where P indicates a position on the surface, V and L are viewing and lighting direction respectively.

4.1 Reflectance Model Fitting

We choose the Phong model with Blinn's specular highlight [2] as the parametric reflectance function f_r in Equation 2:

$$f_{Phong}(V,L) = \kappa_d \times (N \cdot L) + \kappa_s \times cos^n(N \cdot H), \qquad (3)$$

where V is the viewing direction, L is the lighting direction, N is the surface normal, H is the halfway vector defined as $(L+V)/\|L+V\|$, and n is the shininess parameter. The model itself is isotropic and only has three parameters: κ_d, κ_s and n. The view-independent terms, such as the ambient, are combined with the diffuse. Although there exist many BRDF models which are much plausible, the main reason why we adopt the Phong model is that it is simple and can be quickly evaluated. (the Phong model is a major lighting model that is used in real time rendering, especially after the advent of programmable pipelines):

[1]http://btf.cs.uni-bonn.de/

Similar to [18], each pixel in the input reflectance maps is treated as a reflectance measurement with different viewing and lighting directions. A non-linear optimization is used to fit the reflectance model to the data. As many previous studies [13, 12], we take the Levenberg-Marguardt algorithm as the optimization method. For each reflectance map of a specific lighting direction, a set of Phong parameters can be retrieved. Finally, we average all the sets to get a single Phong parameter set. For the appearance data from a BTF, we first calculate its BRDF by averaging the intensity of each BTF images, then the same fitting process is applied to get the Phong parameters.

4.2 Spatial-Varying Residual Function

From the Equation 2, we can get the δ function as

$$\delta(P,V,L) = f_{BTF}(P,V,L) - f_r(V,L). \qquad (4)$$

We name the δ function as the spatial-varying residual function (SRF). In the implementation, the SRF is obtained by subtracting the original reflectance maps (or BTF images) from the reconstructed Phong shading.

4.3 Residual Fitting

Here, we use the δ^* function which is similar to the Blinn's specular component to fit SRF:

$$\delta^*(P,V,L) = s(P) \cdot H. \qquad (5)$$

However, you may apply any kind of δ^* function to fit the residual data. The only purpose of δ^* function is to approximate δ function. Again, H is the halfway vector. To solve the unknown s, for each position P on the reflectance map, the following linear system is formed:

$$\begin{bmatrix} H_{1x} & H_{1y} & H_{1z} \\ H_{2x} & H_{2y} & H_{2z} \\ \vdots & \vdots & \vdots \\ H_{nx} & H_{ny} & H_{nz} \end{bmatrix} \begin{bmatrix} s(P)_x \\ s(P)_y \\ s(P)_z \end{bmatrix} = \begin{bmatrix} \delta(P,V_1,L_1) \\ \delta(P,V_2,L_2) \\ \vdots \\ \delta(P,V_n,L_n) \end{bmatrix}.$$

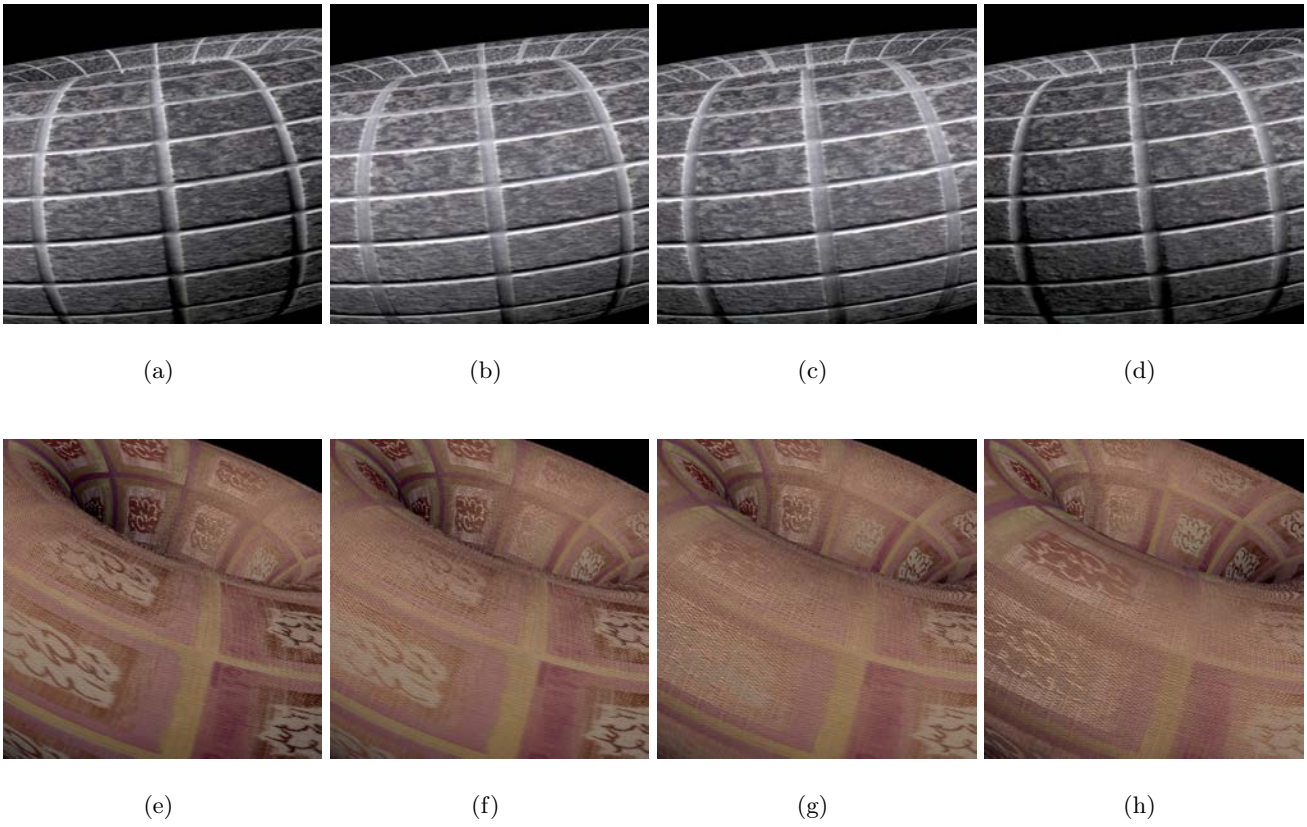
Then we can get the s by a least square operation. For each SRFs in three different color channels the same process is performed. Hence, we may retrieve s_r, s_g and s_b for each P respectively.

The meaning of s is related to normal perturbation in a bump map but differs with following aspects:

1. One may imagine that s is a bump-like map which contains both normal perturbation and color variations. Bump map only stores normal perturbation of the surface meso-structure.

2. However, s is a fitted result from real data and is only used to reconstruct the SRF. It has no definite and strong physical meaning. The SRF may contains information about translucency, self-shadowing and inter-reflection (since all the data are captured from real scenes). If reconstructed well, it could exhibit better appearance than the bump maps could do.

4.4 Postprocessing



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 3: Demonstration of self-shadowing (a)(b)(c)(d) and shading variation (e)(f)(g)(h) effects.

The $s(P)$ vectors are transformed into tangent space first so that we may later use them at the rendering stage[2]. To deal with it, we need the tangent $t(P)$ and binormal $b(P)$ of the object surface. Fortunately, as the target object is a sphere, we can directly evaluate the two vectors by parametric forms. For each position $P = (p,\ q)$, its normal $n(P) = (p,\ q,\ -1)/\left\|(p,\ q,\ -1)\right\|$, then we can get the polar angle $\phi = \arcsin(n(P)_y)$ and azimuthal angle $\theta = \arccos(n(P)_x)/\cos(\phi)$. As a result, the tangent and binormal of a position P are $t(P) = (-\sin(\theta),\ 0,\ -\cos(\theta))$ and $b(P) = (-\sin(\phi) \times \cos(\theta),\ \cos(\phi),\ \sin(\phi) \times \sin(\theta))$ respectively. Finally, each of the $s(P)$ vectors are rotated into tangent space by following equation:

$$\begin{bmatrix} s'(P)_x \\ s'(P)_y \\ s'(P)_z \end{bmatrix} = \begin{bmatrix} t(P)_x & t(P)_y & t(P)_z \\ b(P)_x & b(P)_y & b(P)_z \\ n(P)_x & n(P)_y & n(P)_z \end{bmatrix} \begin{bmatrix} s(P)_x \\ s(P)_y \\ s(P)_z \end{bmatrix}.$$

The three SRF components s'_r, s'_g and s'_b are saved into a single texture file in order for use in the rendering process (if the size of SRF map is $m \times n$, then the size of the texture should be $3m \times n$).

5. RENDERING

The rendering method is similar to ordinary bump mapping techniques [11]. We assume that the input mesh is parameterized, or in other words, each vertex of the mesh has a texture coordinate. Then we can calculate the surface partial derivatives, which are tangent and binormal, for each vertex by using these texture coordinates (the method is described in [7]). In our OpenGL implementation, we use glColor function to pass the tangent vector as color data to the vertex shader. Thus, we do not need to create an additional tangent map and pass it to pixel shader as a texture. However, one may still use an additional texture map to obtain tangent vectors.

The reconstruction of appearance via SRF is simple. For each position P on the surface, we need to transform the lighting and viewing directions into tangent space. Finally, the color of a pixel is calculated as

$$f^*_{BTF}(P, V, L) = f_{Phong}(V, L) + \delta^*(P, V, L). \qquad (6)$$

6. RESULTS

The data manipulation process is implemented in MATLAB and the rendering process is implemented in OpenGL with NVIDIA Cg shading language [16]. The platform of the rendering process is a desktop PC with an Intel Pentium 4 2.4GHz CPU, 512MB memory and a NVIDIA GeForce FX5600 GPU with 128MB video memory. Both of the vertex and pixel shaders are complied in OpenGL NV30 profiles. Real-time rendering results of different materials are shown in Figures 4, 5, and 6. The window size of the rendering system is 600×600 and the refresh rate is averagely more than 20 frames per second (FPS). However, due to

[2]Please notice that for $s(P)$ which is extracted from a BTF, the postprocessing procedure is ignored.

the algorithm is fill-limited, the run-time performance still varies with the pixel number that the projected image takes.

6.1 Visual Effects

The SRF representation effective captures appearance such as self-shadowing and shading variation, which are shown in Figure 3. The major deficiency is that the use of $s(P) \cdot H$ as the δ^* function may brings blurring effect, which is visible at the shading discontinuity, such as the shadow boundaries. Despite of this problem, the SRF representation preserves the low-frequency shading variation well and reveals the feel of 3D texture successfully, especially that it is done by an inexpensive method. Notice in Figures 4 (b) and (f), which demonstrate a sphere made by polished translucent stone. The images reveal the shading variation of the vertical crystal cracks inside the sphere object. These kinds of appearance changes cannot be modeled by traditional bump mapping techniques, which only encodes surface normal perturbation.

6.2 Reconstruction Error Analysis

In order to estimate the reconstruction error, we use a sphere model and render it with the SRF mapped on it. It can be regarded as a reconstruction of the captured spherical objects. For a point (pixel) on the sphere model, we get the gradient space coordinate from its normal and use it to sample the SRF (The normal vector is defined as $(p,\ q,\ -1)$ if p and q are in the gradient space, so if the normal vector $n = (n_x,\ n_y,\ n_z)$ in object space, it is easily to get $p = -n_x/n_z$ and $q = -n_y/n_z$. Then we may use p and q to sample the SRF). A simulation in MATLAB shows that the reconstruction MSEs of the above four materials are less than 5% in average.

6.3 Comparison

This method is an extension of PTM and bump mapping. It is quite different from the eigen-based approaches. It also differs from the SBRDF representation (e.g. diffuse + specular + glossy maps [2]) which does not encode self-occurred shading effects (e.g. occlusion and shadowing). SBRDF representation needs bump mapping techniques to enhance the rendering quality. The proposed technique integrate multiple shading effects, and can be applied to most of the present graphics hardware.

Kautz et al. recently also propose a simple but effective method to render realistic appearance [10]. Unfortunately, their method may result in shading inconsistency. They use the images (shading maps) captured under vertical lighting conditions. So the shadows may vary under these conditions too. However, the same shading map is still used in an arbitrary lighting condition during rendering. The shading of an isotropic material may be correct, but the shadow direction would be incorrect. With our technique, the problem could be solved.

7. CONCLUSIONS AND FUTURE WORK

An inexpensive but effective representation for general complex materials and how to apply it in real-time rendering framework is described in this paper. Comparing to the method that directly fits a function to the real data, the use of the residual function may further reduce the reconstruction errors. The SRF representation is analogous to an algorithm called residually-excited linear prediction (RELP) which is widely used in speech coding. RELP is a variation of the famous linear predictive coding (LPC). LPC fits each input signal to a polynomial, then transmits only the polynomial parameters to the decoder in order to reconstruct the speech. RELP imporves the LPC by quantizing the residual signals and add them into the synthesis waveform to increase the quality of restored speech. The concept of the SRF representation and RELP is close: by means of the residual reconstruction to enhance the synthesized result.

The planned future improvements are included but not limited to the following items:

1. How to synthesize SRF on arbitrary mesh surfaces should be the first priority task in the future.
2. Due to the limitation that our current platform can only sample material appearance under horizontal lighting directions, we need to develop a new one to satisfy all lighting directions. The PTM device [15] draws our attention and is a good reference for the new design.
3. Currently the shape of the material that we measured must be limited to a sphere. By introducing photometric stereo techniques, we hope that in the end we may capture the appearance of objects with arbitrary shapes.
4. Try to make further analysis to find another δ^* function for better reconstruction.
5. Extend the specular reflectance model to an anisotropic one so that we can model the appearance of hairs, furs, CD-ROM disks, etc.

An ambitious goal is to synthesize SRF in a procedural manner. Perlin noise [22] has been widely adopted to create realistic procedural textures, such as woods, marbles, clouds and so on. With the experience of Pixar RenderMan surface shader development, especially the procedural ones, we acknowledge that it is hard to design a shader and adjust its numerous parameters to conform to a specific material appearance. Most of the general purpose surface shader takes dozens of parameters, and shader programmers may found that they are struggling in tuning these parameters. for example, the noise frequency of turbulence or fractional Brownian motion (fBm) functions.

8. ACKNOWLEDGMENTS

We would like to thank the Digimax Studio [1] provides valuable comments on Pixar RenderMan shader development. This work was partially supported by grants from National Science Council, Taiwan, R.O.C. under NSC91-2213-E-002-066 and Silicon Integrated Systems (SiS) Education Foundation.

9. REFERENCES

[1] Digimax studio. http://www.digimax.com.tw.

[2] T. Akenine-Moller and E. Haines. *Real-Time Rendering, 2nd Ed.* A. K. Peters, 2002.

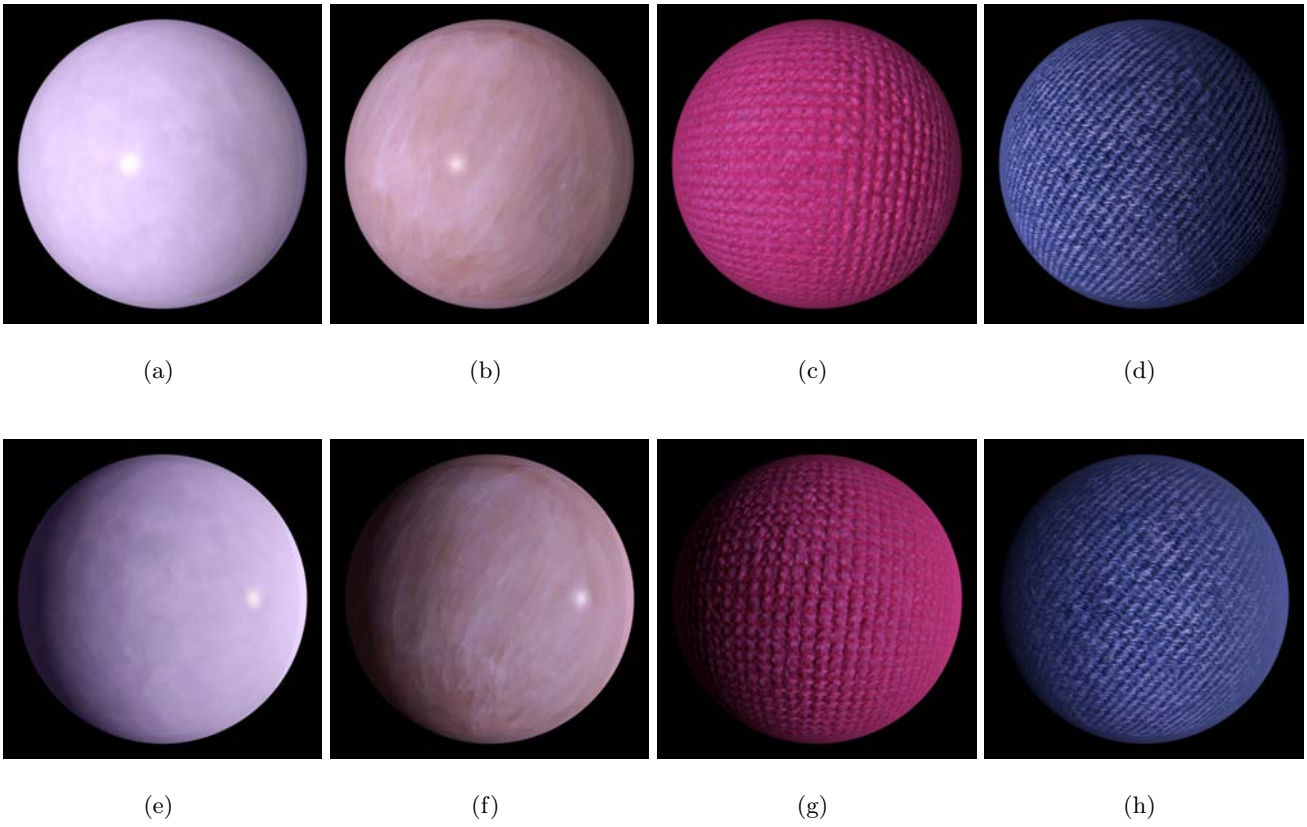


Figure 4: Rendering results of different materials: (a) jade, (b) stone, (c) 3M Scotch-Brite cloth, and (d) jeans, where (e)(f)(g)(h) are the same scenes lit with different lighting direction.

[3] S. E. Chen. Quicktime vr: An image-based approach to virtual environment navigation. In *Proc. SIGGRAPH 1995*, pages 29–38, 1995.

[4] W.-C. Chen, J.-Y. Bouguet, M. H. Chu, and R. Grzeszczuk. Light field mapping: Efficient representation and hardware rendering of surface light fields. *ACM Transactions on Graphics*, 21(3):447–456, 2002. (Proc. SIGGRAPH 2002).

[5] 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.

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

[7] R. Fernando and M. J. Kilgard. *The Cg Tutorial: The Definitive Guide to Programmable Real-Time Graphics*. Addison-Wesley, 2003.

[8] A. Hertzmann and S. M. Seitz. Shape and materials by example: A photometric stereo approach. In *Proc. IEEE CVPR 2003*, volume 1, pages 533–540, 2003.

[9] H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan. A practical model for subsurface light transport. In *Proc. SIGGRAPH 2001*, pages 511–518, 2001.

[10] J. Kautz, M. Sattler, R. Sarlette, R. Klein, and H.-P. Seidel. Decoupling brdfs from surface mesostructures. *Proc. GI 2004*, pages –, 2004.

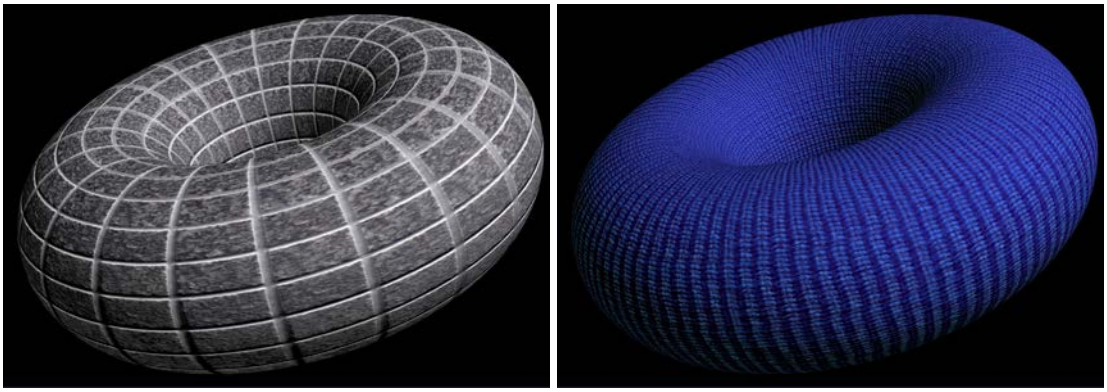
[11] M. J. Kilgard. A practical and robust bump-mapping technique for todays gpus. In *Proc. GDC 2000*, 2000.

[12] E. P. Lafortune, S.-C. Foo, K. E. Torrance, , and D. P. Greenberg. Non-linear approximation of reflectance functions. In *Proc. SIGGRAPH 1997*, pages 117–126, 1997.

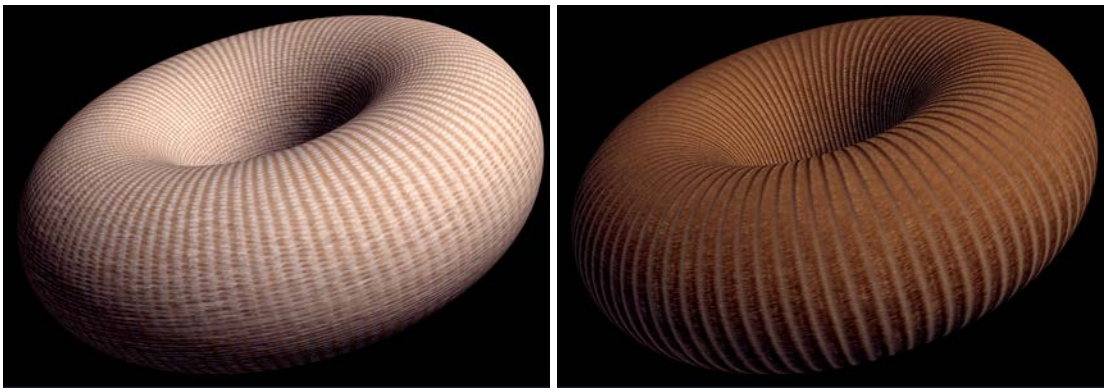
[13] H. P. A. Lensch, J. Kautz, M. Goesele, W. Heidrich, and H.-P. Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM Transactions on Graphics*, 22(2):234–257, 2003.

[14] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using 3d textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.

[15] T. Malzbender, D. Gelb, and H. Wolters. Polynomial texture maps. In *Proc. SIGGRAPH 2001*, pages 519–528, 2001.



(a) (b)



(c) (d)

Figure 5: Rendering results of different materials: (a) granite stone, (b) wool, (c) upholstery, and (d) corduroy. The original data is from BTFs.

[16] W. R. Mark, R. S. Glanville, K. Akeley, and M. J. Kilgard. Cg: A system for programming graphics hardware in a c-like language. *ACM Transactions on Graphics*, 22(3):896–907, 2003. (Proc. SIGGRAPH 2003).

[17] W. R. Mark and K. Proudfoot. The f-buffer: A rasterization-order fifo buffer for multi-pass rendering. In *Proc. Graphics Hardware 2001*, pages 57–64, 2001.

[18] W. Matusik, H. Pfister, M. Brand, and L. McMillan. A data-driven reflectance model. *ACM Transactions on Graphics*, 22(3):759–120, 2003. (Proc. SIGGRAPH 2003).

[19] D. K. McAllister. A generalized surface appearance representation for computer graphics. *Ph.D. Dissertation*, 2002.

[20] D. K. McAllister, A. Lastra, and W. Heidrich. Efficient rendering of spatial bi-directional reflectance distribution functions. *Proc. Graphics Hardware 2002*, 2002.

[21] G. Muller, J. Meseth, and R. Klein. Compression and real-time rendering of measured btfs using local pca. In *Proc. Vision, Modeling and Visualisation 2003*, pages 271–280, 2003.

[22] K. Perlin. Improving noise. *ACM Transactions on Graphics*, 21(3):681–682, 2002. (Proc. SIGGRAPH 2002).

[23] T. J. Purcell, I. Buck, W. R. Mark, and P. Hanrahan. Ray tracing on programmable graphics hardware. *ACM Transactions on Graphics*, 21(3):703–712, 2002. (Proc. SIGGRAPH 2002).

[24] S. M. Seitz and C. R. Dyer. View morphing. In *Proc. SIGGRAPH 1996*, pages 21–30, 1996.

[25] F. Suykens, K. vom Berge, A. Lagae, and P. Dutre. Interactive rendering with bidirectional texture functions. *Computer Graphics Forum*, 22(3):463–472, 2003. (Proc. EG 2003).

[26] R. Szeliski and H.-Y. Shum. Creating full view panoramic image mosaics and environment maps. In *Proc. SIGGRAPH 1997*, pages 251–258, 1997.

(a) (b) (c)

(d) (e) (f)

Figure 6: Rendering results of Venus in different lighting conditions. Because we have not applied texture synthesis on surface to create corresponding SRF map, one may notice that discontinuity may appear across texture patches. Venus consists of 84,668 triangles.