

Yi Gong
Wei Chen
Long Zhang
Yun Zeng
Qunsheng Peng

GPU-based rendering for deformable translucent objects

Published online: 10 January 2008
© Springer-Verlag 2008

Y. Gong · W. Chen (✉) · L. Zhang ·
Y. Zeng · Q. Peng
State Key Lab of CAD&CG, Zhejiang
University 310027,
Hangzhou, China
{ygong,chenwei,lzhang,yzeng,peng}
@cad.zju.edu.cn

Abstract In this paper we introduce an approximate image-space approach for real-time rendering of deformable translucent models by flattening the geometry and lighting information of objects into textures to calculate multi-scattering in texture spaces. We decompose the process into two stages, called the *gathering* and *scattering* corresponding to the computations for incident and exident irradiance respectively. We derive a simplified illumination model for the *gathering* of the incident irradiance, which is amenable for deformable models using two auxiliary textures. In the

scattering stage, we adopt two modes for efficient accomplishment of the view-dependent *scattering*. Our approach is implemented by fully exploiting the capabilities of graphics processing units (GPUs). It achieves visually plausible results and real-time frame rates for deformable models on commodity desktop PCs.

Keywords Sub-surface scattering · BSSRDF · Translucency · Real-time rendering

1 Motivation

Photo-realistic modeling and rendering have proven to be capable of simulating the appearances of almost all types of objects, especially those with ideal diffuse, specular or gloomy materials. Even for the objects with complex materials like skin, jade, marble and milk, there have been well-established techniques in the computer graphics community. Among them, one challenging task is the simulation of translucent materials. Translucent objects are not fully transparent and allow light to transport and diffuse directly through their interiors, making their appearances partially mellow and quite different from other materials.

In the last few decades, the fast development of graphics hardware has left much room for the acceleration of rendering algorithms. The release of shader model 3.0 and shader model 4.0 enable GPU to support more complicated functions in shaders. However, complex materials

like translucent objects are still not taking full advantage of GPU power, especially the deformable ones.

1.1 Previous works

Translucency cannot be simulated by simply using reflection or transparent illumination models because it is formed by single and multiple scattering of the light. As a consequence, to accurately mimic a light transport in a translucent object, time-consuming path sampling and many multi-integral computations are required. Traditional solutions include the finite element methods [1, 19, 20], diffusion approximation [23], path tracing [6, 13], bidirectional path tracing [11] and photon mapping [9].

To make it amenable for real-time processing, sample-based rendering techniques are adopted. Hanrahan and his colleagues [6] used *bidirectional reflectance distribution function (BRDF)* for subsurface scattering, with

which the existing light is assumed to be emitted at the same point as the incident light on the surface of the object. Later, Jensen and his colleagues [10] introduced the dipole source method based on a more accurate *bidirectional subsurface reflectance distribution function (BSS-RDF)* model, which consists of two parts, namely the exact single scattering and the multiple scattering. Although the dipole source method is only suitable for the case where the incident point and the existing point of the light are on a plane, applying it to an arbitrary surface can still generate appealing results with appropriate restrictions on the discretization of the underlying surface [10]. Jensen and his colleagues [8] further proposed an acceleration technique by employing a hierarchical structure. Thereafter, many approaches have been introduced for improving the performance by either pre-computing and reusing the radiance or simplifying the illumination model.

Lenzsch and his colleagues proposed a texture atlas aided algorithm [12] which preprocesses the handled geometry to collect the impulse response of each surface point under subsurface scattering. It separates the response into local and global parts. The local response is modeled as a filter kernel stored in the textures and the global response is calculated and stored in the vertices. It is apparent that the requirement of costly preprocessing limits its application for deformable models. Recently, Hao and his colleagues [7] implemented a spherical harmonics-based real-time rendering approach. Although visually plausible results are achieved, it is likely to be fit for undeformable models. A similar situation happens with the pre-computed radiance transfer (PRT) techniques [21, 26], as they follow the same pre-computation mechanism and demand too many memory consumptions. More recent work by Sloan and his colleagues [22] allows for local deformations, though it is not sufficient for large and deformable models.

By simplifying the illumination computation, interactive rendering of deformable models is made possible at the expense of the loss of image quality. Mertens and his colleagues [16] proposed a clustering-based algorithm that achieves interactive frame rates even for deformable models. They also proposed an importance sampling-based method, to consider only local scattering [15]. Dachsbacher and Stamminger [2] introduced an efficient two-phase image-space technique by using a shadow map to store the irradiance from the light sources for latter scattering computation. Due to the limited accuracy of the sampling and filtering provided by the graphics hardware, the results are not quite satisfactory. In addition, the algorithm cannot simulate all-frequency lighting [26] because the shadow map is only feasible for point lights and directional lights.

There are also some works on the simulation of inhomogeneous materials. Goesele and his colleagues [5] proposed a so-called DISCO algorithm-based on the meas-

ured sub-scattering data. Xin Tong and his colleagues [25] put forward a quasi-homogeneous material modeling and rendering method with a special-purposed sampling device. Peers [18] introduced a compact representation of heterogeneous subsurface scattering functions. Donner and Jensen proposed a multi-layered model for light diffusion in translucent materials [3], and applied it to human skin simulation. They further introduced a spectral BSS-RDF model which considers not only subsurface but also surface scattering [4].

1.2 Our contributions

Based on the investigations on the related literature, we understand that there are still challenges for real-time rendering of deformable translucent objects. Most of existing approaches suffer from either efficiency issues, such as [22, 26] or lack of photo-realistic effects like [2, 12], in terms of deformable models.

This paper describes our efforts on boosting the performance and quality with a new approximate image-space approach. Our method adopts also BSSRDF model, and mimics most kinds of light sources by integrating the contributions from point lights, directional lights and the ambient lights. Conceptually, we decompose the computation of translucency into two stages. In the first stage, the radiances from all types of lights are collected over the surfaces. We call this procedure the *gathering* of irradiance. To facilitate its efficient computation at runtime, we derive a simplified computational model that is especially adoptable for the ambient lights. In addition, we employ an intermediate data sheet to store the gathered incident irradiance.

On the other hand, in our approach the computation of outgoing irradiance under a given viewpoint is a simplified imitation of the light transport inside the objects. We denote it as the *scattering* operation, where the color of each visible surface point depends on a variable number of lightened source points. This makes the implementation with current graphics hardware nontrivial because the rendering pipeline is designed as amenable for *gathering* operations, where the destination of each pixel is defined before rasterization. With an attempt to convert the *scattering* operations into a *gathering* procedure in GPUs, one may implement it at the fragment level. This scheme requires a great many texture lookups at the complexity of the image resolution. We overcome this inefficiency with hierarchical sampling scheme.

In the rest of this paper, we first introduce a simplified translucent illumination model based on a brief overview of the BSSRDF model. We then present the gathering of irradiance in Sect. 3. Section 4 describes how to implement the scattering process in programmable graphics hardware. The experimental results are presented in Sect. 5. Finally, we conclude the whole paper and highlight the future work in Sect. 6.

2 A simplified illumination model

BSSRDF is a more accurate lighting model than BRDF. While BRDF only considers the link between the incoming light and the outgoing light at the same point of the surface, BSSRDF can depict the light transport between two different points, x_i and x_o , on a surface. The relationship between the illumination at one surface point x_i with light distribution at another surface points x_o is expressed as follows [10]:

$$dL_o(x_o, \omega_o) = S(x_i, \omega_i; x_o, \omega_o) d\Phi_i(x_i, \omega_i), \quad (1)$$

where $L_o(x_o, \omega_o)$ is the outgoing radiance at x_o along the direction ω_o . $\Phi_i(x_i, \omega_i)$ is the incident flux at x_i along the direction ω_i , and $S(x_i, \omega_i; x_o, \omega_o)$ denotes the value of BSSRDF. By integrating over all incoming directions and areas, the total outgoing radiance is computed.

Typically, BSSRDF model considers the effects of two terms, single scattering and multiple scattering:

$$S(x_i, \omega_i; x_o, \omega_o) = S_d(x_i, \omega_i; x_o, \omega_o) + S^{(1)}(x_i, \omega_i; x_o, \omega_o), \quad (2)$$

where $S^{(1)}$ and S_d stand for the single and multiple scattering terms, respectively. In fact, a simplification can be made by skipping the single scattering term alone to simulate the effect of highly scattering materials [8]. Moreover, as multiple scattering smoothes the incident light from different directions, S_d can be further approximated by a low-dimensional function, R_d , which depends only on the incident angle and outgoing position [10], yielding a simplified BSSRDF model:

$$S(x_i, \omega_i; x_o, \omega_o) \approx S_d(x_i, \omega_i; x_o, \omega_o) = \frac{1}{\pi} F_t(\eta, \omega_o) R_d(x_i; x_o) F_t(\eta, \omega_i), \quad (3)$$

where $F_t(\eta, \omega_i)$ and $F_t(\eta, \omega_o)$ denote the fresnel transmission terms. R_d is the dipole approximation for multiple scattering, indicating how much proportion of light will be transported from x_i to x_o on a plane. That is, the dipole source method converts the part of incoming light at x_i , which will transport to x_o , to a couple sources above and over x_i (at z_v and z_r) as depicted in Fig. 1.

The single dipole approximation for multiple scattering is formulated as

$$R_d(x_i; x_o) = \frac{\alpha'}{4\pi} \left[z_r (1 + \sigma_{tr} d_r) \frac{e^{(-\sigma_{tr} d_r)}}{d_r^3} + z_v (1 + \sigma_{tr} d_v) \frac{e^{(-\sigma_{tr} d_v)}}{d_v^3} \right]. \quad (4)$$

Here, α' is the reduced albedo, σ_{tr} is the effective transport coefficient, z_r and z_v are the distances from the dipole

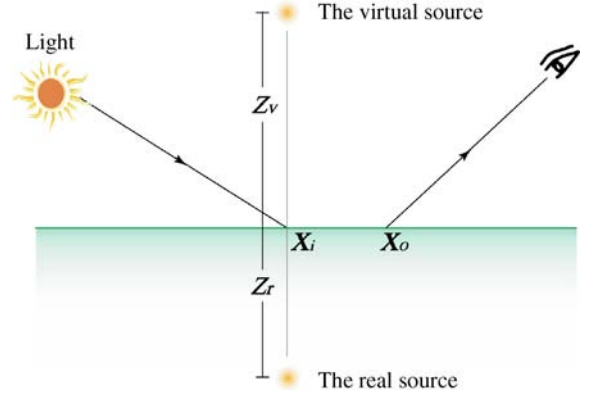


Fig. 1. The dipole approximation of multiple scattering

lights to the surface, d_r is the distance from x to the real source, and d_v is the distance from x to the virtual source.

We make two assumptions here to simplify the translucent simulation process. Firstly, we assume the light transfers uniformly in all directions on its surface. Thus, together with Eqs. 1 and 3, we obtain:

$$L_o(x_o, \omega_o) = \frac{1}{\pi} F_t(\eta, \omega_o) B(x_o) \quad (5)$$

$$B(x_o) = \int_s E(x_i) R_d(x_i; x_o) dx_i \quad (6)$$

$$E(x_i) = \int L(x_i, \omega_i) F_t(\eta, \omega_i) (\mathbf{n}_i \cdot \omega_i) d\omega_i. \quad (7)$$

Secondly, as the light transmission in translucent objects decays exponentially with the distance, the far points on the surface make very little contribution to the current shading pixel and thus can be neglected. Namely, samplings are only necessary in a small neighborhood of current pixel. Moreover, S varies smoothly except in the very local region near x_o . Consequently, we can discretize the surface near x_o into hierarchical pieces – small in its near region, and large in the far region. Thus $E(x_i)$ and $R_d(x_i; x_o)$ can be treated as constant in each piece. Equation 6 can be further reformulated as:

$$B(x_o) \approx \sum_i E(x_i) R_d(x_i; x_o) A(x_i), \quad (8)$$

where $A(x_i)$ is the area of the region surrounding x_i , after the surface discretization.

Finally, suppose that the light varies not very frequently, Eq. 7 can be simplified by discretizing the incident light direction into several bases:

$$E(x_i) = \sum_j E_j(x_i) F_t(\eta, \omega_j) \psi_j(x_i), \quad (9)$$

where $E_j(x_i)$ is the incident irradiance of x_i projected on the basis ψ_j .

3 The gathering of irradiance

With the simplified illumination model, we classify the light sources into two categories, namely the point or directional lights and the ambient lights. We employ an intermediate sheet buffer to keep the diffused incident irradiance from all light sources, called *diffusion source map* (DSM). Ideally, it should collect all possible $E(x_i)$ for each surface point. Although the representation of DSM is flexible, one fundamental requirement is that it should facilitate fast neighbor searching and area estimation for subsequent scattering operations, which will be further mentioned in Sect. 4.2. Accordingly, we propose to generate a equiareal parameterization [14, 24] of the handled surface and keep multiple textures as DSMs. The searching of the neighboring regions can be accomplished by the texture look-up operations in GPUs.

For each element of DSM, we record the surface location, the surface orientation and the sum of the incident irradiance (Fig. 2). These attributes are to be used in the scattering stage. To accumulate the contributions from the point or directional lights and the ambient lights,

we use two auxiliary textures as described below. With hardware-accelerated texture mapping technique, the irradiance from the shadow map and the cube map can be accumulated in the DSM in one rendering pass.

3.1 The point and directional lights

To determine the contributions from each point or directional light to the handled surface, we employ a shadow map for the storage of the irradiance in the visible surface at the viewpoint of the light source. The shadow map [27] is a commonly used technique in real-time rendering. Because we do not consider the decay of the light, this scheme amounts to add a constant value to the irradiance map for each light source. Figure 3 illustrates the usage of the shadow map in the accumulation of the irradiance from the point and directional lights.

3.2 The ambient lights

The second auxiliary texture is a spherical or cubical environment map for storing the irradiance caused by the

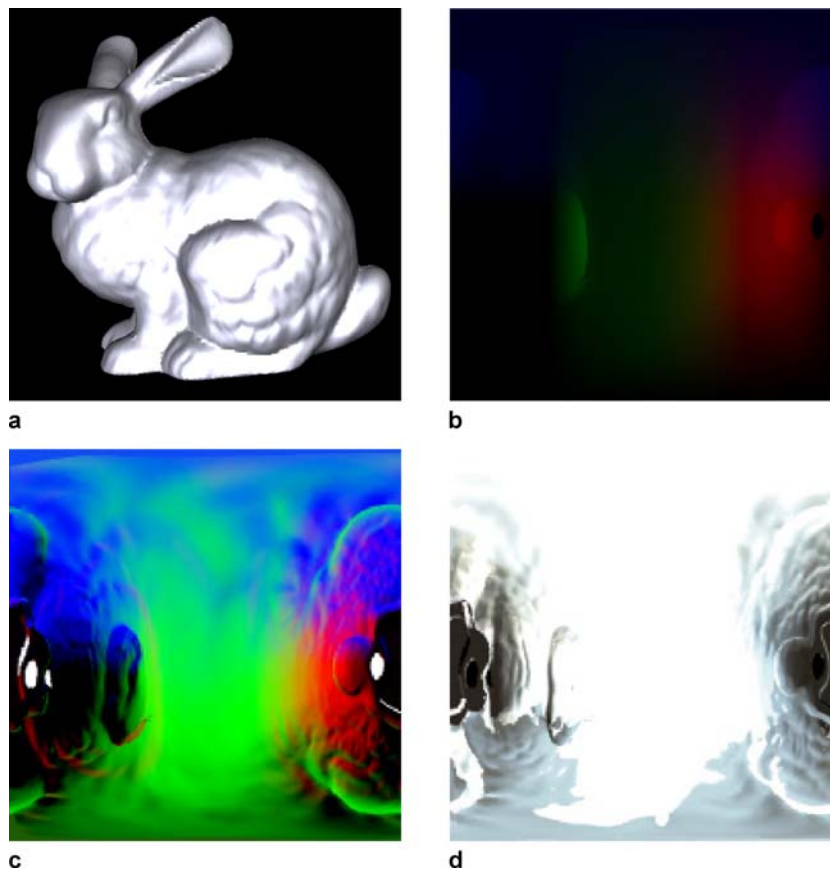


Fig. 2a–d. Illustration of the DSM for the bunny model. **a** The bunny model; **b** The component for the surface location; **c** The component for the surface orientation; **d** The component for the incident irradiance

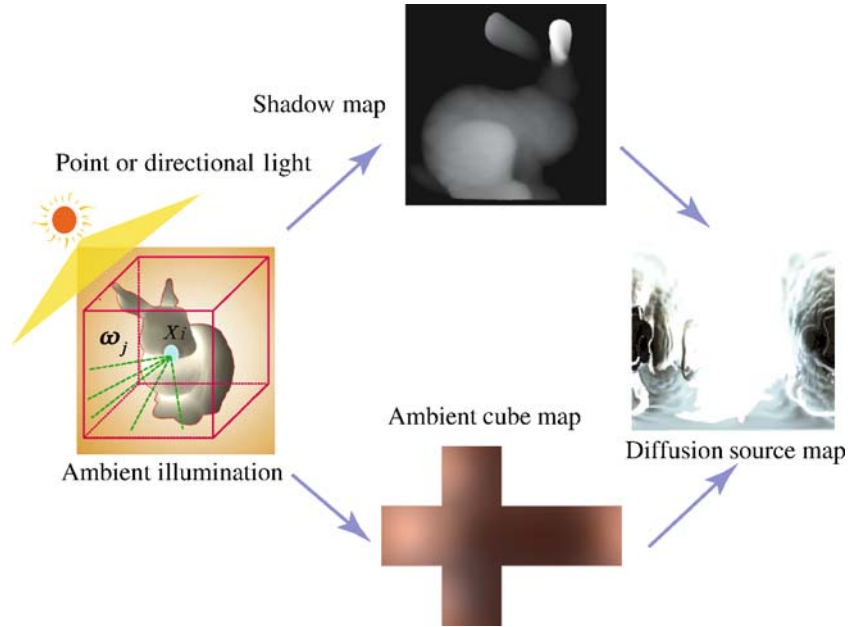


Fig. 3. The DSM is determined by accumulating the contributions from the point or directional lights and the ambient lights at each surface location

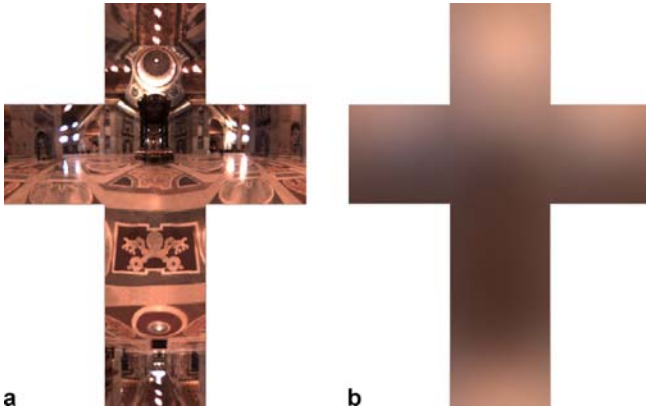


Fig. 4. **a** The original environment cube map; **b** Ambient cube map computed by using spherical harmonic function

ambient lights. We utilize the simplified Eq. 9 to calculate the environmental irradiance at each surface location x_i from multiple lighting directions. The irradiance from ambient lights are determined by using sphere harmonic method adopted in the PRT (pre-computed radiance transfer) algorithm [21]. In the preprocessing stage, we compute the basis and coefficients of the sphere harmonic functions and record them in a cube map. To simulate the influence from ambient light, the cube map is represented as a texture and kept in the local video memory for efficient processing, as depicted in Fig. 4.

4 The scattering of irradiance

After gathering the irradiance from each light source into the DSM, the total outgoing irradiance is to be evaluated at runtime with the derived Eq. 8. This step mainly contains two issues: irradiance sampling and surface area estimation.

4.1 Irradiance sampling

Equation 8 requires excessive texture look-up operations, which are still costly in current programmable graphics hardware. Uniformly sampling a large neighborhood of each vertex in the texture space will definitely slow down the rendering performance. Thus, we have to find an efficient way to reduce the sampling number for each shading point, with the minimum influence to visual effect. Our solution makes use of a hierarchical irradiance accumulation method to sample the DSM for the integral computation. It is based on the observation that light scattering of highly scattering translucent objects attenuates exponentially with the distance. Therefore, we do not need to consider the contributions from very far surface locations. Even for the moderately far surface locations, the irradiance computation can be hierarchically simplified. Inspired by this investigation, for each handled surface location, we sample a set of points (shown with red color) on its near neighborhood uniformly, and eight points (shown with blue color) for moderately far regions. Figure 5 illustrates this hierarchical scheme, where the black point denotes the underlying surface location.

4.2 Estimation of exident irradiance contribution

To compute irradiance contribution of each sampled point, we still need to estimate its surface area. Most translucent rendering methods require that the vertices are distributed evenly, otherwise there must be a pre-process to make them evenly distributed by resampling. In this way, the surface area of all faces will be constant value, which can be extracted from the integral term. In our algorithm, uniform triangle surface area cannot help anything, because Eq. 8 is calculated on the fly in texture space. Our solution is to adopt an equiareal parameterization method [14, 24] to flat our DSM. Namely, each texel is mapped to the original model with the same area value to avoid calculating surface area texel by texel. Afterward, we count on the exident irradiance contribution from each sampled texel and estimate the unsampled area by interpolation. The final interpolation can be easily implemented by applying a weight mask on the sampled value. Finally we get an approximation of the irradiance response curve as Fig. 6 shows.

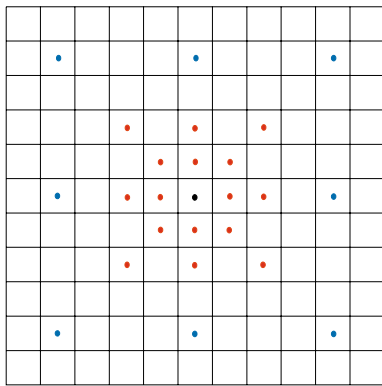


Fig. 5. Hierarchical sampling in DSM. The *black point* denotes the handled surface location. Other *points* with different *colors* show the sampling locations for the irradiance computation of the *black point*

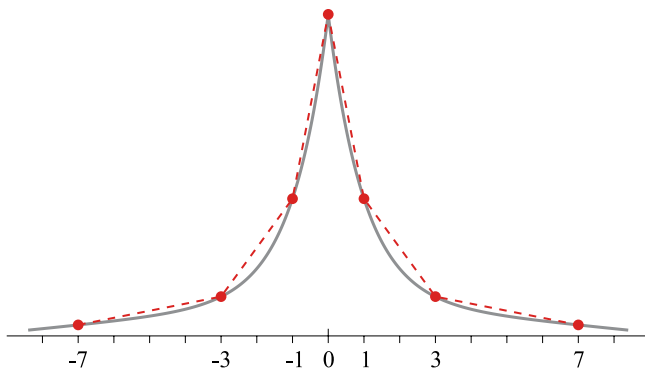


Fig. 6. The approximation of the irradiance response curve

4.3 VTF and FTF

In implementation, there are two ways to implement the scattering processing in GPUs. The first one fulfills the irradiance computation in a per-vertex basis. We call it the VTF mode, which makes use of the newly introduced *vertex texture fetch* feature for accessing the DSM in the vertex processing stage. One of its disadvantages is that the vertex texture accesses are not efficient yet with current graphics hardware [17]. Therefore, VTF mode is suitable for small models, but too expensive for large-sized model. The other mode, called FTF mode, performs the irradiance computation in the fragment processing stage, yielding more accurate results than that of the VTF mode. Although it has image-space complexity, the *early-z-culling* feature of graphics hardware favors efficient culling of hidden surface parts. Our experiments show that the FTF mode leads to higher performance for large-sized models than the VTF mode.

5 Experimental results

We have implemented our algorithms with DirectX 9.0c SDK. Performance has been measured on an AMD AI-



Fig. 7a,b. Our results for the Buddha model: **a** with one point light; **b** with the point light and ambient lights



Fig. 8. Translucent rendering for teapot, bunny and Max Planck data sets with 1024×1024 image resolutions

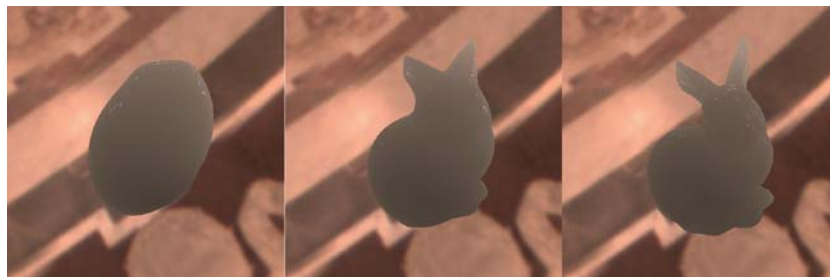


Fig. 9. The selected frames from morphing bunny

Table 1. Performance statistics for various data sets

Data	#T	#V	FTF	VTF
Teapot	2256	1177	53.74	91.80
Buddha	50000	24974	29.95	23.01
Max Planck	50801	25445	35.62	28.93
Bunny	69451	34834	27.73	20.48
Bunny/Egg	50236	25120	29.19	24.12

thone XP 1.84 GHz system with 512 GB memory and an NVidia 7600 GS video card with 256 MB video memory.

We adopted the BSSRDF coefficients presented in [10] for the experiments. Our hardware implementation integrates several commonly used hardware optimization techniques, such as deferred shading and dynamic branching.

We first compared two implementations with and without the ambient lighting effects. Figure 7 demonstrates the results of both schemes. Although we only employ a simple cube map for simulating the ambient lighting effects, it still adds lots of visual realism.

Table 1 lists the performance for various data sets with the FTF and VTF modes. #T and #V denote the triangle and vertex numbers of the models. Apparently, VTF mode is suitable for small-sized models only. We also tested our approach with the morphable models. Note that, our approach can be applied to deformable models directly without special cares. Figure 8 illustrates the rendering results for static models and Fig. 9 shows those of deformable's.

6 Conclusions and future work

This paper presents an efficient image-space rendering approach for highly scattering translucent materials. We divide the simulation procedure into two stages, each of which can be efficiently accelerated with GPUs. Because no pre-computation is involved, our approach is inherently feasible for deformable models. Despite the simplification of the translucent lighting model, our experimental results demonstrate desirable translucency effects. With all the proposed acceleration techniques, it achieves real-time frame rates for large-sized and deformable models.

In our implementations, the influence from the ambient light neglects the self-occlusion or occlusion from other objects, which causes visual reality losing in some cases. We would like to exploit efficient solution to take the ambient occlusion and more general light sources into account. On the other hand, we have great interest in modelling and rendering of inhomogeneous translucent objects as they are quite common in real world. The flattened DSM framework can also be extended to support other complex materials except for BSSRDF. Accelerating the rendering with newest hardware features is also in our near schedule.

Acknowledgement This work is partially supported by 973 program of China (No.2002CB312100), 863 program of China (No. 2006AA01Z314) and NSF of China (No.60503056).

References

1. Blasi, P., Saëc, B.L., Schlick, C.: A rendering algorithm for discrete volume density objects. *Comput. Graph. Forum* **12**(3), 201–210 (1993)
2. Dachsbacher, C., Stamminger, M.: Translucent shadow map. In: *Proceedings of Eurographics Symposium on Rendering 2003*, pp. 197–201 (2003)
3. Donner, C., Jensen, H.W.: Light diffusion in multi-layered translucent materials. In: *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pp. 1032–1039 (2005)
4. Donner, C., Jensen, H.W.: A spectral bssrdf for shading human skin. In: *Proceedings of Eurographics Symposium on Rendering 2006* (2006)
5. Goesele, M., Lensch, H., Lang, J., Fuchs, C., Seidel, H.P.: Disco: acquisition of translucent objects. *ACM Trans. Graph.* **23**(3), 835–844 (2004)
6. Hanrahan, P., Krueger, W.: Reflection from layered surfaces due to subsurface scattering. In: *Proceedings of ACM SIGGRAPH 1993*, pp. 165–174 (1993)
7. Hao, X., Varshney, A.: Real-time rendering of translucent meshes. *ACM Trans. Graph.* **23**(2), 120–142 (2004)
8. Jensen, H.W., Buhler, J.: A rapid hierarchical rendering technique for translucent materials. *ACM Trans. Graph.* **21**(3), 576–581 (2002)
9. Jensen, H.W., Christensen, P.: Efficient simulation of light transport in scenes with participating media using photon maps. In: *Proceedings of ACM SIGGRAPH 1998*, pp. 311–320 (1998)
10. Jensen, H.W., Marschner, S., Levoy, M., Hanrahan, P.: A practical model for subsurface light transport. In: *Proceedings of ACM SIGGRAPH 2001*, pp. 511–518 (2001)
11. Lafortune, E.P., Willems, Y.D.: Rendering participating media with bidirectional path tracing. In: *Proceedings of Eurographics Rendering Workshop 1996*, pp. 91–100 (1996)
12. Lensch, H., Goesele, M., Bekaert, P., Kautz, J.: Interactive rendering of translucent objects. In: *Proceedings of Pacific Graphics 2002*, pp. 214–224 (2002)
13. Li, H., Pellacini, F., Torrance, K.: A hybrid monte carlo method for accurate and efficient subsurface scattering. In: *Proceedings of Eurographics Symposium on Rendering 2005*, pp. 283–290 (2005)
14. Maillot, J., Yahia, H., Verroust, A.: Interactive texture mapping. In: *Proceedings of ACM SIGGRAPH 1993*, pp. 27–34 (1993)
15. Mertens, T., Kautz, J., Bekaert, P., Reeth, F.V., Seidel, H.P.: Efficient rendering of local subsurface scattering. In: *Pacific Graphics 2003*, pp. 51–58 (2003)
16. Mertens, T., Kautz, J., Bekaert, P., Seidel, H.P., Reeth, F.V.: Interactive rendering of translucent deformable objects. In: *Proceedings of Eurographics Rendering Workshop 2003*, pp. 130–140 (2003)
17. NVIDIA: GPU programming guide (2007)
18. Peers, P., vom Berge, K., Matusik, W., Ramamoorthi, R., Lawrence, J., Rusinkiewicz, S., Dutré, P.: A compact factored representation of heterogeneous subsurface scattering. *ACM Trans. Graph.* **25**(3), 746–753 (2006)
19. Rushmeier, H.E., Torrance, K.E.: Extending the radiosity method to include specularly reflecting and translucent materials. *ACM Trans. Graph.* **9**(1), 1–27 (1990)
20. Sillion, F.X.: A unified hierarchical algorithm for global illumination with scattering volumes and object clusters. *IEEE Trans. Vis. Comput. Graph.* **1**(3), 240–254 (1995)
21. Sloan, P.P., Kautz, J., Snyder, J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In: *Proceedings of ACM SIGGRAPH 2002*, pp. 527–536 (2002)
22. Sloan, P.P., Luna, B., Snyder, J.: Local, deformable precomputed radiance transfer. *ACM Trans. Graph.* **24**(3), 1216–1224 (2005)
23. Stam, J.: Multiple scattering as a diffusion process. In: *Proceedings of Eurographics Rendering Workshop 1995*, pp. 41–50 (1995)
24. Surazhsky, V., Gotsman, C.: Explicit surface remeshing. In: *SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 20–30 (2003)
25. Tong, X., Wang, J., Lin, S., Guo, B., Shum, H.Y.: Modeling and rendering of quasi-homogeneous materials. In: *Proceedings of ACM SIGGRAPH 2005*, pp. 1054–1061 (2005)
26. Wang, R., Tran, J., Luebke, D.: All-frequency interactive relighting of translucent objects with single and multiple scattering. *ACM Trans. Graph.* **24**(3), 1202–1207 (2005)
27. Williams, L.: Casting curved shadows on curved surfaces. In: *Proceedings of ACM SIGGRAPH 1978*, pp. 270–274 (1978)



YI GONG received her bachelor's degree from Shanghai Jiao Tong University in 2004 and her master's degree from Zhejiang University in 2006. Now she is a Ph.D. candidate of computer science at University of California, Santa Barbara. Her research work is mainly focused on photo-realistic rendering and GPU algorithms. Besides these, she is also interested in image-based methods and computer vision.

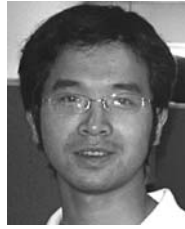
WEI CHEN is an associate professor in State Key Lab of CAD&CG at Zhejiang University. From June 2000 to June 2002, he was a joint Ph.D student in Fraunhofer Institute for Graphics, Darmstadt, Germany and received his Ph.D degree in July 2002. From June 2006 to June



2008 he is a visiting scholar at Purdue University, USA. His current research interests include visualization and efficient modeling and photo-realistic rendering.

LONG ZHANG is currently a Ph.D candidate of the State Key Lab of CAD&CG, Zhejiang University. He received his bachelor's degree in 2003 from the Department of Applied Mathematics at Zhejiang University. His research interests include real-time rendering, visualization and computer animation.

YUN ZENG got his master's degree from Zhejiang University in 2006. Now he is a Ph.D candidate of computer science at SUNY Stony



Brook University. He is interested in research of image and video segmentation as well as editing, GPGPU, computational complexity and optimization.

QUNSHENG PENG is a professor of department of applied mathematics and State Key Lab of CAD&CG at Zhejiang University since 1988. He received his Ph.D degree from the School of Computing studies of East Anglia University, UK in 1983. His research interests include realistic image synthesis, virtual reality, infrared image synthesis, multi-spectral information fusion, computer animation and scientific visualization.

