

# Curvature-Dependent Reflectance Function for Interactive Rendering of Subsurface Scattering



Hiroyuki Kubo<sup>1</sup>, Yoshinori Dobashi<sup>2</sup> and Shigeo Morishima<sup>3</sup>

<sup>1</sup>Graduate School of Advanced Science and Engineering, Waseda University, Japan

<sup>2</sup>Graduate School of Information Science and Technology, Hokkaido University, Japan

<sup>3</sup>Faculty of Science and Engineering, Waseda University, Japan

**Abstract**—Simulating sub-surface scattering is one of the most effective ways to realistically synthesize translucent materials such as marble, milk or human skin. We have developed a curvature-dependent reflectance function (CDRF) which mimics the presence of subsurface scattering. In this approach, we provide only a single parameter that represents the intensity of the incident light scattered in a translucent material. The parameter is not only provided by curve-fitting to a simulated data-set, but also manipulated by an artist. Furthermore, this approach is easily implementable on a GPU and does not require any complicated pre-processing and multi-pass rendering as is often the case in this area of research.

**Index Terms**— local illumination, real-time rendering, reflectance function, subsurface scattering.

## I. INTRODUCTION

Simulating sub-surface scattering is one of the most important issues to realistically synthesize translucent materials and has for more than a decade received a great deal of attention from the rendering research community.

We propose a curvature-dependent reflectance function (CDRF) that mimics the presence of a translucent material which is dominated by higher-order scattering events, such as human skin, marble, jade, and so on. Since the proposed function is a local illumination model, the method synthesizes realistic translucent materials in real-time. The features of our method are as follows.

- Approximating reflectance function of subsurface scattering described by surface curvature.
- computationally lightweight (almost the same cost as Lambert shading.)
- single-pass rendering (saving computation cost and graphics memory.)
- unnecessary mesh parameterization (can be applied various objects compared with [1].)
- can easily be used to stylize subsurface scattering effects.

We focus on the issue of relatively local scale scattering, where the incident light hits the surface, and bleeds into a narrow area. Although, our method does not consider exact

scattering over the whole object, we propose a plausible approach using a local illumination approximation. Fig. 1 shows synthesized images of the Stanford Bunny with different levels of translucency. All images are synthesized in real-time.

Fast approximations of subsurface scattering have been receiving attention from game developers and others. Our method is straightforward for artists to control and much simpler than Kolchin's [2] described below.

## II. PREVIOUS WORK

The rendering of translucent materials has been an interesting research topic for a decade. There are currently two types of approaches to synthesize these objects, one is for off-line rendering and the other is for real-time rendering.

### 2.1 Off-line rendering

The photon mapping [3, 4, 5] is capable of simulating light transport inside a translucent material accurately, though, its computational cost is also known to be notably expensive. Since it basically requires several hours to synthesize a single image, the photon mapping is not practical for real-time rendering.

The method developed by Jensen et al. [6, 7] significantly improves the speed of the simulation. Using a bidirectional surface scattering distribution function (BSSRDF) model, they combine a dipole diffusion approximation with single scattering computation, yet this still cannot produce real-time rendering.

### 2.2 Real-time rendering

In 2007, d'Eon et al. [1, 8] develop a subsurface scattering rendering method using a texture-space diffusion technique. They approximate the profile of multi-pole model using a sum-of-Gaussians formulation, then apply gauss filtering with varying radii of blur to an irradiance map in the texture space. They realize a high quality, GPU accelerated method, however, this also contains two problematic issues that cannot easily be avoided.

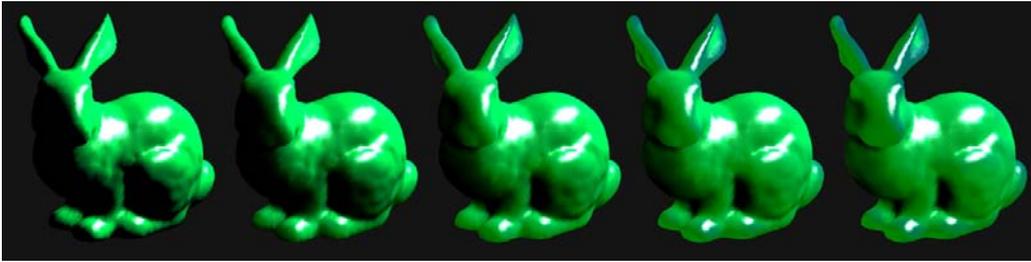


Fig. 1. Synthesized images of translucent objects with different scattering intensity

One is the calculation cost and the size of the required memory. Compared with the latest CPU/GPU, current consumer gaming devices, such as PS3/Xbox360, are unable to compute very rapidly, and are not equipped with sufficient memory. Since their method requires more than ten rendering passes (1-pass for rendering irradiance texture, 12-passes for 6 Gaussian convolutions, 1-pass for stretching and 1-pass for composition), it is computationally expensive and requires a huge capacity for the graphics memory.

The other problem is the necessity of mesh parameterization. Since their technique approximates subsurface scattering by blurring irradiance in the texture space, every vertex of the model is necessary to be parameterized. Mesh parameterization of arbitrary shape is still a challenging issue. Therefore, their technique is not very practical for current consumer gaming devices.

Kolchin [2] develop a curvature-based shading method. The basic idea behind our method is similar to that of Kolchin's, in that the effects of subsurface scattering depend on curvature. Curvature of a surface can be used to derive a local-illumination approximation of the subsurface scattering.

Compared with these works, we propose a much faster technique than Jensen's dipole model and d'Eon's texture space diffusion, and a significantly simpler and easier to control than that of Kolchin's.

### III. OUR APPROACH

According to previous work [6, 7], higher order scattering often hides local bumpy shading on the surface because of light diffusion. Fig. 2(a) and (b) are synthesized images of the Stanford bunny without and with scattering. It is clear that the small wrinkles on the bunny's body are less visible as shown in



(a) w/o scattering (Lambert)

(b) w/ scattering (Dipole)

Fig. 2. Comparison of the bunny without and with scattering

Fig. 2(b).

We realize that the effects of subsurface scattering tend to be more noticeable on small, intricate objects than on simpler, flatter ones, which indicates that the surface complexity largely determines these effects. For the purposes of our research, we decided to use curvature to represent the surface complexity, combined with a simple local illumination model. We propose a curvature-dependent reflectance function (CDRF) for efficiently rendering translucent materials. CDRF is not a global illumination model such as BSSRDF, but a local illumination model. Compared with BRDF which is typically used for local illumination of opaque objects, CDRF depends not only on the incident direction  $\omega_i$  and the normal vector  $\mathbf{n}$  but also on surface curvature  $\kappa$ .

When using the CDRF, there is only one parameter  $\sigma_0$  that represents the degree of light scattering inside the material.  $\sigma_0$  is not only provided by curve-fitting to a simulated data-set, but also manipulated by an artist. Furthermore, our method is easily implementable on a GPU and does not require any complicated pre-processing steps and multi-pass rendering passes such as required by previous method [9].

To synthesize an image of translucent materials, we used the following procedure shown in Fig. 3. The input is a 3D mesh object, and the output is a synthesized image. Prior to rendering, our method computes the surface curvature of the mesh on

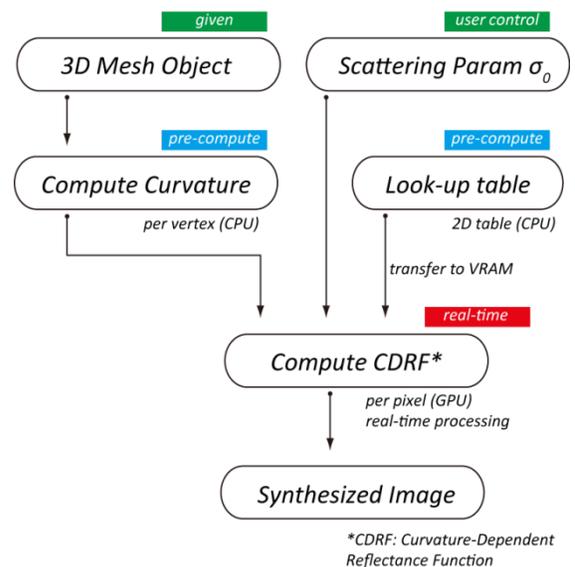


Fig. 3. Workflow

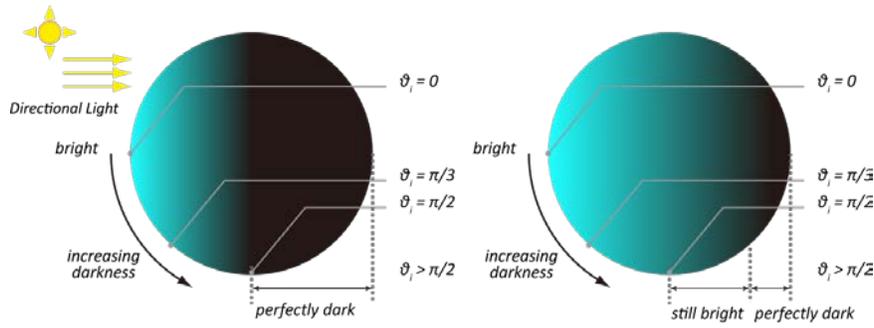


Fig. 4. Spheres without scattering(left) and with scattering(right)

a CPU using the method proposed by [10, 11]. The value of the CDRF depends on the curvature  $\kappa$ , scattering parameter  $\sigma_0$  and  $\omega_i \cdot \mathbf{n}$ . We pre-compute the CDRF for all combinations of these parameters and store the value in a 2-dimensional look-up table (LUT). Then, using this LUT for faster computation, we calculate the CDRF in the pixel shader to render translucent materials in real-time.

#### IV. CURVATURE-DEPENDENT REFLECTANCE FUNCTION

##### 4.1 Multiple-scattering inside translucent spheres

The curvature  $\kappa$  on a sphere of radius  $r$  is  $1/r$ . In our method, we approximate an arbitrary surface locally as a sphere of the corresponding curvature. The lighting model of a surface with curvature  $\kappa$  is represented by that of a sphere, radius  $r = 1/\kappa$ . In other words, we treat radiance on a sphere as a primitive of the lighting model.

Before discussing the phenomenon of subsurface scattering inside a translucent material, we start by reviewing the reflection on a Lambertian surface without scattering. Suppose that a Lambertian sphere, radius  $r$ , is illuminated by a directional light from the left (Fig. 4-left). The diffuse radiance  $L_d$  of the sphere is determined by the Lambertian model  $L_d = k_d \max(\cos \theta_i, 0)$ , where  $k_d$  is a diffuse coefficient and  $\theta_i$  is incident light angle. The radiance on the area where  $\theta_i > \pi/2$  is zero (perfectly dark), because there is no direct illuminations. Next, let us consider a sphere made of a translucent material illuminated from the left (Fig. 4-right). The radiance on the left hand side of the translucent sphere is very similar to a Lambertian sphere. However, on the area, where the incident angle  $\theta_i$  is just  $\pi/2$  greater, we can observe a slight, but not quite zero (still bright), radiance due to subsurface scattering inside the material.

Meanwhile, consider several translucent spheres of the same material; the only difference is their radius. The spheres are illuminated by a directional light from the left. Then, the smaller the radius is, the more noticeable the effect of subsurface scattering becomes, as shown in Fig. 5.

Therefore, we realize that the effect of subsurface scattering has a strong correlation with the radius of the sphere. As stated

above, we approximate an arbitrary surface as a sphere with the corresponding radius. Thus, subsurface scattering effect is highly dependent on the curvature.

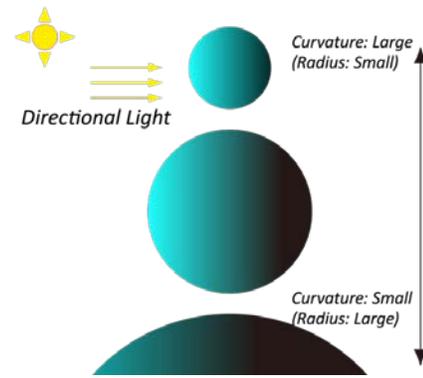


Fig. 5. Different size of spheres with scattering

##### 4.2 Curvature-Dependent Reflectance Function

To represent the subsurface scattering effect, we propose curvature-dependent reflectance function (CDRF)  $f_r^c(\theta_i, \kappa)$ . CDRF is defined by a convolution of the incident light energy  $E(\theta_i)$  and a Gauss function  $g(\theta_i, \sigma)$  to provide blurring effect.

$$f_r^c(\theta_i, \kappa) = (E * g)(\theta_i) \quad (1)$$

where,

$$E(\theta_i) = \max(\cos \theta_i, 0) \quad (2)$$

$$g(\theta_i, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\theta_i^2}{2\sigma^2}\right) \quad (3)$$

$\sigma$ , which represents the scattering intensity inside the material, depends on the material properties and the radius of the sphere.

Again, consider a scattering with a translucent sphere, radius  $r$ . We assume that the incident light scatters along an arc on the sphere. For a given length of mean free path  $l$  in the material, the pass is along an arc, then the central angle of the arc is  $l/r$ .

Accordingly, observing on the  $\theta_i$  axis,  $\sigma$  is supposed to be, relatively, in inverse proportion to the radius  $r$ . Therefore, we assume that

$$\sigma(\kappa) = \frac{\sigma_0}{r} = \sigma_0 \kappa \quad (4)$$

$\sigma_0$  corresponds to the scattering intensity for the unit sphere.

Using this reflectance function, diffuse radiance  $L_d$  on a surface is calculated from the following equation,

$$L_d = \int_{\Omega} k_d f_r^c(\theta_i, \kappa) L_i(\omega_i) d\omega_i \quad (5)$$

Note that,  $L_i(\omega_i)$  is the intensity of incident light from the direction  $\omega_i$ ,  $k_d$  is the diffuse albedo, and  $\Omega$  is over all incident light directions.

To confirm the validity of the formula in Eq. 1, we rendered several spheres of varying radii to reveal the relationship between curvature and radiance using the photon tracing. The spheres are illuminated by a directional light from the left hand side, as shown in Fig. 6(a). The calculated color of each pixel on the equatorial line represents the relationship between the angle of incidence and the radiance of the sphere's particular curvature(Fig. 6(b)). Then, we fit curves to the obtained data using the CDRF formula in Eq.1. The curves are not exactly the same, but are similar and the fit is good. This means that the approximate CDRF formula we propose is fairly accurate. In terms of the way to determine the scattering parameter  $\sigma_0$ , it is not only provided by curve-fitting to a simulated data-set, but also manipulated by an artist.

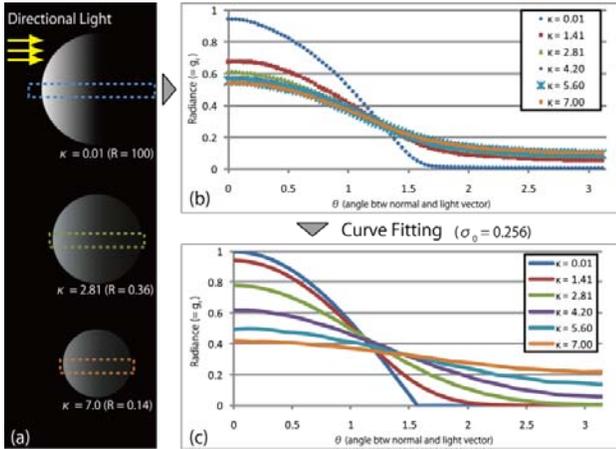


Fig. 6. Curve fitting

### 4.3 Rendering spheres

As described above, the effect of subsurface scattering highly depends on the surface curvature. Fig. 7 (a)-(c) are synthesized images of spheres that resemble the appearance of human skin. It is known that the mean free path of red light through human skin is relatively longer than other colors. Fig. 7(a) shows a Lambertian sphere without scattering, and Fig. 7(b), (c) are images of translucent spheres, synthesized using CDRF. Although the sphere (b) is a translucent, the appearance is almost the same as the Lambertian sphere (a) because its radius is large. Instead of the image (b), thin red scattering is found on the smaller radius sphere (c).



(a) Lambert (b) Our method (r = 10:0) (c) Our method (r = 1:0)

Fig. 7. Without and with scattering spheres of different radius

## V. IMPLEMENTATION

To synthesize translucent materials using CDRF, it is necessary to compute the convolution in Eq. 1. However, for current consumer gaming devices, numerical computation of the convolution is computationally expensive.

The value of CDRF will change depending on the product of the scattering parameter and curvature  $\sigma_0 \kappa$  and on the dot product of the light and the normal vector  $\omega_i \cdot \mathbf{n}$ . We pre-compute these for all combinations of  $\sigma_0 \kappa$  and  $\omega_i \cdot \mathbf{n}$  and store the value in a 2-dimensional look-up table (LUT). We prepare this LUT as a 2D texture and it is accessed based on  $\sigma_0 \kappa$  and  $\omega_i \cdot \mathbf{n}$  by a pixel shader on the GPU.

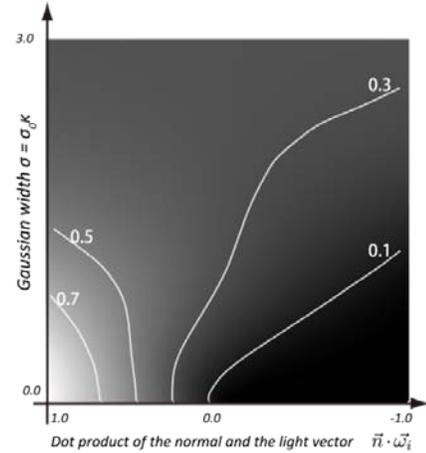


Fig. 8. CDRF look-up table

The horizontal axis of this LUT is the cosine of incident angle  $\cos \theta_i = \omega_i \cdot \mathbf{n} \in [-1.0, 1.0]$ , and the vertical axis is  $\sigma = \sigma_0 \kappa \in [0.0, 3.0]$ . The LUT is a single channel, floating point (R32F) format. Based on a given scattering parameter  $\sigma_0$ , the incident angle  $\theta_i$ , and the curvature  $\kappa$ , we are able to compute  $f_r^c(\theta_i, \kappa)$  by sampling this LUT.

Additionally, it is possible to determine the scattering parameter  $\sigma_0$  from a diffuse color map.

$$\sigma_0 = d_c^2 \quad (6)$$

where  $d_c$  is the value of diffuse color map of a corresponding texel. For example, for human skin, the scattering parameter is not the same everywhere, because human skin contains regional features such as moles and skin mottles. A diffuse color map generated from photographs or painted by an artist

often contains these features. Therefore, by obtaining the scattering parameter from a diffuse color map, we are able to synthesize more realistic human skin.

## VI. RESULTS

The results are summarized in Table 1, Fig. 1 and 9-12. These images are obtained from our implementation running on a 3.06GHz Intel® Core™2 Duo CPU with NVIDIA® GeForce® 9300M GS GPU. We implement our algorithm as a hardware-accelerated real-time renderer. The renderer is implemented as a pixel shader in HLSL.

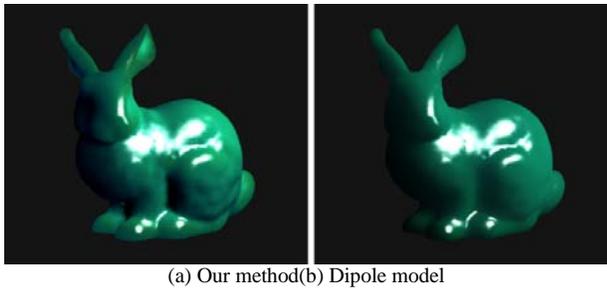


Fig. 9. Comparison of our method and dipole model

Fig. 9(a) and (b) show a comparison of synthesized images using our method and the dipole model. Although, the appearance around the front part of the right leg and the ears of the bunny is slightly different, the overall appearance of the bunny rendered by our method (Fig. 9 (a)) is similar to that in Fig. 9 (b). Fig. 9 (a) is rendered in real-time but the computational time to synthesize Fig. 9 (b) using the dipole model is 22 sec.

Additionally, to validate our algorithm, we test our algorithm using a human head model. Fig.10 (a) displays the surface curvature of the model. Hot colors represent higher curvature,

and vice versa. For comparison, Fig.10(b) and (e) are rendered using the Lambert shading method, and Fig.10(c) and (f) are rendered using the texture blurring method [1]. Fig.10(d) and (g) are synthesized images using our CDRF. Compared with the Lambertian, we are able to synthesize more natural, softer shading effects especially around the nose and cheek. Although the appearance of ours (Fig.10(d),(g)) may not be much more effective than that of the texture blurring method (Fig.10(c),(f)), the advantages of our method are the lower calculation cost and the non-necessity of mesh parameterization.

Table 1 shows the frame rates for synthesizing images using

TABLE 1: RENDERING SPEED (512\*512 PIXELS)

technique	frame rate [fps]	
	human head (20,154 vtx.)	bunny (48,903 vtx.)
Lambert	575.2	693.2
Texture Blur	41.5	35.3
Our method w/o LUT	133.2	110.9
<b>Our method w/ LUT</b>	<b>492.1</b>	<b>482.6</b>

the Lambertian method, the texture blurring method, our method without LUT, and our method with LUT. Compared with the Lambert shading, the additional cost to compute the CDRF is the cost for only 3-times texture sampling and a few arithmetic operations, thus the frame rate of our method is not significantly reduced. Since our method is implemented as a single-pass shader, according to our implementation and computational environment, it is over 10 times faster than the texture blurring method which requires more than 10-passes. Furthermore, our method with LUT is more than twice as fast as that without LUT.

Fig. 1 and 11 show other examples of green and blue jewel stones, something like a jade and a sapphire. In Fig. 1, we set

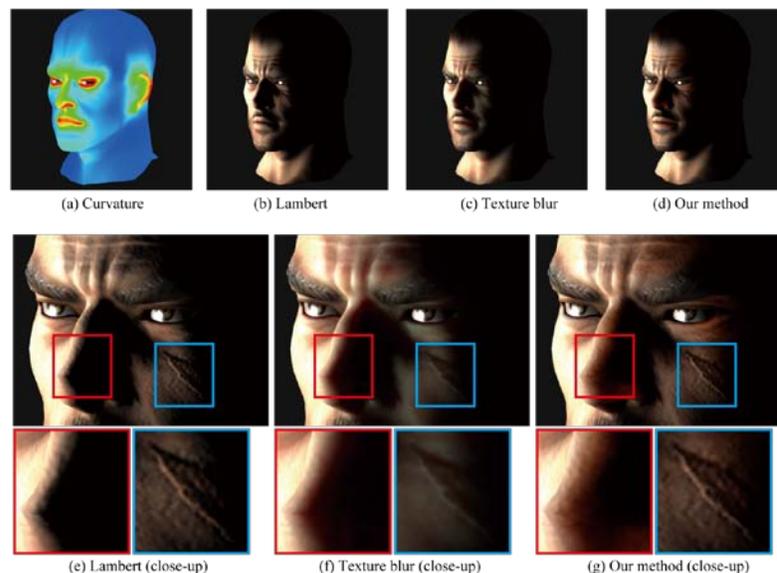


Fig. 10. A human head. Scattering parameter is obtained from a diffusion texture ©2009 TECMO KOEI GAMES CO., LTD. Team NINJA All rights reserved. (Color Plate 6)

different levels of scattering parameters, and the appearance we cannot consider global object shape such as the object

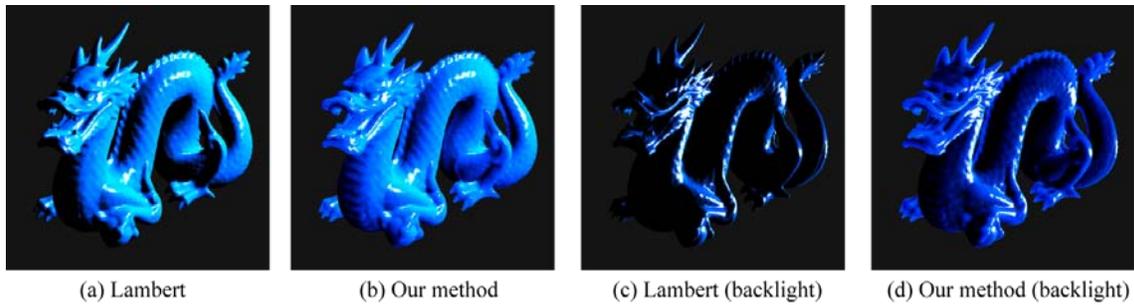


Fig. 11. Dragon without and with subsurface scattering. Scattering parameter 0 for each color channel is (0:09; 0:13; 0:23)



Fig. 12. A female without and with subsurface scattering.  
©2009 TECMO KOEI GAMES CO., LTD. Team NINJA All rights reserved. (Color Plate 7)

changes straightforwardly with the value of the parameter. Fig. 11 shows the dragon with different light directions, the light in front of the object, (a) and (b), and from behind, (c) and (d). Furthermore, we rendered another example of a human skin as shown in Fig. 12. Compared with the hard appearance of the image on the left, our method is able to synthesize skin with a soft appearance.

## VII. CONCLUSION AND DISCUSSION

We have developed a method for approximating the effects of subsurface scattering using a curvature-dependent reflectance function. Since the function is a local illumination model, we are able to synthesize realistic translucent materials in real-time. Furthermore, our system can easily be used to stylize subsurface scattering effects because only one parameter  $\sigma_0$  is required.

We approximate an object surface locally as the surface of a sphere of corresponding radius according to the curvature. Thus,

thickness. Therefore, compared with a real object, the difference in appearance tends to be more noticeable, when the object is especially thin such as for leaves and papers. However, our method can be applied to almost all practical scenes, and it is possible to synthesize translucent materials plausibly. In our future work, we will apply our techniques to deformable objects by computing curvature on a GPU.

## ACKNOWLEDGEMENTS

We appreciate the feedback offered by TECMO KOEI GAMES CO., LTD.

## REFERENCES

- [1] Eugene d'Eon, David Luebke, and Eric Enderton. Efficient rendering of humanskin. In *Rendering Techniques 2007: 18th Eurographics Workshop on Rendering*, pages 147-158, June 2007.

- [2] Konstantin Kolchin. Curvature-based shading of translucent materials, such as human skin. In Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia, *GRAPHITE '07*, pages 239–242, New York, NY, USA, 2007. ACM.
- [3] James T. Kajiya. The rendering equation. In Proceedings of the 13th annual conference on Computer graphics and interactive techniques, *SIGGRAPH '86*, pages 143–150, New York, NY, USA, 1986. ACM.
- [4] Julie Dorsey, Alan Edelman, HenrikWann Jensen, Justin Legakis, and Hans K ohling Pedersen. Modeling and rendering of weathered stone. In Proceedings of the 26th annual conference on Computer graphics and interactive techniques, *SIGGRAPH '99*, pages 225–234, New York, NY, USA, 1999. ACM Press/Addison- Wesley Publishing Co.
- [5] Henrik Wann Jensen, Justin Legakis, and Julie Dorsey. Rendering of wet materials. In *Rendering Techniques '99*, pages 273–282, 1999.
- [6] Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, *SIGGRAPH '01*, pages 511–518, New York, NY, USA, 2001. ACM.
- [7] Henrik Wann Jensen and Juan Buhler. A rapid hierarchical rendering technique for translucent materials. *ACM Trans. Graph.*, 21:576–581, July 2002.
- [8] Hubert Nguyen. GPU Gems 3. Addison- Wesley Professional, 2007.
- [9] Tom Mertens, Jan Kautz, Philippe Bekaert, Hans-Peter Seidelz, and Frank Van Reeth. Interactive rendering of translucent deformable objects. In Proceedings of the 14th Eurographics workshop on Rendering, *EGRW '03*, pages 130–140, Aire-la- Ville, Switzerland, Switzerland, 2003. Eurographics Association.
- [10] Mark Meyer, Mathieu Desbrun, Peter Schr oder, and Alan H. Barr. Discrete differential–geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, pages 35–57, 2003.
- [11] Jack Goldfeather and Victoria Interrante. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Trans. Graph.*, 23:45–63, January 2004.



**Hiroyuki Kubois** is a doctor course student at Graduate School of Advanced Science and Engineering, Waseda University. His research interests include computer graphics and computer animation. He received the BS and MS degrees from Waseda University, in 2006 and 2008, respectively. He is also a research fellow of the Japan Society for the Promotion of Science since 2008.



**Yoshinori Dobashi** is an Associate Professor at Hokkaido University in the Graduate School of Information Science and Technology, Japan since 2006. His research interests center on computer graphics including realistic image synthesis, efficient rendering, and sound modeling for virtual reality applications. Dobashi received his BE, ME and PhD in Engineering in 1992, 1994, and 1997, respectively, from Hiroshima University. He worked at Hiroshima City University from 1997 to 2000 as a research associate.



**Shigeo Morishima** is a Professor of School of Science and Engineering, Waseda University Japan since 2004. His research interests include Computer Graphics, Computer Vision, Multimodal Signal Processing and Human Computer Interaction. He received the B.S., M.S. and Ph.D. degrees, all in Electrical Engineering from the University of Tokyo, Tokyo, Japan, in 1982, 1984, and 1987, respectively. From 1987 to 2001, he was an associate professor and from 2001 to 2004, a professor of Seikei University, Tokyo.