Real-time Rendering of Heterogeneous Translucent Objects using Voxel Number Map

Keisuke Mochida[1], Midori Okamoto[1], Hiroyuki Kubo[2] and Shigeo Morishima[3]

[1]Waseda University, Japan, [2]Nara Institute of Science and Technology, Japan, [3]Waseda Research Institute for Science and Engineering, Japan

Abstract

Rendering of tranlucent objects enhances the reality of computer graphics, however, it is still computationally expensive. In this paper, we introduce a real-time rendering technique for heterogeneous translucent objects that contain complex structure inside. First, for the precomputation, we convert mesh models into voxels and generate Look-Up-Table in which the optical thickness between two surface voxels is stored. Second, we compute radiance in real-time using the precomputed optical thickness. At this time, we generate Voxel-Number-Map to fetch the texel value of the Look-Up-Table in GPU. Using Look-Up-Table and Voxel-Number-Map, our method can render translucent objects with cavities and different media inside in real-time.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

Rendering techniques for tranlucent objects, such as skin, milk, marble and wax, are necessary for generating realistic computer graphics, however, they are still computationally expensive. In particular, rendering translucent objects having complicated structure, such as jellyfish, flower, and plastic bottle, in real-time is a challenging task. *Translucent Shadow Maps* (TSM) [DS03] provides a high speed rendering of translucent objects whose translucency depend on thickness using *shadow maps* [Wil78]. However, in case of concave shape, TSM can not generate accurate results because thickness contains calculation errors. Kosaka et al. [KHKM12] have extended TSM method using *Multi-view Depth Maps*. Their method can obtain more accurate thickness in TSM in case of concave shape, however, *Multi-view Depth Maps* are not correctly generated for objects that contain cavities inside. In addition, this method can not render heterogeneous objects.

In this paper, we propose a novel technique for real-time rendering of heterogeneous translucent objects that contain cavities and different media inside. Our rendering method is suitable for heterogeneous translucent objects in which single-scattering effects are dominant and increase in the radiance due to in-scattering effects hardly occurs. In this case, translucency highly depends on the optical thickness which gives a measure of how transparent media are to radiation passing through inside. In our method, the optical thickness is obtained by precomputation and stored into a texture called *Look-Up-Table* (LUT). Furethermore, to refer to the optical thickness stored in LUT at a high speed in GPU, we generate *Voxel-Number-Map* (VNM) in screen space from light's view. VNM has texture coordinate values on objects surface for accessing LUT.

2. Our Method

Our method contains two parts: (1) the precomputing part and (2) the run-time part. In the precomputing part, voxelization of objects and computation of the optical thickness of objects are executed. In the run-time part, radiance of the object surface due to sub-surface scattering is computed using the precomputed optical thickness.

2.1. Precomputation of Optical Thickness

In our method, the optical thickness of objects is precomputed and stored into every pixel of LUT before the rendering. For generating LUT, we first voxelize and discretize the structure of objects. By sampling the surface of the object by voxelization, the optical thickness can be stored into the LUT. Second, we extract the surface voxels from the whole voxel and number each surface voxel. At the same time, we make the number of the surface voxels correspond to each texture coordinate (u,v) original object's vertex has. Due to this numbering, it is made possible to fetch the texel value of LUT in GPU at a high speed. Finaly, we compute the optical thickness between two surface voxels for each medium. Optical thickness obtained by this computing is stored into every pixel of LUT.

2.2. Computation of Radiance

In the run-time part, we compute the translucency due to sub-surface scattering. When the incident light at a surface point $\vec{x}_{in}$ is transported through the object and emitted in the direction of $\vec{\omega}_{out}$ from a surface point $\vec{x}_{out}$, the final radiance $L(\vec{x}_{out}, \vec{\omega}_{out})$ at an

DOI: 10.2312/egp.20161037

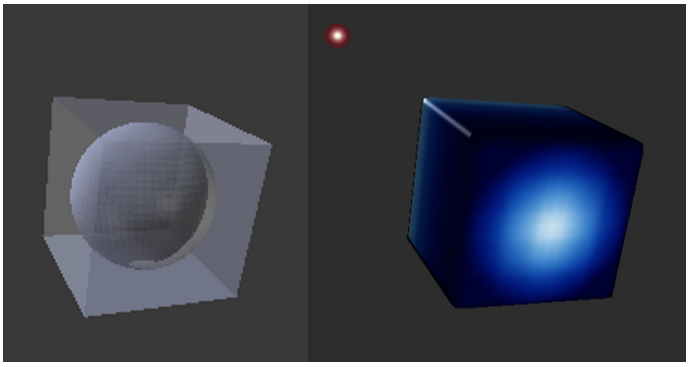


Figure 1: *A translucent cube model containing a spherical cavity (left) and rendering result (right).*

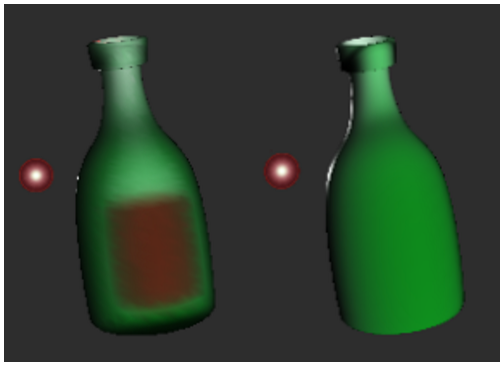


Figure 2: *Comparison between our result (left) and TSM result (right) for a translucent bottle containing wine.*

emittion point is simply:

$$L(\vec{x}_{out},\vec{\omega}_{out}) = \frac{1}{\pi}F_t(\eta,\vec{\omega}_{out})\int_S E(\vec{x}_{in})R_d(\vec{x}_{in},\vec{x}_{out})d\vec{x}_{in} \quad (1)$$

where, $F_t(\eta,\vec{\omega}_{out})$ is the Fresnel transmittance term at the interfaces between the layers, η is the relative index of refraction, $E(\vec{x}_{in})$ is irradiance at a surface point and $R_d(\vec{x}_{in},\vec{x}_{out})$ is the diffuse subsurface reflectance function. For an irradiance impulse $I(\vec{\omega}_{in})$ from a light source in the direction of $\vec{\omega}_{in}$, in the above integral term $E(\vec{x}_{in})$ is computed by the following:

$$E(\vec{x}_{in}) = F_t(\eta,\vec{\omega}_{in})|N(\vec{x}_{in})\cdot\vec{\omega}_{in}|I(\vec{\omega}_{in}), \quad (2)$$

and $R_d(\vec{x}_{in},\vec{x}_{out})$ in Equation 1 is computed by

$$R_d(\vec{x}_{in},\vec{x}_{out}) = e^{-\tau(\vec{x}_{in},\vec{x}_{out})} \quad (3)$$

where, $N(\vec{x}_{in})$ is a surface normal at $\vec{x}_{in}$, and $\tau(\vec{x}_{in},\vec{x}_{out})$ is the optical thickness between an incedent point and an emission point. The optical thickness can be written as:

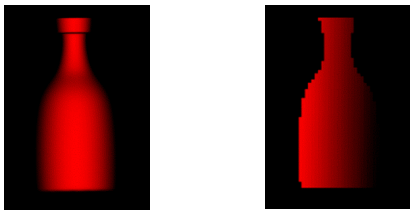
$$\tau(\vec{x}_{in},\vec{x}_{out}) = \int_{\vec{x}_{in}}^{\vec{x}_{out}} \sigma_t(\vec{x'})d\vec{x'} \quad (4)$$

where, $\sigma_t(\vec{x'})$ is extinction coefficient. For this computation, we assume that in-scattering does not occur and effects of reflection and refraction at interfaces between the media inside the object are sufficiently small.

In our method, the computation is separated two passes. In the first pass, Equation 2 is computed from the light's view and results are stored into every pixel of *Irradiance Map* (Figure 3). At the same time, VNM is generated from the light's view (Figure 3). VNM has the numbers of the surface voxel. Using VNM in the second pass, it is made possible to fetch the texture coordinate of LUT. In the second pass, Equation 1 is computed from the user's view. $R_d(\vec{x}_{in},\vec{x}_{out})$ in Equation 1 is computed using the optical thickness $\tau(\vec{x}_{in},\vec{x}_{out})$ stored in LUT. The integral in Equation 1 is computed using the 21-taps sampling filter the same as TSM method.

3. Results

We implement our algorithm in C++ and DirectX11. Results were generated on a Intel Core i5-4200M 2.50 GHz and an Intel HD Graphics 4600. Figure 1 shows a result of a simple model, a translucent cube containing a spherical cavity inside (2,020 vertices). The dividing number of voxel is $25\times25\times25$ and the FPS is 405. In our method, light transmission of middle part by the effect of cavity can be realized. Figure 2 shows a comparison between our method and TSM for a translucent bottle containing wine inside (10,604 vertices). In our method, the dividing number of voxel is $60\times60\times60$ and the FPS is 238. Our method can realize the rendering for the multi-layered object. In above our result, the resolution of all maps is 1024×1024.



(a)Irradiance Map (b)Voxel Number Map

Figure 3: *The texture maps generated from the light's view.*

4. Conclusions and Future Work

We have presented a novel technique for real-time rendering of heterogeneous translucent objects that contain cavities and different media inside using the precomputed optical thickness and *Voxel-Number-Maps*. In the future, we consider effects of reflection and refraction at interfaces between the media inside objects using the precomputed refraction map.

References

[DS03] DACHSBACHER C., STAMMINGER M.: Translucent shadow maps. *Rendering Techniques 2003* (2003), 197–201. 1

[KHKM12] KOSAKA T., HATTORI T., KUBO H., MORISHIMA S.: Rapid and authentic rendering of translucent materials using depth-maps from multi-viewpoint, 2012. 1

[Wil78] WILLIAMS L.: Casting curved shadows on curved surfaces. In *ACM Siggraph Computer Graphics* (1978), vol. 12, ACM, pp. 270–274. 1