Course Project of 2018 Spring Computer Graphics

Rendering Translucent Object Using OptiX

Jiajian Zhang 18613287 zhangjj1@shanghaitech.edu.cn

Muyang Zou 79351195 zoumy@shanghaitech.edu.cn

Introduction

Translucent object can be hard to render. However, with the full use of GPU's computing power, this task can be achieved in a simple and quick way. OptiX is a framework designed by Nvidia used to easily get parallel calculation in ray tracing process. In our project, the render can even be done in real-time.

Theory

The light distribution in highly scattering media tends to become isotropic. This is true even if the initial light source distribution and the phase function are highly anisotropic. [1] We can assume that after a light shooting into the surface, it will randomly go to all the direction. This is the same for the ray we tracing from the camera. Translucent objects have such kind of nature. We can simulate this process and as far as possible to reach the real effect.

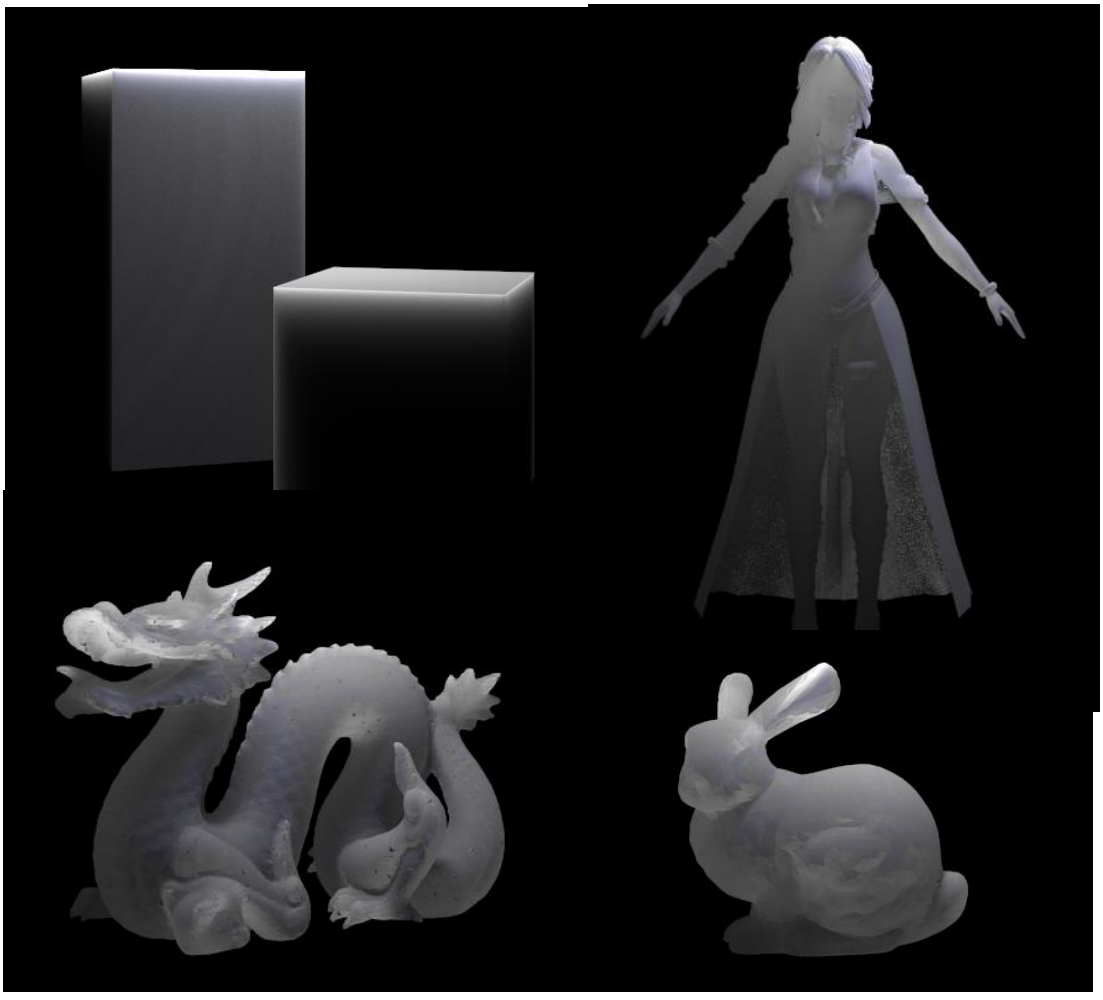
The algorithm is simple. When a ray hit the surface of a translucent object, we

[1] A Practical Model for Subsurface Light Transport Henrik Wann Jensen Stephen R. Marschner

first calculate the direct light emission of the hit point $\mathrm{p_h}$. Then from $\mathrm{p_h}$, shoot rays $\mathrm{r_{in}}$ into the object in different direction. We call $\mathrm{r_{in}}$ as a scatter ray. The rays are sampled uniformly in a hemisphere space. Find the insect point $\mathrm{p_{insect}}$ of $\mathrm{r_{in}}$, calculate the direct light emission of $\mathrm{p_{insect}}$, add it to the result of the $\mathrm{p_h}$ multiplied by a decrease function $\mathrm{f}\,(p_h, p_{insect}) = 1 \,/\, \exp(\,1 + distance(p_h, p_{insect}) \;/\; decs)$. In the function decs is a parameter set for different sizes and materials of the object.

Result

We test four models with different materials and different transparencies.



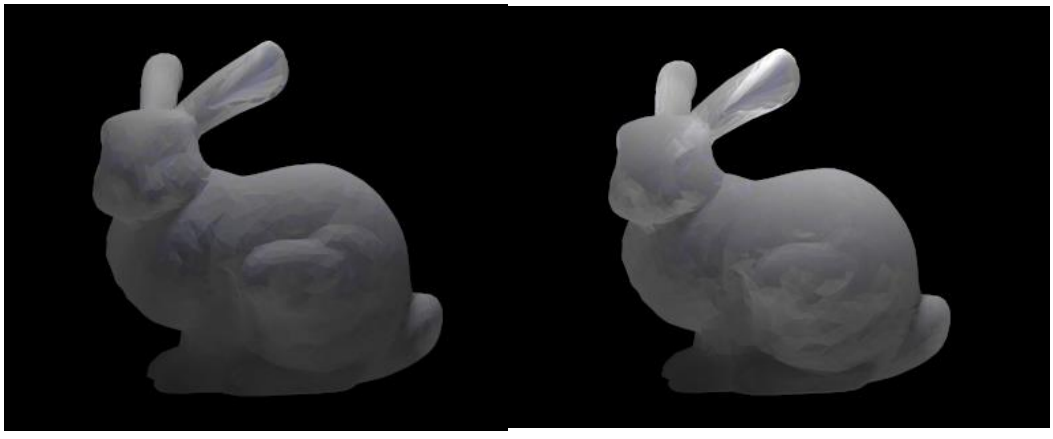
For the two blocks, we set the transparency of the them as a low value, so it looks like marble. For the dragon and bunny model, we set a higher transparency so the look like jade. For the princess model, we can see even under the cover of the skirt, the performance of legs is still acceptable. The performance of the skirt is also proved that the algorithm is robust for narrow, thin and detailed object.

The time used for rendering is also short. Under GTX960M 4G, Intel Core i5 2.3GHz, the dragon and princess used about 30 seconds and the bunny uses only less than 20s. All the images shoot 256 rays per pixel and 15 scatter rays per hit point. If we shoot about 4 rays per pixel we can get a real-time rendering result with a DPS around 10.

Detailed

1. Number of scatter rays

This is the most important term that influence the performance of the algorithm. For the left image, we shoot 5 scatter rays for each hit point. For the right image, we shoot 15 scatter rays for each hit point. We can see the performance of the 15-rays version is a lot better.



However, as my GPU is not that good, the maximum number of scatter rays is about 15. This is because OptiX put the calculate into processers of GPU. If the number of rays we shoot at the same is too large, my GPU does not have that much of processers and might crush. The good thing is that as the calculate is fully divided into GPU processers, the time cost is low.

2. Color and texture

We can also set different color and texture for our models. When calculate the light emission of $\mathrm{p_h}$ and $\mathrm{p\_insect}$, we also calculate the color of them or mapped them into a texture.