Photon Mapping

Shiyao Wang

COMP 6906, Fundamentals of Computer Graphics

Professor Gong, Minglun

April 19, 2016,

Photon Mapping

Shiyao Wang

Abstract

The topic of this project is Photon Mapping, which is a rendering algorithm that approximates Global Illumination. We implement three different progressive rendering approaches to compare the results.

1. Introduction

Photon mapping is an advanced rendering technique based on irradiance caching. It speeds up global illumination computation by caching photons on the non-specular geometry and using them in a ray trace step to obtain global illumination effects. Since photons stored have a certain size, it is not feasible to store an extremely large number of photons. We bypass this limitation by implementing progressive photon mapping where every iteration shoots a certain number of photons in the scene and the irradiance is averaged across multiple iterations.

2. Related Works

Real-time ray tracing is a hot topic in graphics and has been called the "future of computer graphics". However, it has also been said that ray tracing will "always be the future of computer graphics". This is because it involves a large compute to memory ratio. With massively parallel processors, i.e. modern GPUs, being made more and more easy to program upon, this future goal seems to be coming closer and closer.

Photon mapping was proposed by Henrik Wann Jensen [1]. It simulates photons that are shot out from a light source and stored on non-specular surfaces. These photons carry flux and color with them. They are changed when the photon hits a surface according to the material properties of that surface. Photons are then either stored, reflected or absorbed based on a probabilistic model related to the bidirectional reflectance distribution function (BRDF).

To make sure that the density estimation stage of photon mapping is fast, a spatial acceleration structure is essential. Without it, one would be traversing all the photons of each ray intersection to find the photons that are contributing. Considering the fact that photons are in the order of millions, this linear check is an expensive operation. A KD-tree is an acceleration structure suggested by the creator of photon mapping, Jensen [2].

An extension to photon mapping involves removing the limit on the number of photons that can be stored by shooting a new set of photons each iteration and averaging the results [3, 4]. This helps in getting a better estimate of indirect illumination, especially at caustic regions and regions that are completely in shadows with respect to ray tracing. This progressive photon mapping technique allows for millions and millions of photons contributing but only requires storage for far fewer photons.

3. Methodology

We implemented stochastic progressive photon mapping with a KD-tree for photon storage. This section will describe each of these in detail.

3.1. Photon Mapping

For photon mapping, we do a light trace from the light source after randomly generate a direction. Each photon's initial position is chosen randomly on the surface of the emitting light source. Currently, only point can be chosen as emitting sources. A photon is bounced around by intersecting with all the objects in the scene. If the object that it intersects with is a diffuse object, the color carried by the photon is multiplied with the color of the object and the photon is marked as stored and then continued intersecting by a new direction. If the surface is reflective or refractive, the photon's color is modulated but it is not marked as stored. The new direction of the photon is decided based on the BRDF. These photons are the photons that contribute to the caustics. We are not storing a separate caustic photon map, but using the global photon map for the caustics as well since the number of photons is quite high.

The number of photons to be sent out is pre-determined and they are distributed amongst light sources based on the energy of the light source. Each photon should scale its flux according to the energy of the light source and the total amount of photons.

Now, for density estimation using photons stored, we ray trace an image to collect its illumination. Once a ray hits a surface, the k nearest photons are gathered and their weighted flux is used to estimate the irradiance at that point.

3.2. KD-Tree

For finding the k-nearest neighbors among all the intersections, we could loop through all the photons but that takes a long time. Thus, we decided to implement a KD-Tree. While KD-Tree are not the best for finding the k-nearest neighbors, it is still suitable for accelerating the performance.

3.3. BRDF

The BRDF is the "Bidirectional Reflectance Distribution Function." It provides the reflectance of a target as a function of illumination geometry and viewing geometry. The BRDF is determined by the structural and optical properties of the surface, such as shadow-casting, reflection, refraction, absorption, and emission by surface elements. In our project, BRDF is approximated by combining the ideal diffuse BRDF, ideal mirror reflection BRDF and ideal refraction BRDF by corresponding albedos. The albedos are calculating according to Fresnel equations and the refraction angle is obtained by Snell's (Descartes') law.

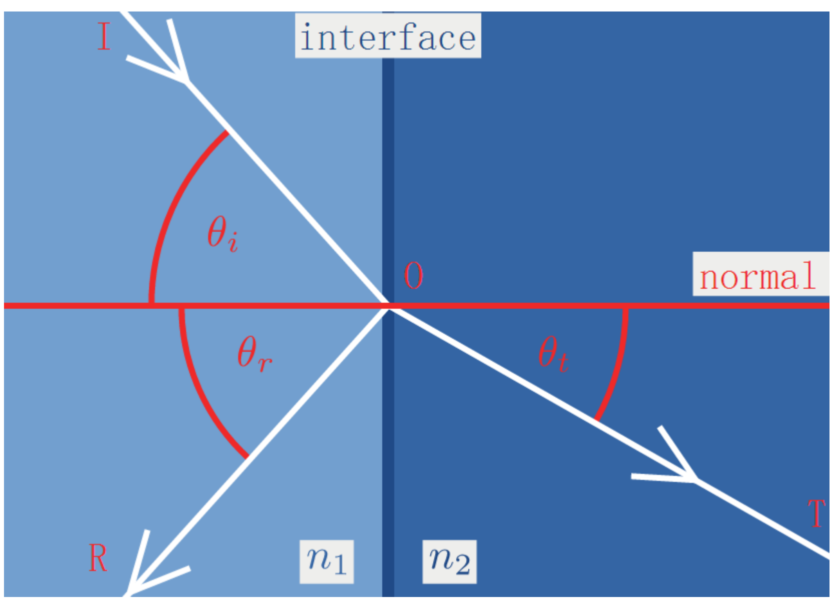

Figure 1. Variables used in BRDF calculation

The relationship between θ_i, θ_r and θ_t are given by the law of reflection(1) and Snell's law(2), where n_1 and n_2 are the refractive indices of two media;

$$\theta_i = \theta_r \quad (1)$$

$$n_1 sin\theta_i = n_2 sin\theta_t \quad (2)$$

Then the fraction of the incident power that is reflected from the interface is given by the reflectance or reflectivity, R, and the fraction that is refracted is given by the transmittance or transmissivity, T, by Fresnel Equations:

$$\begin{aligned} R_s &= \left| \frac{n_1 cos\theta_i - n_2 cos\theta_t}{n_1 cos\theta_i + n_2 cos\theta_t} \right|^2 \\ R_p &= \left| \frac{n_1 cos\theta_t - n_2 cos\theta_i}{n_1 cos\theta_t + n_2 cos\theta_i} \right|^2 \\ R &= \frac{R_s + R_p}{2} \\ T &= 1 - R \end{aligned}$$

where R_s and R_p are the reflectance for s-polarized and p-polarized light.

3.4. Progressive Method

As the memory size is limited for the computer, we need a progressive progress to increasing the final result without storing corresponding number of photons. Since every photon is generated randomly, the rendering result is obtained by a stochastic progress. We enhance the output by averaging several results. In the averaging process, pixel with a higher difference from other results with other result will be considered noise and ignored. The threshold is predefined in the config file.

3.5. Rendering Pass

We divide the irradiance calculation into two parts: the direct lighting and indirect lighting. In our project, both parts are approximated with two rendering strategies: global illumination model and photon rendering. In this case, we implement three different rendering algorithms. The first one uses the global illumination model for both direct and indirect lighting, the second one uses the photon rendering for both direct and indirect lighting, while the last one uses the global illumination model for direct lighting and the photon rendering for indirect lighting.

4. Results

4.1. Result of Direct lighting and Indirect Lighting

Figure 2, 3 and 4 display the result of direct lighting, indirect lighting and the combined final result in Global Illumination Model.

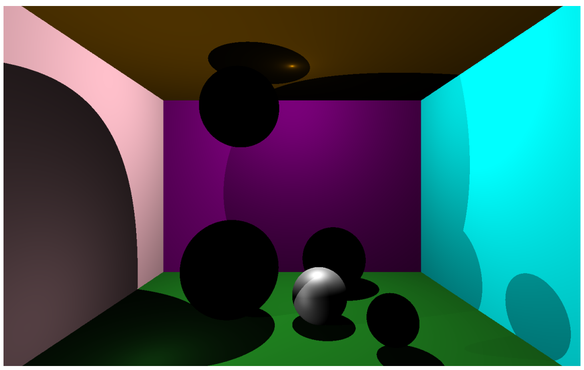

Figure 2. Direct Lighting in GI Model

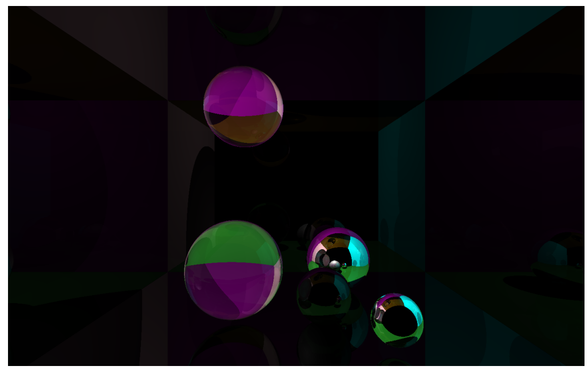

Figure 3. Indirect Lighting in GI Model

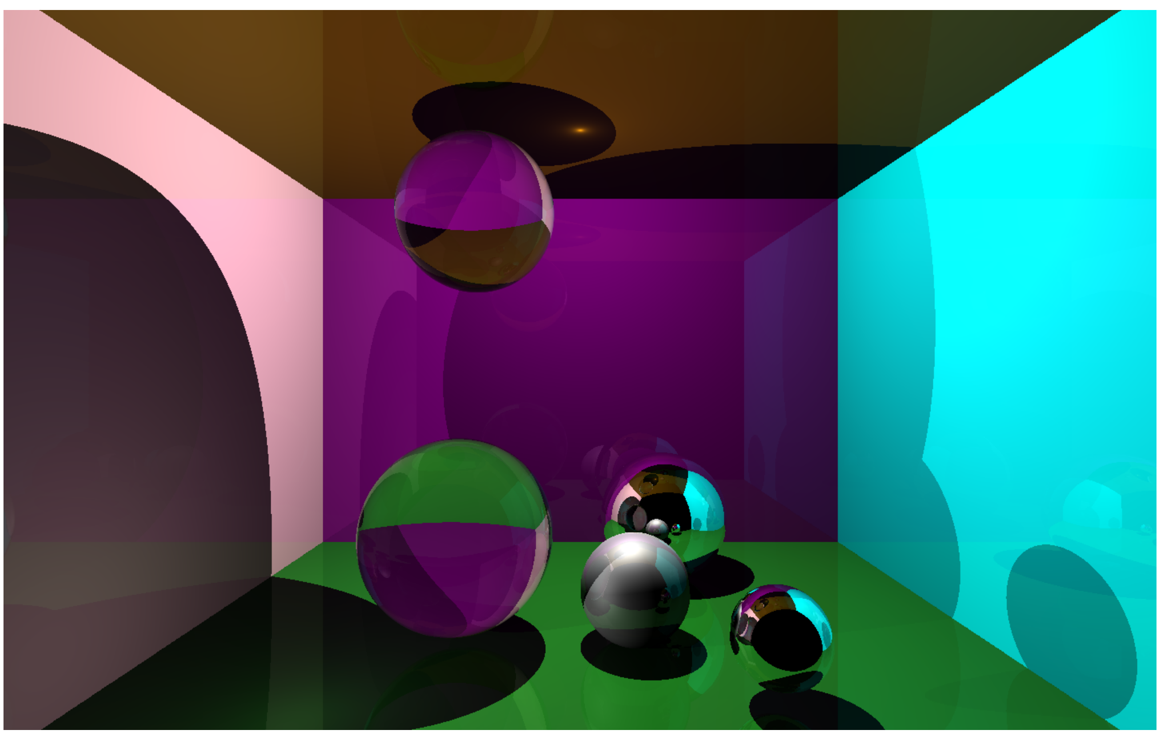

Figure 4. Final Result in Global Illumination Model

4.2. Result of Progressive Methodology

Figure 5 illustrates 4 stochastic results in a single rendering pass and the blending output obtained by 10 of them.

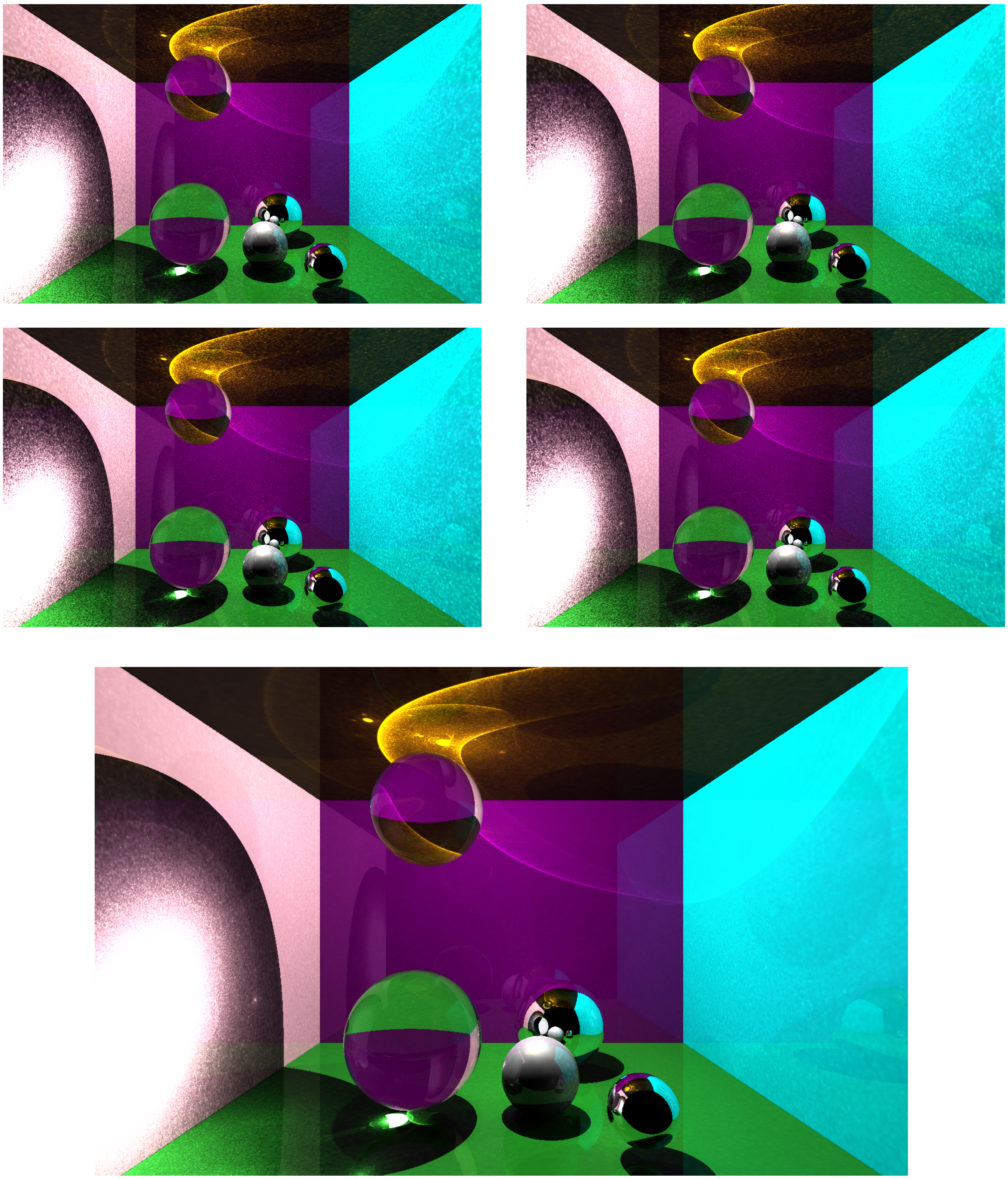

Figure 5. Progressive Result for Photon Mapping

4.3. Result of Multiple light source

Additionally, we implement the blending method for multiple light sources. Figure 6 shows the blending result of 2 point lights by Global Illumination Model (left) and Photon Rendering (right).

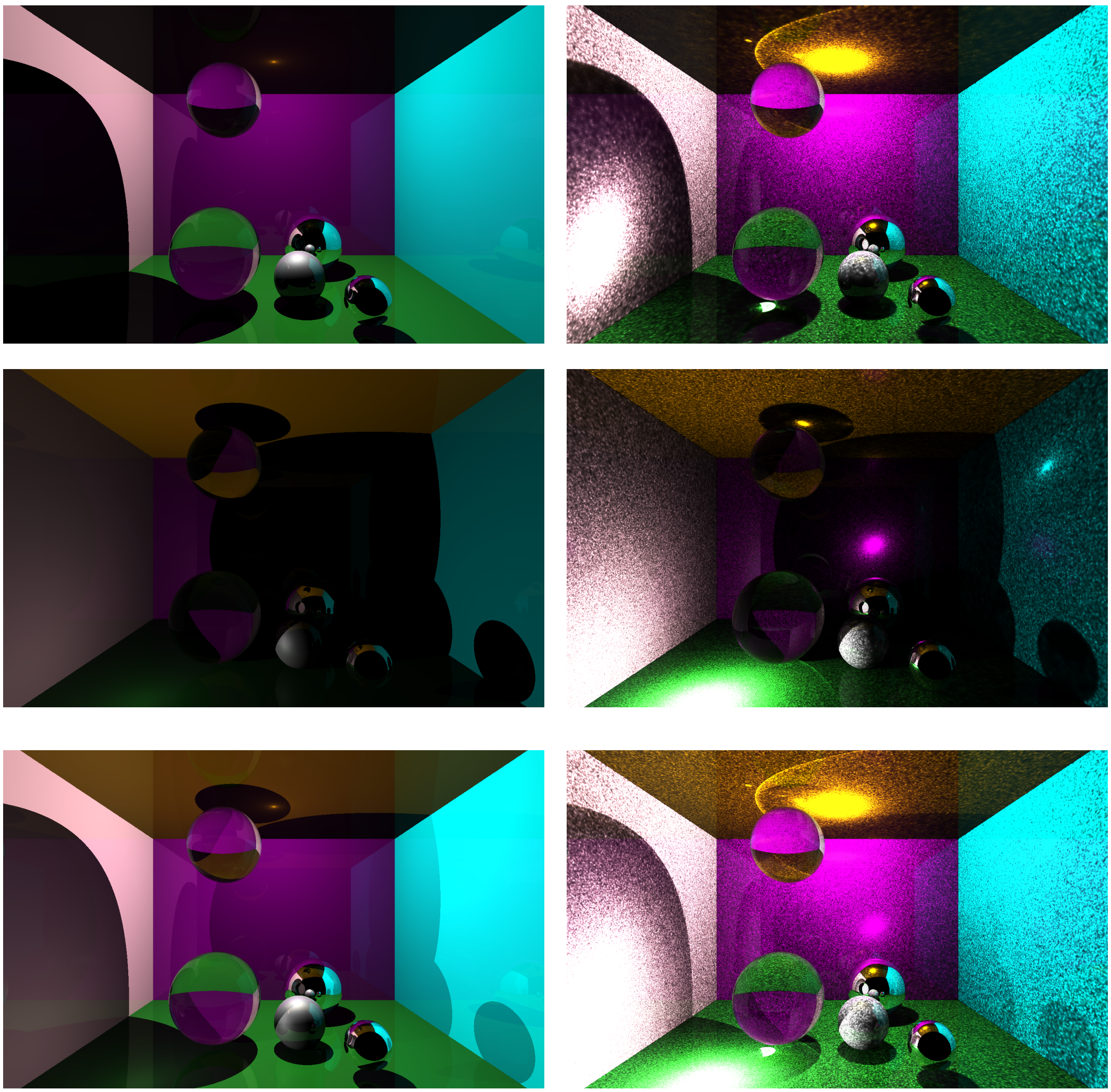

Figure 6. Multiple Point Lights, GI Model (left) and Photon Rendering (right)

4.4. Result of Three Rendering Pass

Finally, the comparison of three diffrent rendering pass is displayed in Figure 7. The top-left result is the result of pure Global Illumination Model and the top-right one is the output of pure Photon Rendering. The bottom image is the combination of these two rendering algorithms.

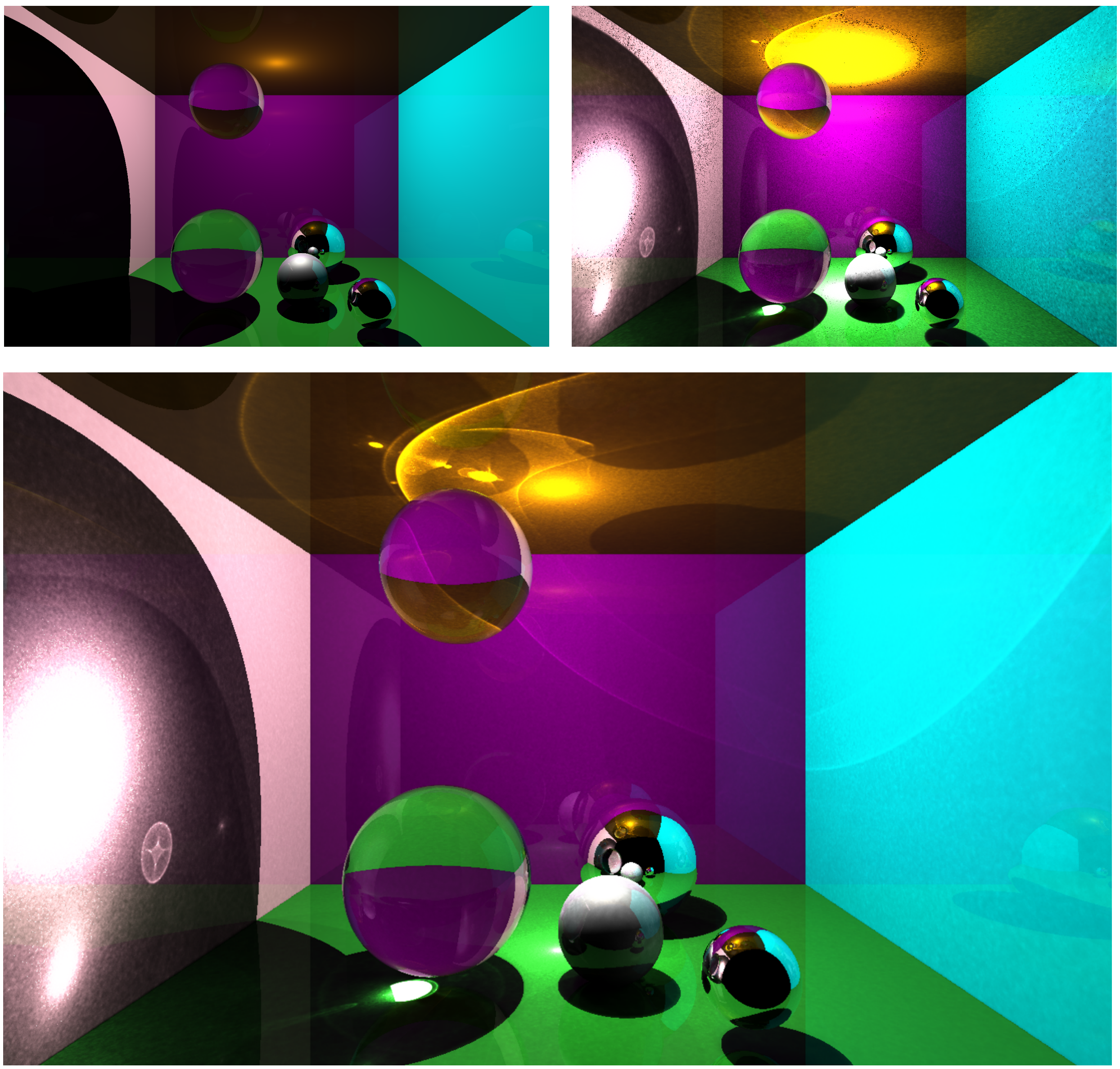

Figure 7. Comparison between different rendering pass

5. Conclusion

In the combination rendering methodology, we can increase the total photon number four to five times larger than the pure photon rendering methodology. The reason is that the direct lighting is approximated by global illumination model, therefore we do not need to store the photons which hit the surface directly from the light source. These photons are the main components in stored photons. Avoid of storing these photons, we can save more memory space for indirect lighting photons. These indirect lighting photons can produce better caustics effect, and because the global illumination model is a good approximation for direct lighting, it is a good combination of them.

6. Future Work

- Area Light and other light sources
- Box and more primitive objects
- .obj file support
- Depth of field effect
- GPU acceleration

References

[1] H. W. Jensen and N. J. Christensen, "Photon maps in bidirectional Monte Carlo ray tracing of complex objects," *Computers & Graphics*, vol. 19, no. 2, pp. 215–224, 1995.

[2] H. W. Jensen, "Global illumination using photon maps," in *Rendering Techniques 96*. Springer, 1996, pp. 21–30.

[3] T. Hachisuka, S. Ogaki, and H. W. Jensen, "Progressive photon mapping," *ACM Transactions on Graphics (TOG)*, vol. 27, no. 5, p. 130, 2008.

[4] C. Knaus and M. Zwicker, "Progressive photon mapping: A probabilistic approach," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 3, p. 25, 2011.

[5] H. W. Jensen, *Realistic image synthesis using photon mapping*. AK Peters, Ltd., 2001.