AVDGR Assignment 3

Saiyang Zhang 4664302

1 Introduction

In this assignment I tried to implement a working bidirectional path tracer, but only certain components are functioning. The components are as follows:

- Photon tracer.
- Kd-tree that stores the photon map, and the function to find the k-nearest photons within radius r.
- Radiance estimator (not functioning).
- Path tracer with NEE.
- Bidirectional path tracer(not functioning).

2 Implementation

Photon tracer

The photon structure is defined in Photon. The photons are generated from the surface of shapes with light material as in GetPhoton, the power of a photon is the color of the light, and the direction of a photon is sampled according to cosine weighted distribution, which is based on the assumption that light is emitted isotropically from the light source, and light emitted in a particular direction decreases as the angle between the surface normal and the direction of emission increases.

To generate a lot of photons, sampling from a light source will be repeated as in GeneratePhoton. After a photon will enter the photon tracer after its generation.

When tracing a photon as in PhotonTrace, photons that hit a mirror surface will be reflected, and photons that hit a glass surface will be reflected or refracted according to Fresnel's law. When hitting a diffuse surface, the photon will continue random walk if it passes the Russian Roulette. The power of the photons will be changed accordingly when hitting different surfaces.

The photons are stored in photons.

Kd-tree

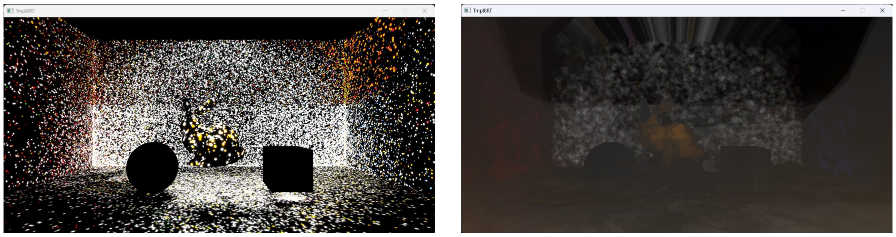
The indices of the photons will be stored in a Kd-tree according to their positions in photonMap. The Kd-tree node structure is defined in KDNode, it stores the index of the photon where it split, the axis of the split, and the indices of its left and right children.

The Kd-tree is built using BuildKDtree, the function is changed based on BuildBVH.

To estimate the radiance in a region, gathering the photons close to the region center is needed. Given an intersection point and an integer k, knnSearch will return the k nearest photons to the intersection points. It is implementd by maintaining the indices of the k nearest photons among the searched photons, and it will reduce the radius of searched region when the $k-$th nearest photon is updated.

Radiance estimator

The radiance estimator is defined in GetRadiance first gathers n nearest photons to the given intersection point, then it computes the flux in the region based on the power of the photons and the direction of the ray.



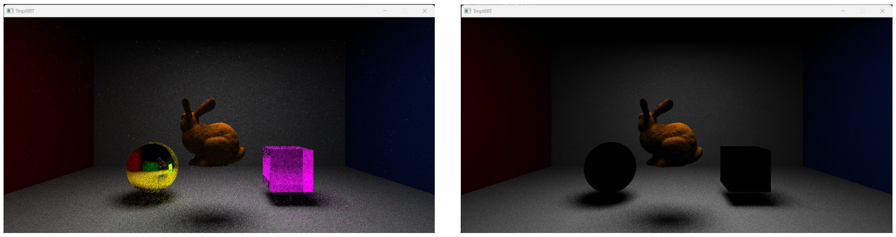
(a) With area introduced

(b) Without area introduced

Figure 1: Photons in the scene.

In invarea, the variable is not the area of the region of the n nearest photons. If the area is introduced, the photons in the scene will look like figure1(a). I read lots of materials but still do not know how to write the code. In random attempts to make the final result look better, I found not introducing the area will be better, the photons rendered this way look like figure1(b).

Path tracer with NEE



(a) Path tracer with NEE

(b) Direct illumination

Figure 2: Path tracer with NEE.

The path tracer with NEE is implemented in BDPT, if the code in Ld is uncommented and Ld is commented, the rendering result should be like figure3(a). This is a direct quote from the slides on the lecture. The direct illumination part looks like figure3(b), which is of the same use to the final rendering result as that of figure1(b).

Bidirectional path tracer

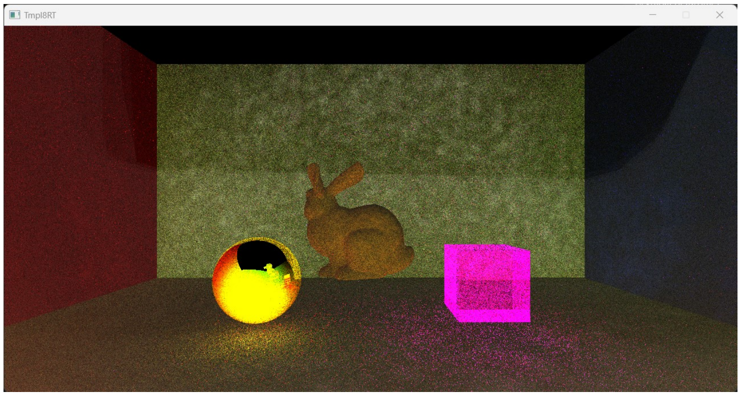


Figure 3: Bidirectional path tracer.

After reading the materials, I found that the original path tracer with NEE estimates direct illumination directly sampling the light source. By replacing that part with the photon mapping estimation, it is tracing many light paths with the information in the photon map. This would result in a more accurate representation of direct illumination compared to the original implementation, if it had a correct photon radiance estimation.

References

Henrik Wann Jensen. 2001. Realistic image synthesis using photon mapping. A. K. Peters, Ltd., USA.

Jensen, Henrik. (2001). A Practical Guide to Global Illumination using Photon Maps.

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. Physically Based Rendering: From Theory to Implementation (3rd ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

cs.wpi.edu/ emmanuel/courses/PhotonMapping

https://dezeming.top/photon_mapping

https://dezeming.top/BDPT