CPU-based Ray Tracing for Global Illumination

Teng-Sung Yu
Uppsala University
sodamax1740@gmail.com

Zhen-Tang Huang
Uppsala University
huang1473690@gmail.com

Tse-An Hsu
Uppsala University
jasonhsutsean@gmail.com

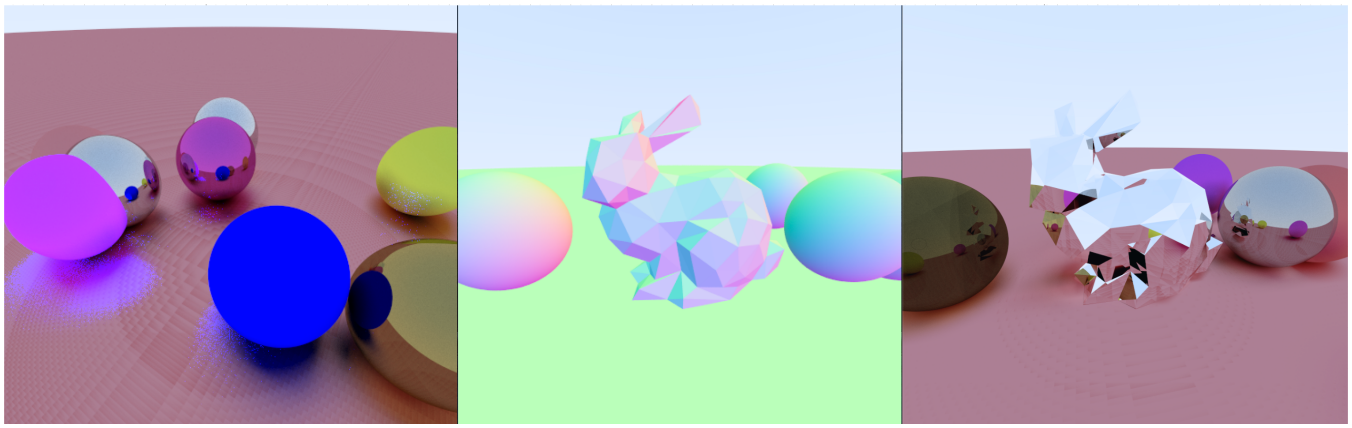

Figure 1. Results of ray tracer

1 Introduction

In computer graphics, ray tracing is a rendering method that is frequently used to produce photorealistic images. To accurately produce reflections, and shadows, it involves modelling the path of light as it interacts with the objects in a scene. A crucial aspect of ray tracing is global illumination, which is the accurate simulation of light interacting with all surfaces in a scene. In this project, we will be mainly based on the book "Ray Tracing in One Weekend".[1]

- Implement anti-aliasing.
- Implement rendering of diffuse surfaces.
- Add a few more spheres and metal surface types to our scene.
- Add an ImGUI window for controlling rendering parameters such as material properties, anti-aliasing, number of bounces, etc.
- Improve the speed of rendering low-polygon meshes.

2 Our Approach

2.1 Implement anti-aliasing

We implemented anti-aliasing by adding per-sample jitter to the camera rays.

2.2 Implement rendering of diffuse surfaces

First, pick a random point in the unit cube where x, y, and z all range from -1 to +1. Reject this point and try again if the point is outside the sphere. Then update the ray_color() function by using the new random direction generator plus normal vector as new ray direction. In this way, we could create ideal diffuse surface.

2.3 Add a few more spheres and more surface types to your scene

We implemented metal surface to the spheres. We added an abstract material class that allows us to control the surface material for each sphere. The method we implement metal surface is that we calculate the reflect ray direction by the formula of v + 2b. We also add a fuzz variable to control the roughness of the metal surface.

2.4 Add an ImGUI window for controlling rendering parameters

This task is relatively simple to implement. We have ImGUI window for controlling rendering parameters such as material properties(value of fuzz), anti-aliasing, and number of bounces. It could also display the surface normals and toggle gamma correction on/off. We create some corresponding variables to control the parameters.

2.5 Speed up rendering low-polygon meshes

We firstly calculate the bounding sphere to the low-polygon meshes. We add a bounding volume test to check the ray. We will determine whether the ray will intersect the bounding sphere or not. If the ray does intersect the volume, we will continue to calculate the ray hitting on the meshes. Otherwise, we could skip computing this ray. We could almost reduce half of rendering time by this approach comparing to the original ray tracer.

References

[1] Peter Shirley. 2020. *Ray Tracing in One Weekend.* https://raytracing.github.io/books/RayTracingInOneWeekend.html