Raytracing Report

s2337768

October 2025

1 Module 1

1.1 Blender exporter

I tested/debugged this task by inputting the results back into Blender and checking that it overlayed perfectly onto the original object. To help with this, my Export.py also converts the rotation to Euler degrees.

1.2 Camera space transformations

I tested/debugged this task by taking the output origin and direction for each ray, and writing a script for Blender that added a cylinder from the origin along the direction vector to visually check that it is aligned with the camera.

1.3 Image read and write

I tested this by first using my class to write a .ppm file with a gradient colour and visually checking that this worked (and that the dimensions were as expected). I then tested the read/alter method by reading this .ppm file and altering pixels to add two white lines to the file, which I checked visually.

2 Module 2

I changed the way the export python file identifies mesh types. Instead of using the name of the mesh, it uses a more robust method of counting polygons.

In fig 1 2 3 and 4 is the evidence of speedup for a one scene.

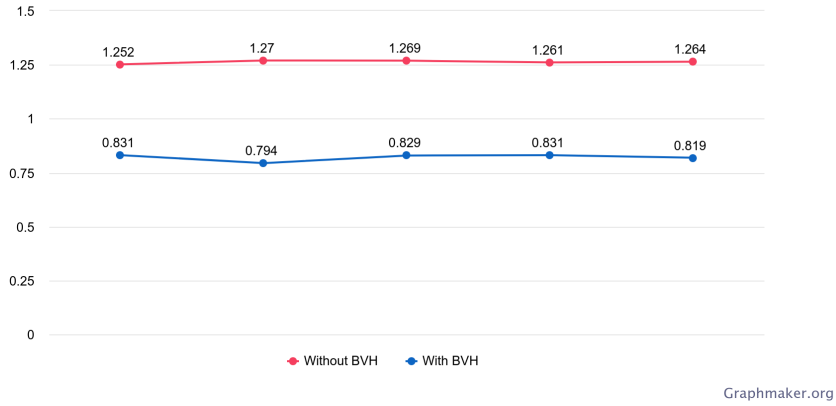

Figure 1: Graph comparing render time with BVH and without BVH

Without BVH	With BVH
1.252	0.831
1.270	0.794
1.269	0.829
1.261	0.831
1.264	0.819

Figure 2: Raw data

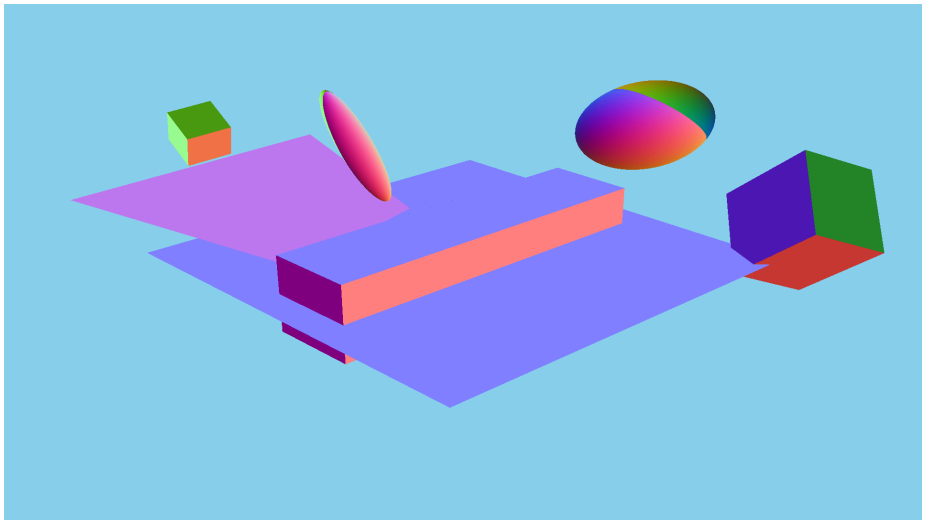

Figure 3: The rendered image

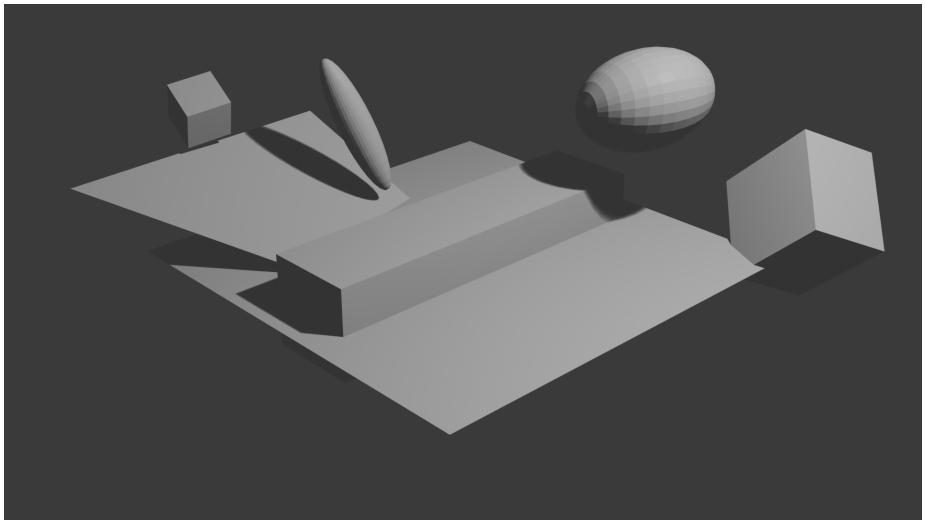

Figure 4: The Blender scene used