Introduction

Last week, we explored the topic of anti-aliasing by increasing the number of samples. However, you may have noticed that the ray tracer takes an unbelievably long time to run (the Cornell Box is only some 30 triangles). Part of the reason that was the case is the fact that to trace a ray into the scene, we did a ray-triangle intersection against every triangle in the scene. This is costly and unnecessary! In this homework, we will briefly explore the usage of uniform grids and bounding volume hierarchies to accelerate the ray tracer. This will allow us to shoot more rays into the scene and thus increase the number of samples we can take as well as perform refraction and reflection. As always, if you find a bug, please report it on Piazza.

Downloading the Code

In the writeup last week, I mentioned that you could fork the git repository and make it private. Turns out this was incorrect. It is now recommended just to clone the cs148raytracer-public repository. Sorry!

Assuming that you cloned the repository, you can just run:

git pull origin release

to download the new code!
Assignment

The only code that you have to modify to complete this homework is located within assignment6/Assignment6.cpp and all the tasks below should be run on the default scene. Note that you should find that the naive acceleration runs faster on a more complex scene. I spent some time optimizing the scene ray tracing this week so do not try to compare the numbers you got last week with those you get this week.

The code is setup so that you can specify two separate acceleration types: one for the entire scene and one for each scene object.

Part 1

In Assignment6.cpp, find where it says

```cpp
#define ACCELERATION_TYPE 1
```

When this macro is set to 0, the scene will use naive acceleration (it will check every scene object for an intersection). When the macro is set to 1, the scene will use a bounding volume hierarchy and when it is set to 2, the scene will use a uniform grid. Record the time it takes to run the scene using naive acceleration, a bounding volume hierarchy, and a uniform grid.

Part 2

From the previous part, you should see that the uniform grid runs slower than naive acceleration! The Cornell Box is a fairly simple scene so that is part of the reason why it runs slower (stepping through the voxels is slower than intersecting with the bounding boxes of the scene due to the sheer number of voxels). However, there are ways of alleviating this problem! Find where it says:

```cpp
accelerator->SetSuggestedGridSize(glm::ivec3(10, 10, 10));
```

This line sets the uniform grid dimensions to (10, 10, 10) which is fairly dense for this small and simple scene. Try changing the grid size to (5, 5, 5) and (3, 3, 3). Record the run-time of the ray tracer for all three options.

Grading

This assignment has the following requirements:

- Show a spreadsheet that contains the run-time from the tasks in parts 1 and 2 (be sure to clearly label what is what).
- Show the test scene rendered using one sample per pixel.

according to the following rubric:

- + – Meet the requirements and render your scanline image (provided that it is complex enough) or a similarly complex scene OR exceed the requirements via one or more technical contributions.
- ✓ – Meet the requirements.
- – – Render the test scene (aka show up to grading session with the program compiled).
- 0 – Do not show up to the grading session. :(

2