1. 总览

在本次课题中，首先我们将会实现基础的光线生成算法，具体而言是光线传输、光线与三角形求交。我们采用了这样的方法寻找光线与场景的交点：遍历场景中的所有物体，判断光线是否与它相交。

在场景中的物体数量不大时，这种方法能够取得很好的效果,但物体的数量增多，模型变得更加复杂时，该做法会变得很低效。因此，在完成基础算法以后，我们将会使用物体划分算法 Bounding Volume Hierarchy(BVH)来对算法进行改进。

我们已经提供了一个较为完整的代码框架。具体而言，本课题需要在代码框架上完成：

- **Render()**函数 in Renderer.cpp: 根据注释指引完成光线生成过程函数的编写。
- **Triangle::getIntersection()**函数 in Triangle.hpp: 在这个函数内部完成光线-三角形相交的过程。
- **IntersectP(const Ray& ray, const Vector3f& invDir, const std::array<int,3>& dirIsNeg)** 函数 in the Bounds3.hpp: 这个函数的作用是判断包围盒 BoundingBox 与光线是否相交，你需要按照相应的算法实现求交过程。
- **getIntersection(BVHBuildNode* node, const Ray ray)** 函数 in BVH.cpp: 建立 BVH 之后，我们可以用它加速求交过程。该过程递归进行，你将在其中调用你实现的 Bounds3::IntersectP

2. 开始实现

基础代码依赖于 CMake，为了避免编写过程过于冗长，我们推荐将认为分为**基础光线生成**以及 BVH **加速渲染**两部分完成。

```
$ mkdir build
$ cd ./build
$ cmake ..
$ make
```

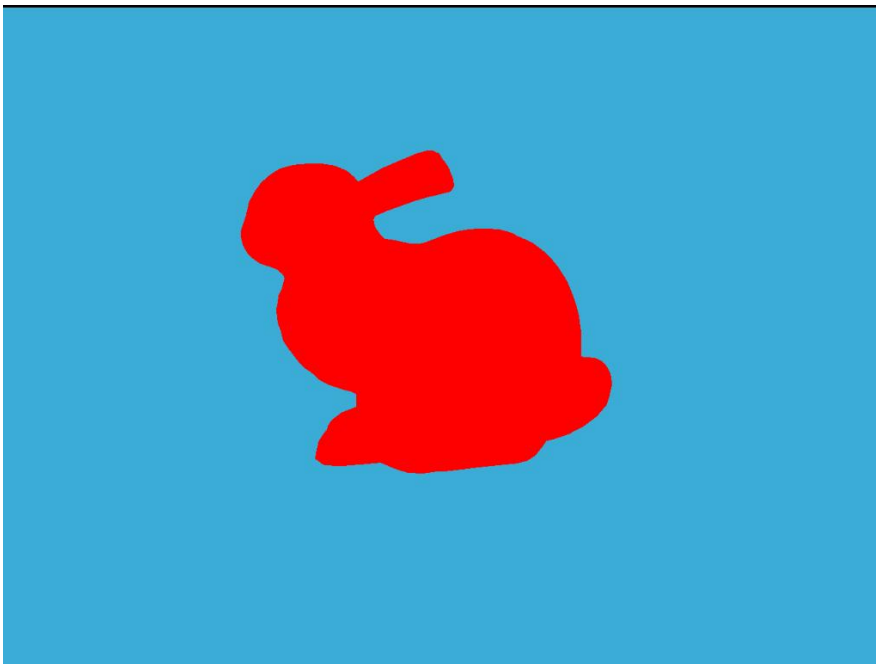
编译的过程与 CMake 项目编译过程类似

编译完成以后你可以通过**./RayTracing** 来执行你的程序

2.1. 基础光线生成

在这一步中，你需要完成 **Render()** 与 **Triangle::getIntersection()**函数。

当你完成这两个函数以后，你可以编译你的程序。然后通过**./RayTracing check** 来检验你的这部分代码是否得到正确的执



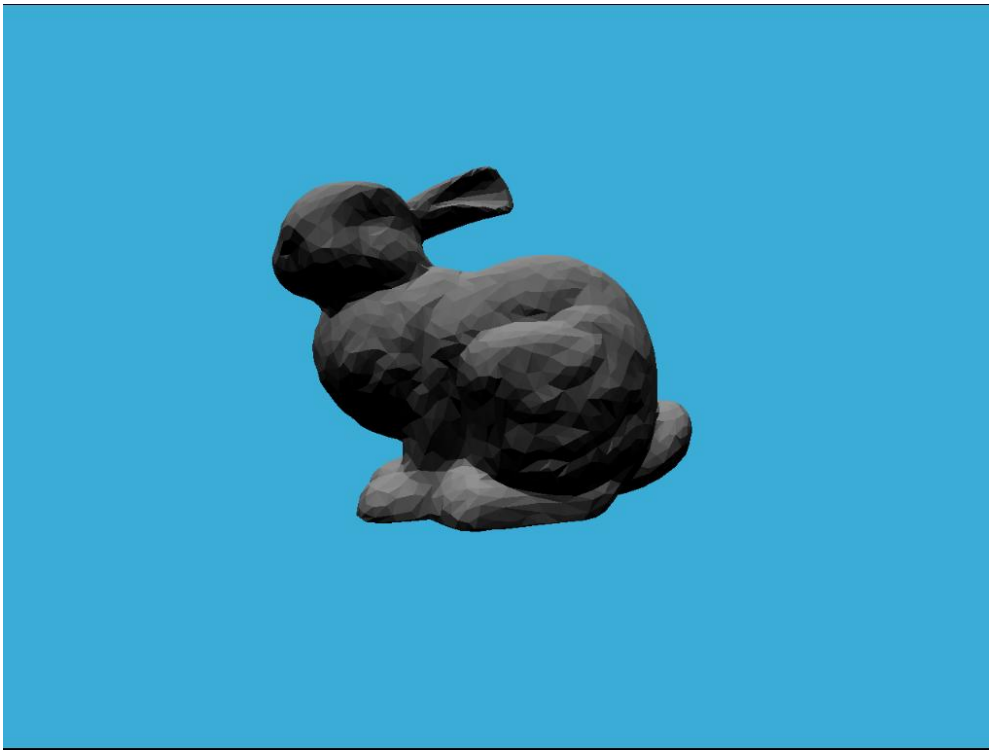
行。

如果你的代码正确，你会得到一个不完整的渲染图像。如图中所示。

在渲染过程中可以留意终端信息，未经过优化的渲染过程较长(参考值 7min，根据电脑性能存在差距。)

2.2.BVH 加速渲染

在这一步中，你需要完成 IntersectP(const Ray& ray, const Vector3f& invDir, const std::array<int,3>& dirIsNeg) 与 getIntersection(BVHBuildNode* node, const Ray ray)。当你正确完成了这两个函数，可以编译代码并执行./RayTracing。你应该可以以一个相当快的速度(参考值 2s)得到下面的渲染图像。



2.3.框架代码说明

仔细阅读框架代码能够帮助你更好的完成任务，代码中有以下内

容值得注意：

- **Material.hpp:** 这是一个单独的材质类文件
- **Intersection.hpp:** 这个数据结构包含了相交相关的信息。
- **Ray.hpp:** 光线类，包含一条光的源头、方向、传递时间 t 和范围 range
- **Bounds3.hpp:**包围盒类，每个包围盒可由 pMin 和 pMax 两点描述（请思考 为什么）。**Bounds3::Union** 函数的作用是将两个包围盒并成更大的包围盒。 与材质一样，场景中的每个物体实例都有自己的包围盒。
- **BVH.hpp:** BVH 加速类。场景 scene 拥有一个 **BVHAccel** 实例。从根节点开始，我们可以递归地从物体列表构造场景的 BVH.

2.4.额外内容

或许已经注意到，当我们完成基础光线生成的代码以后，我们并不能正确渲染出完整的着色模型。

这是因为为了提高渲染速度，代码框架中的的检测代码删除了着色模型相关的内容。这部分代码在 Scene.cpp 文件中的 **castray_noBVH** 函数，如果感兴趣，也可以在这里参照同一文件上方的 **castray** 函数，完成这一部分代码。

3. 提交

- 当你完成作业后， 请整理好项目文件，记得在你的文件夹中包含 CMakeLists.txt 和所有的程序文件 (无论是否修改)；
- 同时，请新建一个/images 目录，将所有实验结果图片保存在该

目录下；

- 再撰写一份实验文档简要描述你在各个函数中实现的功能，以及最后的结果；
- 最后，将上述内容打包，并用“姓名_学号_大作业.zip”的命名方式提交。