

光线追踪

曹伦郗 2020011020

项目架构

我基于PA1的框架构成了该项目，在`./include`和`./src`下新增了所需的源文件。

算法选型

我实现了PT算法，代码位于`./include/render.hpp`中。

`PathTracer::render`用于渲染，可通过运行时参数设定PT算法的参数，如同一像素点射出射线次数`samples`，以及迭代深度`depth`。

`ray_color`用于计算像素点颜色，并根据物体材质将光线作漫反射、反射和折射处理。

实现效果

层次包围盒（新增）

我在【验收后新增】了使用层次包围盒加速求交的功能，代码位于`./include/bvh.hpp`。

构建

主要思想是为一些种类的物体设计一个包围盒（主要包括球类、运动的球类、三角形类、网格类和参数曲面），将这些有包围盒的物体作为树的叶节点。

构建内部节点时：

1. 若只有1个物体，则左右子节点均为该物体；
2. 若有2个物体，则它们分别作为左右子节点，新节点的包围盒将它们的两个包围盒包围；
3. 若有更多物体，则将它们随机按某个坐标轴进行排序，分成左右两部分，再通过递归分别构建左右子节点，最后用一个包围盒将左右子节点的两个包围盒包围。

使用

层次包围盒的求交逻辑为：先同自己的包围盒求交，若有交再同左右子节点求交。

在`render`中，先用场景中带包围盒的物体们构建出层次包围盒，其余不带包围盒的物体保存在向量`obj_no_box`中，而在`ray_color`中求交时，光线先尝试同层次包围盒求交，再同其余不带包围盒的物体求交。

法向插值（新增）

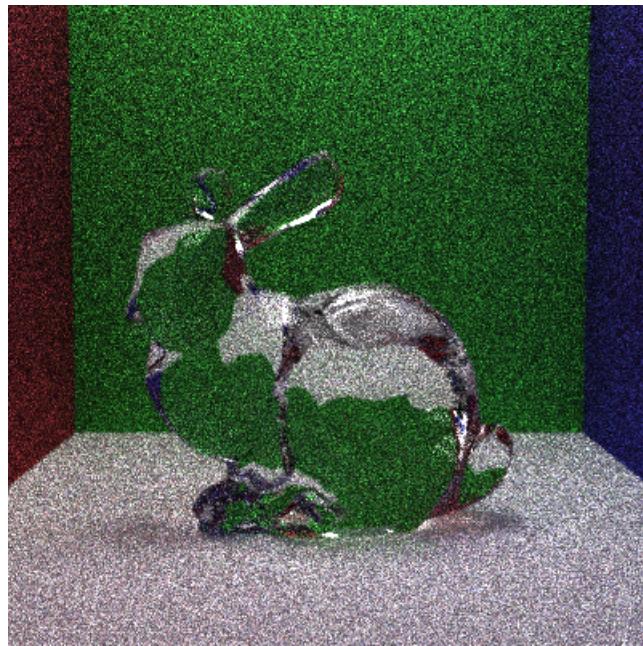
我在【验收后新增】了法向插值效果的实现，代码位于`./include/triangle.hpp`。

通过修改`./src/mesh.cpp`可以读入输入文件中各顶点的法向量，从而使三角形类得到三个顶点的法向量。

在求交时，以交点 P 和三个顶点 A, B, C 构成的三个小三角形的面积作为三个顶点的权重，例如顶点 A 的权重为三角形 $\triangle BCP$ 的面积，从而通过三个顶点的法向量加权得到最终交点处的法向量。



无法向插值



有法向插值

漫反射、折射、反射与阴影

这部分效果实现于 `./include/render.hpp` 下的 `ray_color` 中。

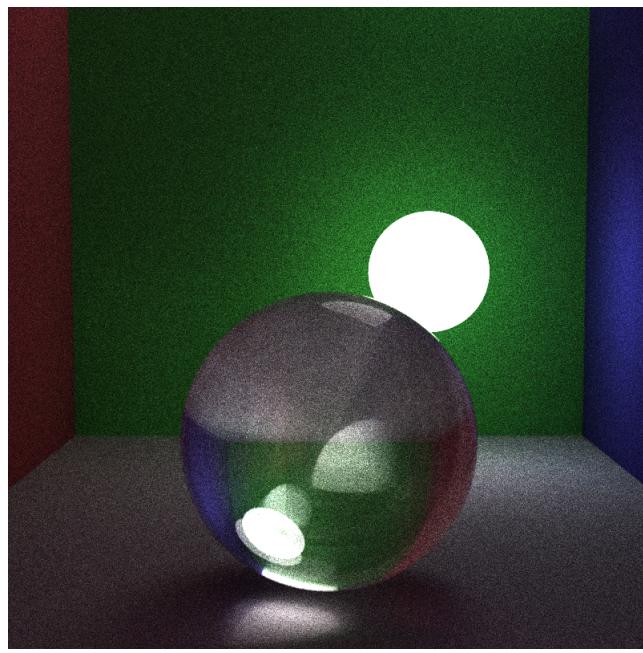
颜色计算

如果光线能与物体相交，则取一路上打到的物体的漫反射系数（漫反射时）或反射系数（反射时）的乘积作为衰减系数，与物体的发光强度相乘作为该物体对该像素点颜色的贡献值。

光线更新

首先要更新光线的起始点为发生相交点，再根据物体的材质更新光线的方向：

1. 漫反射时，取交点的法向，再加上一个模小于1的随机向量作为光线漫反射后的方向；
2. 反射时，根据入射和反射光以及法向量的几何关系，不难计算出反射光的方向；
3. 折射时，需要根据折射率进行实际计算，判断是全反射还是折射。

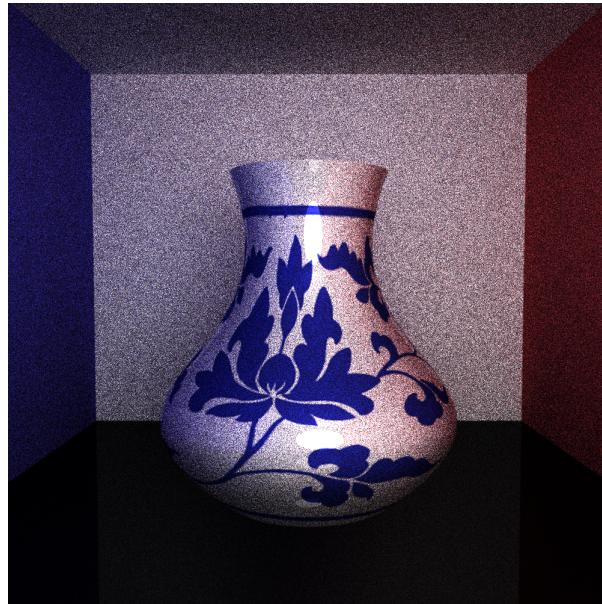


漫反射、反射与折射

参数曲面

参数曲面实现于 `./include/revsurface.hpp` 中。

主要是实现了牛顿法解析求交。通过先给参数曲面配一个包围盒，以光线同包围盒的交点作为迭代初值。



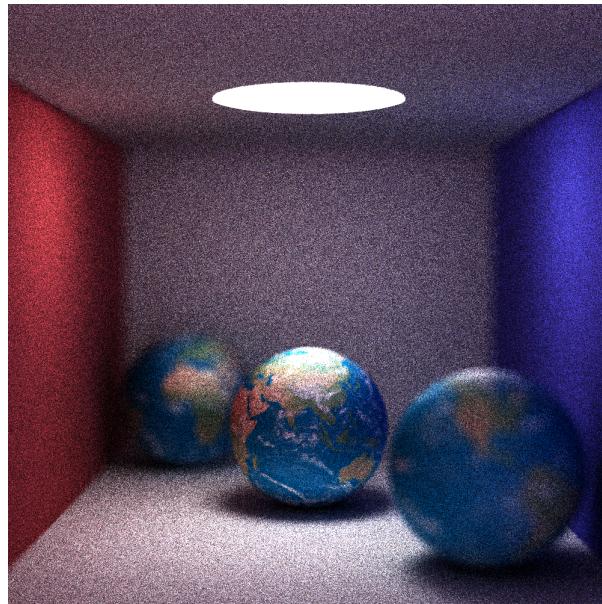
参数曲面

景深

景深效果实现于 `./include/camera.hpp` 中。

为相机类新增了光圈直径大小 `aperture` 和小孔上凸透镜的的焦距 `focalLen`，默认值分别为0和1，即相当于理想的直径无限小的小孔。

在相机生成射线时，需要在以相机中心为圆心，以光圈直径 `aperture` 为直径的圆上随机取值作为射线的起始点，相应的，射线的方向在x轴和y轴上也有不超过光圈直径的微扰。



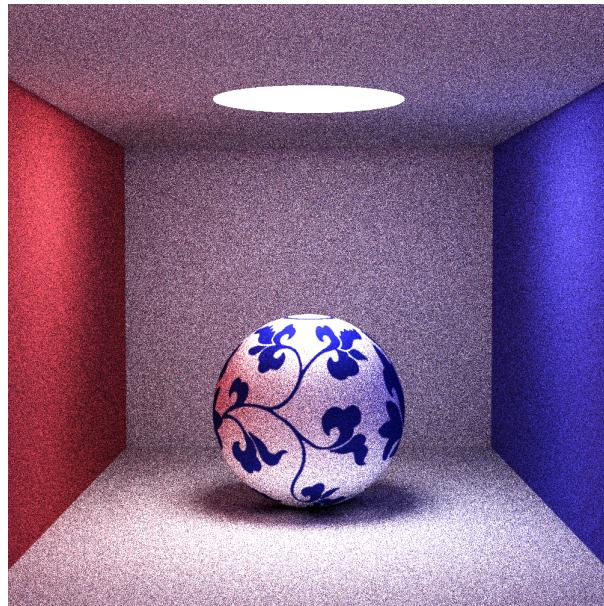
景深

纹理贴图

纹理贴图实现于 `./include/material.hpp` 中。

对于不带纹理的材质，获取材质的颜色时固定返回漫反射系数；而对于带纹理的材质，获取材质颜色时需给出交点对应的平面的 (u, v) 坐标，这就需要将三维曲面展开为平面。

故在交点类中新增成员 `color` 用于保存交点处的颜色，而对于球、参数曲面等物体，新增了根据交点求解对应的 (u, v) 坐标的函数。

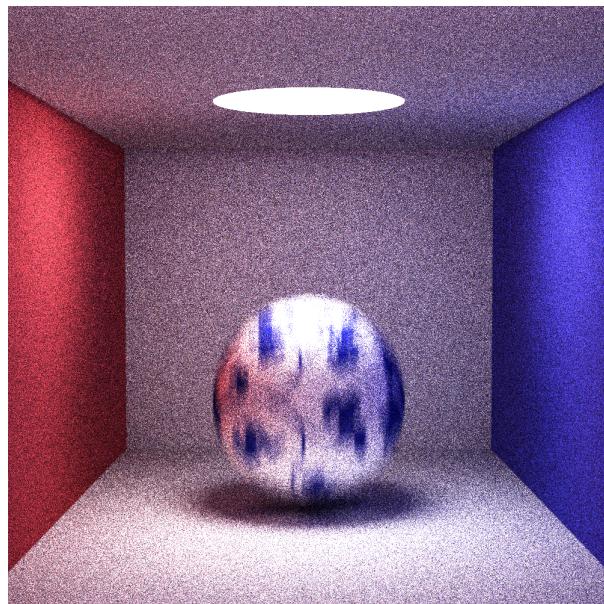


纹理贴图

运动模糊

运动模糊实现于 `./include/sphere_moving.hpp` 和 `./include/camera.hpp` 中。

运动的球类有两个球心，分别对应两个时刻 `time1` 和 `time2`，并实现了一个关于时间的一次插值函数，输出该时刻的球心。在求交时，光线会带有一个新增的成员 `time`，由相机类在生成射线时从相机的 `time1` 和 `time2` 之间随机给出，运动的球类就用插值函数得到该时刻的球心用于求交。



软阴影

PT算法渲染出的图片自带软阴影效果。具体效果从之前的图片中可以看出。

抗锯齿

抗锯齿实现于 `./include/render.hpp` 中。

PT算法对一个像素点的颜色值进行多次采样求平均值，故只需将该像素点的位置作微扰（我选择在两轴上都作 $[-1, 1]$ 的微扰），即以该像素点附近一小块区域内的平均颜色作为像素颜色，即实现了抗锯齿。

具体效果从之前的图片中可以看出。

最终效果图

