

图形学大作业：真实感渲染

范文祥 计82 2018011311

1. 运行指南

1. 由于本次作业采用的是PA1所提供的框架，因此在编译时可以直接参照PA1的编译方法，即在linux系统中在RayTracer文件夹下执行"bash ./run_all.sh"命令，之后在output文件夹下可以找到渲染成功的图片。
2. 运行时可以通过更改run_all.sh中的内容来实现想要的操作，具体参数意义在run_all.sh中有说明。
3. testcases文件夹下设有三个场景，场景的设计格式相较于原PA1的格式有部分改变，去除了光源和环境光模块，并对其余模块的属性做了与功能相对应的改变，在更改场景设计时需要注意。

2. 渲染结果

1. 图1: scene01_pt.txt即场景一所对应生成的图片，一个由六个平面组成的长方体空间，其内有一个反射球和一个折射球，还有一个球充当光源，只有一部分在长方体中。主要表现pt算法和软阴影，抗锯齿，贴图的功能，取样数为：4*5000，如下图：



2. 图2: scene02_mesh.txt即场景二所对应生成的图片，一个长方体空间和一个充当光源的球，还有一个斯坦福兔子的复杂网格模型。主要表现复杂网格模型的求交，取样数为：4*500，如下图：



3. 图3: scene03_depth.txt即场景三所对应生成的图片, 为与场景1相同的空间内放两个反射球。主要表现景深, 取样数为4*1000, 如下图:



3. 主要算法

1. 本次作业使用的是PA1的代码框架, 而算法则是主要借鉴了smallpt算法的逻辑。即通过对射出的光线经过反射、折射、漫反射处理后到达光源并返回的PathTracing算法来实现对图片的绘制, 并且为了实现算法以及其他功能对PA1的原框架进行了一些更改, 如删除lihgt类等。
2. 主要采取的是递归模式, 选择一个球当作光源, 光线与光源球相交时返回值为光源的发光颜色加上物体本身颜色与下一轮递归返回值的乘积, 否则直接返回本身颜色与下个结果的乘积。经过若干轮递归后若仍未到达光源球, 则返回(0,0,0)。
3. 每次光线与物体相交时, 根据物体表面的属性来决定是镜面反射还是折射、漫反射。若是镜面反射和折射, 则直接按照两者各自的物理逻辑算出对应的反射光线并开始下一轮递归, 若是漫反射, 则随机选择一个方向进行反射。若是两个或三个属性混合, 则以各自的比例为概率随机进行下一轮递归。每一个像素点分成4个小像素点, 并对这4个小像素点对应的区域随机射出光线, 并对每个像素点计算多次, 最后得到的结果取均值, 即可得到较好的效果模拟。若每个像素点计算samples次, 则最后的取样数为4*samples, 并且得到图片的效果好坏与取样数成正相关。
4. 计算折射时, 需要考虑是从光疏介质到光密介质还是从光密介质到光疏介质, 由于计算相交时只有一个表面属性, 必然是材料本身的折射率, 因此需要考虑到方向问题, 这里人为规定只发生物体与空气之间的折射。若是只有球体发生折射, 则可以通过球的特殊性质来判断方向, 但是为了使算法

更具有普遍性（虽然最后并没有用到），在递归函数中加了一个isAir参数，若参数为真，则表示光线在空气中，相交时光线从空气射入物体，反之从物体射出到空气，并根据是否发生折射控制isAir参数的值。

5. 主要算法对应的代码段为main.cpp中的PathTracing函数和main函数中的一部分。

4. 附加功能

1. 软阴影：smallpt算法中的光源取为球体，在场景中表现为一个弧面，由于是面光源，因此产生的阴影就是软阴影。没有特定的代码段，可以从main.cpp中的PathTracing函数中看出其逻辑。
2. 抗锯齿：也是smallpt算法的逻辑中自带的功能，通过对每个像素取四个小像素点，并对每个小像素点方块区域内随机射出光线，即可以防止相邻像素之间的差距过大，从而达到抗锯齿的效果。对应的代码段为main.cpp中的main函数中循环内的代码。
3. 贴图：实现了在平面中贴图的效果。使用PA1框架自带的Image类读入tga图片，并在读入场景时规定四个点，为平面中与图片四个角对应的点，在与Plane类的物体相交时，判断交点与四个点之间的位置关系，并得到图片上对应位置的颜色。为了实现这个功能，对hit类也做了一定的改变，加了颜色属性，存储相交时得到的颜色，并用一个bool变量来判断是否需要记录颜色。对应的代码段为plane.hpp中plane类的构造函数和intersect函数，并且hit.hpp也做了部分改变。
4. 景深：实现了景深的效果。通过将出射光线时的针孔换成一个有面积的圆来实现景深的效果，将圆上随机的采样点到原光线与焦平面的交点作为新的出射光线，从而达到景深的效果。对应的代码段为camera.hpp中的generateRay函数。
5. 网格模型求交加速：PA1的框架中原本的网格求交留有选做内容，因此较慢，将其改为正常的求交算法之后，使用包围球算法进行求交加速。求包围球的算法使用的是课件中提供的一个算法，分别找到x、y、z坐标最大和最小的三组点，取距离最大的一组点作为初始点，然后开始迭代，对每个在包围球外的点，扩大包围球，将其包含进去，最后得到的即为完全包含所有点的包围球。对应的代码段为mesh.cpp中的构造函数和intersect函数。
6. 样例叠加：由于在渲染时每次都要完全重新渲染，为了方便，设计了一种可以在之前得到的渲染图片上继续叠加样例数的功能，即每个像素渲染之前的颜色初始值设成读入图片对应点的颜色乘上原样例数再除以目标样例数，之后只需要渲染相差的次数即可以得到对应效果的图片。对应的代码段在main.cpp内的main函数中。