

学号：

姓名：

专业方向：

企业名称：

毕设课题名称：

企业指导教师：

院内代管教师：

**信息与软件工程学院**

**毕业设计（顶岗实习）**

**中期报告**

目 录

[第一章 概述 3](#_Toc479089669)

[1.1 综述 3](#_Toc479089670)

[1.2 课题背景与意义 3](#_Toc479089671)

[1.3 国内外研究情况 4](#_Toc479089672)

[第二章 概要设计 5](#_Toc479089673)

[2.1 渲染概述 5](#_Toc479089676)

[2.2 光线跟踪原理与设计 5](#_Toc479089677)

[2.2.1 光线原理 5](#_Toc479089678)

[2.2.2 光线发射与光线跟踪 6](#_Toc479089679)

[2.3 算法相关图形学概论 6](#_Toc479089680)

[2.4 优化数据结构设计 7](#_Toc479089681)

[第三章 详细设计 8](#_Toc479089682)

[3.1 数据结构设计与实现 8](#_Toc479089686)

[3.2 求交算法设计与实现 8](#_Toc479089687)

[3.3 光线跟踪算法流程设计 8](#_Toc479089688)

[3.4 优化结构设计 8](#_Toc479089689)

[第四章 开发难点与要点总结 8](#_Toc479089690)

[第五章 功能与性能测试 8](#_Toc479089691)

说明:

1.报告要求2000字以上。

2.本模板仅为基本参考，请各位同学根据个人情况进行目录结构扩展

1. 概述
   1. 综述

随着计算机硬件的发展，其对于图像的处理能力也越来越强大。计算机图形在我们的生活中也有了越来越多的应用。小到电影特技，游戏开发；大到灾难模拟，地理信息遥感，虚拟现实；这些产品中都有着计算机图形学的重要应用。人们对于计算机所绘制的图像的真实性，实时性上，要求也越来越高。如何快速，真实的渲染图像，是计算机图形学研究者们努力的目标。

在现代的计算机图形学中，主要有两种渲染方式，分别为光栅化和光线跟踪。

光栅化渲染是表示在计算机上显示三维形状的流行渲染算法。栅格化目前是生成实时三维计算机图形最流行的算法。此方法采用了流水线的设计思想，将三角形图元经过三维变化，投影变换，确定那些像素在顶点表示的图元内部，生成对应的坐标值，以及图元顶点的其他信息比如法线，颜色深度等等。然后根据深度信息确定遮挡关系。这种方法的好处是并行计算能力强，独立性高便于优化，能够立即响应用户输入，并且通常有每秒 24 帧的速率。与辐射着色、光线跟踪等其它渲染技术不同，栅格化的速度非常快，但是由于它不是根据光传输的物理规律进行处理的，所以无法正确模拟许多复杂真实光照环境。

比起光栅化的渲染方式，光线跟踪算法背后有着数学和物理模型的理论支持。其基本思想是根据光线传播的原理，模拟光线在实际场景中的传播过程：从光源发出的光线经过折射反射最终会到达我们眼睛，在计算机图形学中则是到达我们的屏幕；同时光路具有可逆性，所以我们可以从屏幕，追踪射出的光线，直到达到收敛条件为止。这种按照绝对物理规律计算出来的图形具有很高的真实性能，但是在计算的复杂度上，均要求很高。

* 1. 课题背景与意义

光栅化渲染的方式因为其计算的速度快，常用于计算机游戏软件；而光线跟踪的渲染方式因为其渲染图像的真实常常用于电影特效，CG动画等离线渲染的领域。不过随着时代的发展，人们对于画面的要求越来越高，为了满足人们的需求，光栅化渲染的方式为了追求更加真实的渲染效果，让设计变得越来越复杂，学习和使用成本也更加的高昂。

而光线跟踪算法其优势在于，按照真实物理规律构建渲染场景，随着硬件发展的今天，将光线跟踪的思想部分甚至全部应用于实时渲染，能够极大的提升渲染画质，也是人们研究的一个重点。

在光线跟踪中，主要的性能消耗在于求解光线与几何面片的相交问题上，优化几何面片的数据组织减少每次求交过程当中的冗余计算，能够极大的提升渲染速度。这篇论文中，将使用光线跟踪的方式设计并实现一个光线跟踪渲染器，并且引入多种优化方式，与传统渲染方式进行比较。

* 1. 国内外研究情况

近年来，在光线跟踪算法的研究方面各个机构都取得了不少的研究成果，在场景组织上，提出了空间网格结构，基于层次的结构的BVH，基于空间划分的BSP、八叉树、KD-Tree结构等等。

在数据优化方面，Hunt和Shevtsov等人提出了采用桶近似方法以达到快速计算SAH启发函数的目的。Ingo等人，通过场景面片进行分类，利用分而治之的思想实现了快速构建kd-tree的方法。SaarLand大学更是拿出了业界首个硬件加速光线追踪算法方案，这个被称作SaarCOR的光线追踪加速卡，采用FPGA(Field Programmable Gate Array现场可编程门阵列)堆砌，在66MHz的频率下就能获得和Pentium 4 2.5GHz相当的光线追踪性能。

在交互场景方面，Brigade开源引擎采用实时光线跟踪作为基础，在传统的光栅化基础上，引入光线追踪的的方式，表现力出色，其上有Arauna等游戏应用。Jonas Wagner则采用高效的Js引擎实现了实时光线跟踪的demo；Nvidia也有开发使用基于Nvidia GPU的高效光线追踪的引擎框架Optix。

1. 概要设计
3. 1. 渲染概述

渲染（render）在电脑绘图中，是指：用软件从模型生成图像的过程。模型是用语言或者数据结构进行严格定义的三维物体或虚拟场景的描述，它包括几何、视点、纹理、照明和阴影等信息。

渲染器是3D引擎的核心部分，它完成将3D物体绘制到屏幕上的任务。是具备根据发送过来的三维图形顶点信息，进行顶点变换，正交投影，纹理映射等过程绘制图片信息，并且添加反射，光照等额外信息的软件或硬件工具。并且根据运行的模式，常常分为软件渲染器和硬件渲染器：

硬件渲染器是将上述流程硬件化实现加速的渲染工具，我们称之为图形卡或者显卡，硬件渲染器通常基于底层图形API(应用程序接口)构建，采用适合硬件架构的光栅化方法进行渲染。图形API负责与硬件的通信，常用的图形API包括DirectX和OpenGL。根据硬件加速结构设计一整套的渲染管线，提供可编程借口。

软件渲染器则纯粹利用CPU的能力进行计算，通常采用光线追踪的方法进行渲染。软件渲染器没有统一的应用程序标准，但是有很多通用的渲染算法，如光子映射，蒙特卡洛，辐射度等等。

* 1. 光线跟踪原理与设计
     1. 光线原理

光是能量传播的一种方式，常见的光线从光源发出，并进行传播，直至遇到能够阻挡其传播的介质。光线在空间中沿直线传播，并且不需要其他介质作为传播媒介，在传播过程中会产生如下的物理现象：

反射现象，光在遇到不同介质时候会在介质表面产生入射光线返回原有介质的情形，其运动遵循如下三条定律：

反射线、入射线与法线在同一平面上。

反射线与入射线在法线的两侧。

反射角等于入射角

折射现象，光在不同密度的介质之间穿过时，会产生光路的偏转现象，不同的介质可以体现出不同的折射角，其计算由折射率公式决定：

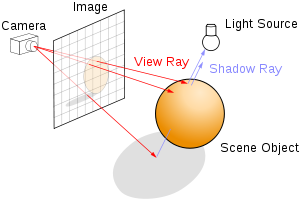


漫反射现象，由于物体的表面不是绝对的平整，所以上述两个过程在实际的材质上发生的时候，会在物品不同倾斜度的微表面进行，可以建模表示为照到物体表面的光线沿着四面八方产生漫反射。

* + 1. 光线发射与光线跟踪

光线发射算法最早由Arthur Appel提出，其主要思想为，从视点向屏幕的每个像素发射光线，照到光线传播路径上，照到最近的相交点，根据该点的物理性质，计算颜色的数值。

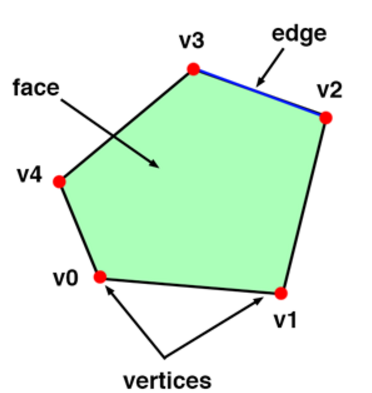
Turner Whitted优化了光线发射算法提出了光线跟踪算法，原有的光线发射算法在光线与第一最近的物体相交之后就结束了计算。Whitted 提出的算法，在光线与物体碰撞之后，会产生三种新的光线：折射光线，反射光线，阴影光线。其中反射光线和折射光线颜色之小于等于1（考虑到材质吸收）。弱平面为光滑平面，反射光线沿着镜面反射方向进行传播，最先相交的物体便是漫反射中看到的物体。折射光线在经过向量的变换之后，进行如下的计算，同样最先击中的物体就是半透明物体透过的物体。阴影光线用于检测当前点对于光源来说是否可见如果有不透明物体对光线遮挡，则后面的计算都不用进行跟踪计算。



上述方法的主要思想为利用光线计算的递归性，不断的进行跟踪直到达到收敛条件。其优点为基于真实的光的物理性质进行计算，对于反射和阴影等其他方式非常难以模拟的效果对于光线跟踪却是最基本的表示。并且每条光线之间相互独立，可以很简单的实现并行计算。不过其缺点也是十分明显的，就是因为要计算每一条光线导致算法的开销非常的大，很难用于实时的光线计算。

* 1. 算法相关图形学概论

计算机图形学是一种使用数学算法将二维或三维图形转化为计算机显示器的栅格形式的科学。简单地说，计算机图形学的主要研究内容就是研究如何在计算机中表示图形、以及利用计算机进行图形的计算、处理和显示的相关原理与算法。为了实现光线跟踪算法我们需要着重讨论实现几种对于三维世界进行建模的方法。

为了数学中为了描述点与点的关系，我们引入了坐标系的概念。一个平面内两条相互垂直的线段，作为X轴和Y轴，取垂直于XY平面的直线作为Z轴，根据右手或者左手定则规定规范坐标系的朝向。这样三维空间中任意的一个点我们都可以使用形如(X,Y,Z)这样地形式进行描述。线是点运动的轨迹，又是面运动的起点。线具有位置和斜率两个属性。直线绕着一点进行旋转形成面。对于三维空间中的物体，在计算机图形学中，根据上述最基本的几何概念，点线面，进行建模。在计算机图形学中，对于三维直接的物体，使用顶点表示对其外形采样后的表示；声明点与点的连接顺序构成平面，表示三维几何物体。

点的数目的多少决定三维建模物体多边形的精细程度，点的数目越多，则图形越精细，外形越接近真实的几何物体。同时，对于三维世界的物体来说，光至少能够在照射到其表面的时候产生反射，所以在建模的时候，还需要额外添加法线信息，来表示这个顶点的法线坐标，通常法线为原点到这个顶点的连线。在计算机图形领域，一个模型只需要包括最基本的顶点信息，顶点连接关系，就可以在计算机屏幕上绘制出来。

* 1. 优化数据结构设计

如上文所示，光线跟踪算法优化的一大重点，就是减少光线与场景中的几何物体求交点的次数。优化算法的基本思路为（回去补充）

1. 详细设计

4. 1. 数据结构设计与实现

如上文所描述，为了实现光线跟踪渲染算法，首先需要在软件中建立描述光线和点的数据结构。对于三维空间中的物体，在数学上，常用向量描述其点线以及面的关系，向量有方向和大小，可以进行加减运算，可以进行点乘和叉乘。所以我们用如下的结构进行表示：

要在屏幕上显示图像，需要先定义图像的颜色结构。在现代物理学中，光可以被建模为三种光子组成。有红绿蓝三种通道，可以进行加法减法，乘法运算，并且每个通道运算可以单独处理，互相不干涉，所以我们可以进行如下结构的建模。

光线需要在三维空间中进行传播，拥有方向和光强两个属性，其中光强直观的展示为光线的明亮度与颜色。所以我们使用向量和颜色对于光线进行描述：

* 1. 求交算法设计与实现
  2. 光线跟踪算法流程设计
  3. 优化结构设计

1. 开发难点与要点总结
2. 功能与性能测试