学号	姓名	论文规范性（10）	问题分析与调研（30）	方案创新性（20）	实验结果分析与讨论（40）	结课论文总成绩（100）
21301095	何宇航	9	27	19	39	94

北京交通大學

BEIJING JIAOTONG UNIVERSITY

计算机图形学课程论文

静态光追渲染

学院 软件学院

专业 软件工程

学号 21301095

姓名 何宇航

2024 年 6 月 23 日

摘 要

光线追踪是一种高质量的图形渲染技术，通过模拟光线在真实世界中的传播路径，实现物理上精确的阴影、反射、折射和全局光照效果。本文详细介绍了光线追踪的基本原理，包括光照模型、光线表示方式及其与物体的相交求解，并给出了光线追踪的递归渲染算法。实验部分展示了在不同视角下渲染的水晶球图像，验证了光线追踪的效果。结果表明，光线追踪技术可以显著提升图像的真实性和视觉效果。

关键词： 光线追踪；图形渲染；光照模型；递归算法

Abstract

Ray tracing is a high-quality graphics rendering technology that achieves physically accurate shadows, reflections, refractions, and global illumination effects by simulating the propagation paths of light in the real world. This paper details the basic principles of ray tracing, including the lighting model, the representation of rays, and the solution for intersections between rays and objects, and presents a recursive rendering algorithm for ray tracing. The experimental section demonstrates the ray tracing effect through crystal ball images rendered from different perspectives. The results show that ray tracing can significantly enhance the realism and visual quality of images.

Keywords: Ray tracing; Graphics rendering; Lighting model; Recursive algorithm

目录

1 介绍 Introduction

光线追踪，简称光追，是一种渲染技术，是三维计算机图形学中的特殊渲染算法，追踪光线从来源开始照射到物体上，再由物体反射的光线“路径”，利用算法来模拟真实世界中的光线的物理特点，能够做到物理上精确的阴影、反射和折射以及全局光照。

光线追迹方法首先计算一条光线在被介质吸收，或者改变方向前，光线在介质中传播的距离，方向以及到达的新位置，然后从这个新的位置产生出一条新的光线，使用同样的处理方法，最终计算出一个完整的光线在介质中传播的路径。

光线追踪技术的历史可以追溯到 20 世纪 60 年代，最早由 Arthur Appel 在 1968 年提出 [1]。随着计算机性能的不断提升，光线追踪技术得到了广泛应用，尤其在电影特效、游戏和虚拟现实等领域中表现尤为突出。光线追踪可以生成高质量的视觉效果，但计算量较大，因此对硬件要求较高。

在计算机图形学中，光线追踪一般会经过几个步骤。首先从摄像机位置出发，通过像素平面生成视线，视为光线；然后对于每一条光线，检测它是否与场景中的物体相交，并找到最近的交点。如果没有交点，该像素的颜色为背景色；然后对各种光照进行计算，主要包含直射光、反射光、折射光；最后确定各个像素点的颜色。

目前光线追踪被广泛应用于游戏中，通常来说开启光追的游戏往往能提供更加真实的视觉表现。如图1.1所示，左边为未开启光追的效果，右边为开启光追的效果。开启光追后，水面的反光更能真实的反应现实周围的环境。

图 1.1 光追效果对比

2 相关工作 Related Work

光线追踪技术自提出以来，受到了广泛关注和研究。

1968 年，Arthur Appel[1] 首次提出了利用光线追踪进行阴影计算的方法，这是光线追踪技术的初步应用。1980 年，Turner Whitted 首次提出了递归光线追踪算法 [6]，使得光线追踪技术能够模拟更加复杂的光照效果，如反射和折射。

近年来，实时光线追踪技术的发展得到了显著推进。Distributed Ray Tracing（分布式光线追踪）[3] 介绍了分布式光线追踪技术，通过在光线追踪过程中引入随机性，改善了图像质量并减少了锯齿现象。Path Tracing（路径追踪）[4] 提出了双向路径追踪算法，通过在光线追踪中同时追踪光线的前向路径和反向路径，提高了全局光照效果的模拟精度。Photon Mapping（光子映射）[5] 则是一种基于光子采样的光线追踪扩展方法，能够有效模拟复杂的光照效果。

最近，实时光线追踪技术的发展得到了显著推进，特别是通过 NVIDIA 的 RTX 显卡，实现了在游戏中的实时光追效果。这些技术的出现使得光线追踪在实际应用中变得更加普及和实用。

3 方法 Method

3.1 光照

使用 Phong 光照 [2] 反射模型，光照的公式如下所示：

$$L = k_a I_a + k_d (I/r^2) \max(0, \mathrm{n} \bullet \mathrm{l}) + k_s (I/r^2) \max(0, \mathrm{n} \bullet \mathrm{h})^p \tag{3.1}$$

下面对上述公式进行分解，Phong 光照模型分为三个部分。

第一部分为环境光项，可以理解为当前光经过若干次反射之后传播到某一点所呈现的亮度。对于环境光照来说，一般假设所有点的环境光照都相同，而不依赖于观察的位置或者光源的位置。其表达式如下：

$$k_a I_a \tag{3.2}$$

其中 k_a 表示环境反射系数，I_a 表示环境光强度。

第二部分为漫反射项，光在某一个物体的表面时，会被均匀地反射到四周。一般来说，我们认为一个点从任何方向来看，其漫反射光的强度和颜色都是相同的。从该项公式也可以看出，其与视角所在的方向无关。同样可以看出，如果光线垂直照在表面，那么漫反射的光强度最强。如果光几乎和表面平行，那么漫反射的光强度最弱。其表达式如下：

$$k_d (I/r^2) \max(0, \mathbf{n} \bullet \mathbf{l}) \tag{3.3}$$

其中 k_d 表示漫反射系数，I/r^2 表示光到当前点的强度，$\mathbf{n}$ 表示法线，$\mathbf{l}$ 表示光线。

第三部分为高光项。如果光线的反射方向和我们观察方向极度接近时，此时会有直接的高强度的反射光（可能会表现为光滑、刺眼等视觉感官）。可以通过计算半程向量和法向量的夹角来计算光的强度。注意要加上一个指数，来控制衰减速度。其表达式如下：

$$k_s (I/r^2) \max(0, \mathbf{n} \bullet \mathbf{h})^p \tag{3.4}$$

其中 k_s 表示镜面反射系数，$\mathbf{h}$ 表示半程向量，p 为高光指数，其余定义与上式相同。

3.2 光线表示方式

首先将光线想象成一条射线，那么它将由光源和方向组成。具体来说，光线的公式化表达如下所示：

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d} \quad 0 \leq t < \infty \tag{3.5}$$

其中，t 表示光线沿方向行进的长度，$\mathbf{d}$ 表示光线的方向，$\mathbf{o}$ 表示光源位置。

3.3 光线与物体相交的求解

光线与隐式曲面的会有一些交点的产生，下面以图3.1为例，假设球体的表达式为：

$$\mathbf{p} : (\mathbf{p} - \mathbf{c})^2 - R^2 = 0 \tag{3.6}$$

直接将光线的表达式带入，可得

$$(\mathbf{o} + t\,\mathbf{d} - \mathbf{c})^2 - R^2 = 0 \tag{3.7}$$

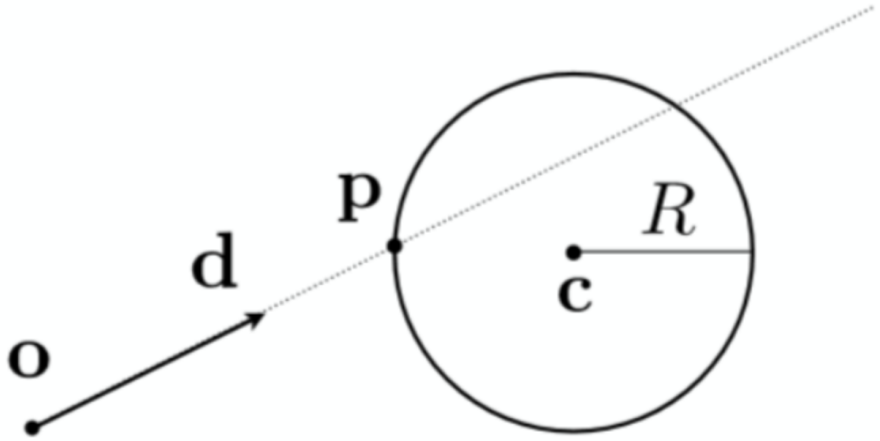

图 3.1 光线与球体的交点

可以根据判别根的正负性判断有几个交点。当然，对于更加一般的情况，我们也可以列出下面的通用方程组：

$$\begin{aligned} \mathbf{p} &: f(\mathbf{p}) = 0 \\ & f(\mathbf{o} + t\,\mathbf{d}) = 0 \end{aligned} \tag{3.8}$$

然而，在实际的图形学模型构建中，更多出现的是显示曲面，如图3.2所示。比如，我们常使用许多微小的三角形对曲面进行细分。而计算光线与显性曲面的交点实际上就是计算平面与直线的交点。

$$(\mathbf{p} - \mathbf{p}') \cdot \mathbf{N} = 0 \tag{3.9}$$

其中，$\mathbf{p}$ 表示交点，$\mathbf{p}'$ 表示平面上的另一点，$\mathbf{N}$ 表示平面的法向量。

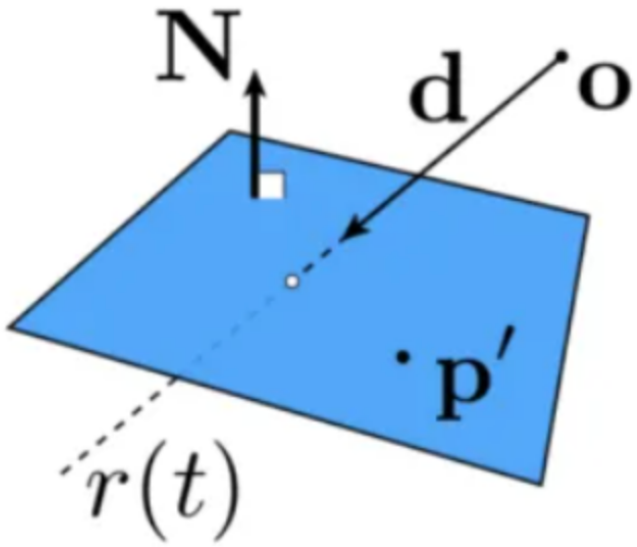

图 3.2 光线与平面的交点

3.4 光线追踪原理

首先需对光线的性质进行假设，假设光线满足以下三条物理性质：

1. 光线一定沿直线传播。
2. 光线之间无法碰撞。
3. 光线路径可逆。

基于上述假设，下面分步骤完成对光线追踪模型的构建。

第一步先进行光线投射。一条光线可能与多个物体相交，但是我们只需要考虑第一个相交的物体，如果一个像素点是第一个被光线相交的，那么认为该像素点被照亮，否则认为其处于阴影中。

第二步考虑光线的反射与折射。如果光线与可以造成反射、折射的物体相遇，那么光线将会对应的反射与折射。因此，当出现上述情况时，不仅仅是第一个相遇的像素点会被照亮，其他的像素点也可能出现不同程度的光亮。综上所述，一个像素点的颜色贡献可能来源于几种不同的类型：直射光照、反射方向间接光、折射方向间接光。

现在大部分光线追踪算法都采用递归渲染的方法，伪代码如下所示。首先输入为射线 (ray)，代表从观察点发出的光线。输出为反向光颜色，表示该射线在场景中的最终颜色。然后判断射线是否与任何对象相交，如果没有相交的就返回背景颜色，否则找到与射线最近的对象。然后分别计算反射和折射光线，获取对象的主颜色，然后递归计算反射和折射颜色，最终混合颜色获得最终的颜色。

Algorithm 1 Ray Tracing

Input: 射线 ray
Output: 反向光颜色
tracing(ray)
if no intersection with any object **then**
  return background color
else
  $obj \leftarrow$ find nearest object from the ray
  $reflect\_ray \leftarrow$ getReflectRay(obj)
  $refract\_ray \leftarrow$ getRefractRay(obj)
  $main\_color \leftarrow$ the radiance of obj
  $reflect\_color \leftarrow$ tracing($reflect\_ray$)
  $refract\_color \leftarrow$ tracing($refract\_ray$)
  return mix($main\_color$, $reflect\_color$, $refract\_color$)
end if

4 实验 Experiments

4.1 实验环境

静态光追渲染的实验在 macOS Sonoma 14.5 上进行。首先使用 g++ 完成对应用的编译，然后运行编译后的文件。待程序运行结束后，即可以查看渲染出的静态效果图。

4.2 实验步骤

实验分为以下几个步骤：

1. 场景构建：使用 OpenGL 创建包含多个物体和光源的三维场景。
2. 光线投射：从摄像机位置出发，通过像素平面生成视线，视为光线。
3. 光线-物体相交检测：对于每条光线，检测是否与场景中的物体相交，并找到最近的交点。
4. 光照计算：计算直射光、反射光和折射光的强度及颜色。
5. 递归渲染：采用递归算法计算光线的颜色贡献，直至光线能量低于设定阈值或达到最大递归深度。

4.3 实验结果

我们一共渲染了五张图像，分别代表了上下左右中五种不同的视角。通过不同视角可以看出水晶球内部的光追实现。

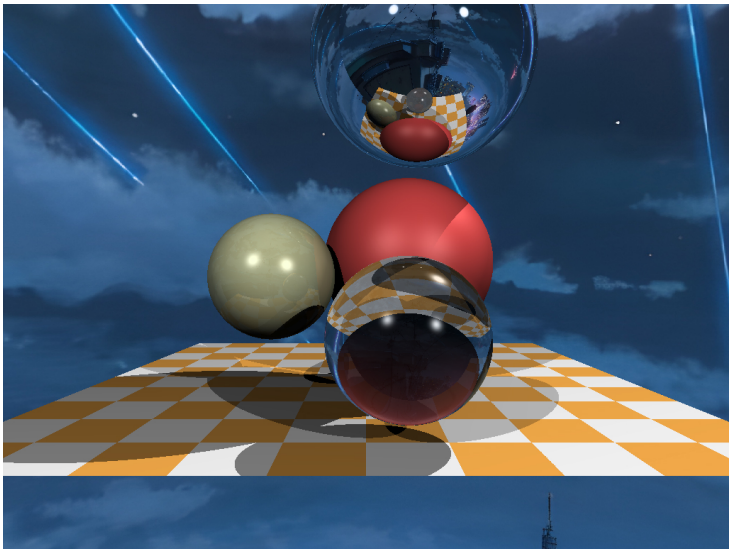

图 4.1 正视图

正视图展示了光线追踪在水晶球中心视角下的效果。可以清晰地看到水晶球表面的反射和内部的折射光线。光线在进入水晶球后，通过折射改变了方向，形成了真实感极强的视觉效果。水晶球表面的高光部分表现出光线追踪的镜面反射特性。

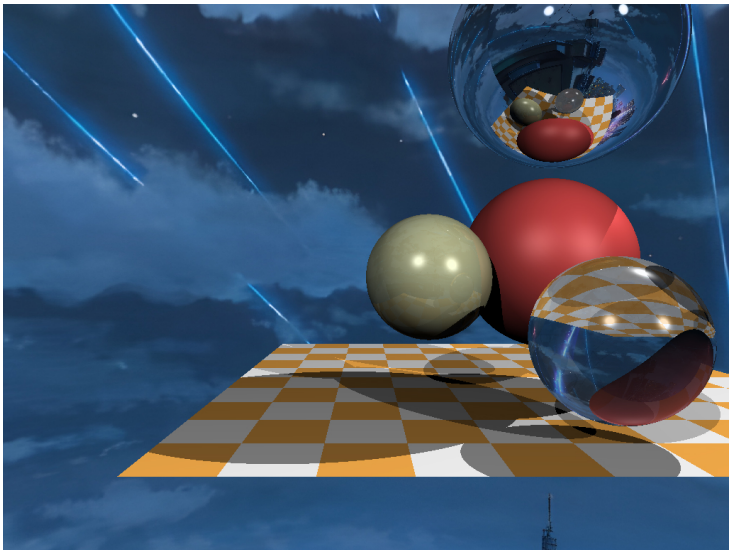

图 4.2 左移视图

左移视图从左侧角度观察水晶球，可以看到光线追踪技术如何处理不同角度的光线折射和反射。水晶球内部的光线路径变得更加明显，尤其是光线在水晶球内部多次折射后形成的复杂路径。

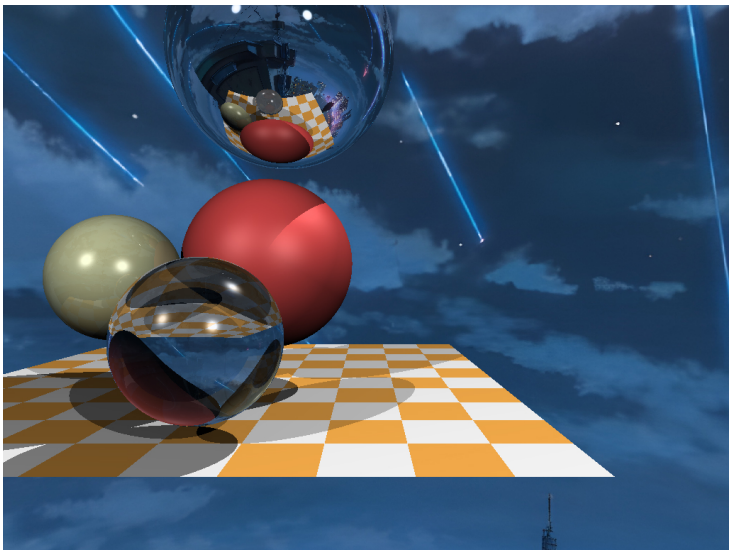

图 4.3 右移视图

右移视图与左移视图类似，但从右侧角度观察水晶球。可以看到不同角度的光线反射和折射效果。水晶球内部的光线路径清晰可见，展示了光线追踪技术在处理复杂光线路径时的准确性和细腻程度。

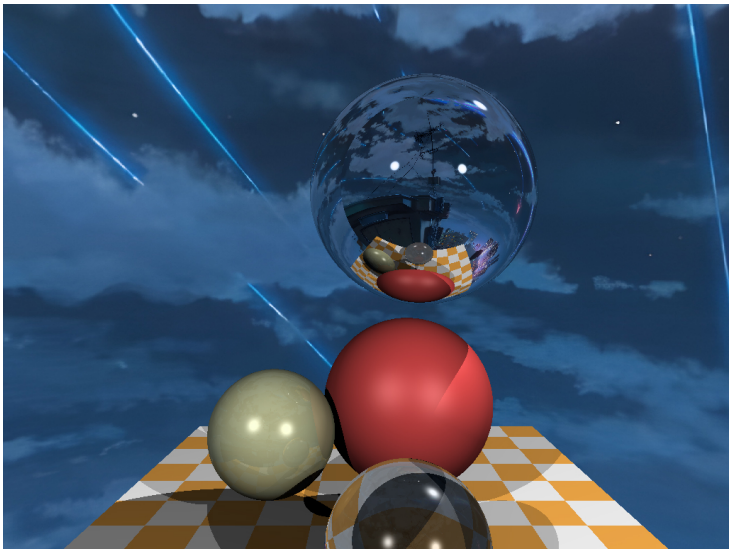

图 4.4 上移视图

上移视图从上方观察水晶球，展示了光线在通过水晶球顶部时的折射和反射效果。这种视角下，光线追踪技术的全局光照效果尤为突出。

图 4.5 下移视图

下移视图从下方观察水晶球，展示了光线在通过水晶球底部时的折射和反射效果。这种视角下，光线追踪技术在处理光线与物体交互时的能力得到了充分展示。

通过上述实验，我们可以得出光追技术有如下特点：

1. 真实感：光追能够精确模拟光线在不同介质中的传播路径，真实地再现阴影、反射和折射效果，显著提升了图像的真实性和视觉质量。

2. 视角变化对光线路径的影响：不同视角下，光线在水晶球内部的传播路径和相应的视觉效果有所不同，体现了光追在处理光线与物体交互时的准确性和灵活性。

3. 高光和阴影效果：光追能够准确模拟光线的镜面反射和漫反射效果，从而生成真实的高光和阴影。这在正视图和上移视图中表现尤为明显，体现其在光照计算方面的优势。

5 总结与未来工作 Conclusion and Future Work

在本文中，我们详细探讨了光线追踪技术在静态图形渲染中的应用。光线追踪是一种高质量的图形渲染技术，通过模拟光线在真实世界中的传播路径，实现物理上精确的阴影、反射、折射和全局光照效果。我们介绍了光线追踪的基本原理，包括光照模型、光线表示方式及其与物体的相交求解，并给出了光线追踪的递归渲染算法。在实验部分，我们通过渲染不同视角下的水晶球图像，验证了光线追踪技术的效果。实验结果显示，光线追踪技术可以显著提升图像的真实性和视觉质量。

尽管光线追踪技术在生成高质量图像方面表现出色，但是我们的程序只能实现静态图像的渲染，而无法达到实时渲染的效果。为了解决上述为问题，未来将尝试比如引入硬件加速技术，使用现代 GPU（图形处理单元）进行并行计算；或探索采用更高效的光追算法，如路径追踪算法，提高全局光照效果的模拟精度，减少计算量。

参考文献

[1] Arthur Appel. Some techniques for shading machine renderings of solids. In *Proceedings of the April 30–May 2, 1968, spring joint computer conference*, pages 37–45, 1968.

[2] Gary Bishop and David M Weimer. Fast phong shading. *ACM Siggraph Computer Graphics*, 20(4):103–106, 1986.

[3] Robert L Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 137–145, 1984.

[4] Eric P Lafortune and Yves D Willems. Bi-directional path tracing. 1993.

[5] Timothy J Purcell, Craig Donner, Mike Cammarano, Henrik Wann Jensen, and Pat Hanrahan. Photon mapping on programmable graphics hardware. In *ACM SIGGRAPH 2005 Courses*, pages 258–es. 2005.

[6] Turner Whitted. An improved illumination model for shaded display. In *Proceedings of the 6th annual conference on Computer graphics and interactive techniques*, page 14, 1979.