基于物理的水面渲染技术研究

123037910031 曹英凯

一、研究背景与目标

1. 研究背景

随着计算机图形学的发展，三维场景渲染已成为虚拟现实、电影制作和视频游戏等领域的重要组成部分。在这些应用中，对自然现象的高度逼真模拟成为提高用户体验的关键，其中，水面的渲染因其独特的视觉效果和复杂的光学性质而备受关注。然而，实时且真实的水面渲染一直是一个挑战，因为它涉及复杂的物理现象，如光的反射、折射、漫反射、菲涅尔效应、水下光散焦、波浪动力学等。

在硬件性能的提升方面，随着计算机硬件性能的提升，现代图形处理单元（GPU）能够处理更复杂的水面模拟和渲染算法，从而实现实时渲染；在应用领域扩展方面，不仅仅是游戏，实时水面渲染技术还在飞行模拟、虚拟现实、电影制作等领域有广泛应用。这些领域对高质量水面渲染的需求也在增长；在现实感的提高方面，随着计算机图形学和物理仿真的发展，人们对视觉和听觉的现实感要求越来越高，水面是其中一个关键要素；在环境交互性方面，虚拟环境中的水面通常是与玩家或用户互动的一部分。因此，实时水面渲染技术的研究也与用户交互体验有关。

总之，实时水面渲染技术的研究背景包括了游戏、虚拟仿真、图形质量的要求、硬件性能的提升、物理仿真、虚拟现实、电影制作以及科研进展。这些因素共同推动了实时水面渲染技术的研究和发展。

火焰模拟

烟雾模拟

2. 研究目标

当研究实时水面渲染技术时，主要目标在于实现逼真的水面效果，确保在有限的计算资源下实现实时性，支持用户互动，提供高性能，同时兼容多种计算平台。此外，研究也着眼于模拟水体的物理行为以保持真实性，同时可以用于增强各种应用的视觉效果，包括游戏、虚拟现实、电影制作和模拟环境。这些多样的目标要求结合计算机图形学、物理仿真、计算机视觉等多个领域的知识和技术，以满足跨领域的需求，实现水面渲染技术的不断进步和发展。

本研究旨在探索一种集成的方法，能够在实时场景中准确、高效地模拟和渲染水面，特别是考虑到与水相关的多种物理现象。我们的目标是建立一个灵活的框架，用于处理不同类型的水体，从小型水面（如水坑、小溪）到大型场景（如海洋、湖泊）。

二、问题分析与研究内容

1. 问题分析

实时水面渲染技术的研究课题涉及多个关键问题，需要深入分析和解决。首先，实时性是一个关键问题，因为在游戏、虚拟仿真和虚拟现实等应用中，水面效果必须以每秒多少帧的速度实时渲染，以提供流畅的用户体验。这要求开发高效的算法，以减少计算资源的需求。

其次，水面的逼真性是实时水面渲染技术的核心挑战。模拟水面的物理特性，如波浪、涟漪、反射和折射，需要复杂的物理模型和算法。研究人员需要考虑如何平衡逼真性和性能，以实现高质量的视觉效果。

最后，实时水面渲染技术的发展也需要考虑跨学科合作，因为这一领域涉及计算机图形学、物理仿真、计算机视觉和工程学等多个领域的知识。跨学科合作可以促进问题的综合解决，推动技术的不断改进。

水面渲染的主要挑战在于：复杂的水面建模，需要能够处理静态和动态水面，同时考虑波浪、涟漪、雨滴影响等因素；光学现象的模拟，涉及水的光学属性，如反射、折射、菲涅尔效应和水下散焦等；实时性能要求，对于视频游戏和虚拟现实应用，需要在保证视觉效果的同时，进行高效的实时渲染。

2. 研究内容

实时水面渲染技术的研究内容涵盖多个重要领域，旨在提供高质量的水面效果。首要内容是水面模拟，这包括了波浪的生成、水体的涟漪、光的反射和折射等物理特性的模拟。研究人员需要探索如何利用复杂的数学和物理模型来模拟水体行为，以实现逼真的视觉效果。与此同时，实时性是关键考虑因素，因为这些模拟必须在每秒多少帧的速度下进行，以提供流畅的用户体验。因此，研究包括了开发高效算法，以在计算资源有限的情况下实现水面的实时渲染。

另一个研究领域涉及到光照和阴影，因为水面的逼真表现需要考虑光线在水体内部和表面的传播方式。研究人员需要研究如何模拟不同光照条件下的水面反射和折射，以呈现出多样的视觉效果。

基于上述分析，本研究将集中在以下几个方面：

（1）水面建模技术：研究和比较不同的水面建模方法，包括基于物理的模型和几何建模方法。

（2）光照和着色模型：探索适用于水面的光照模型，包括阳光、天空光、水下光线等，以及它们如何影响水面的视觉属性。

（3）物理现象的集成：研究如何在单一框架内集成反射、折射、菲涅尔效应、水下散焦等现象。

（4）优化和实时渲染技术：发展针对实时应用的优化技术，包括硬件加速、着色器优化等。

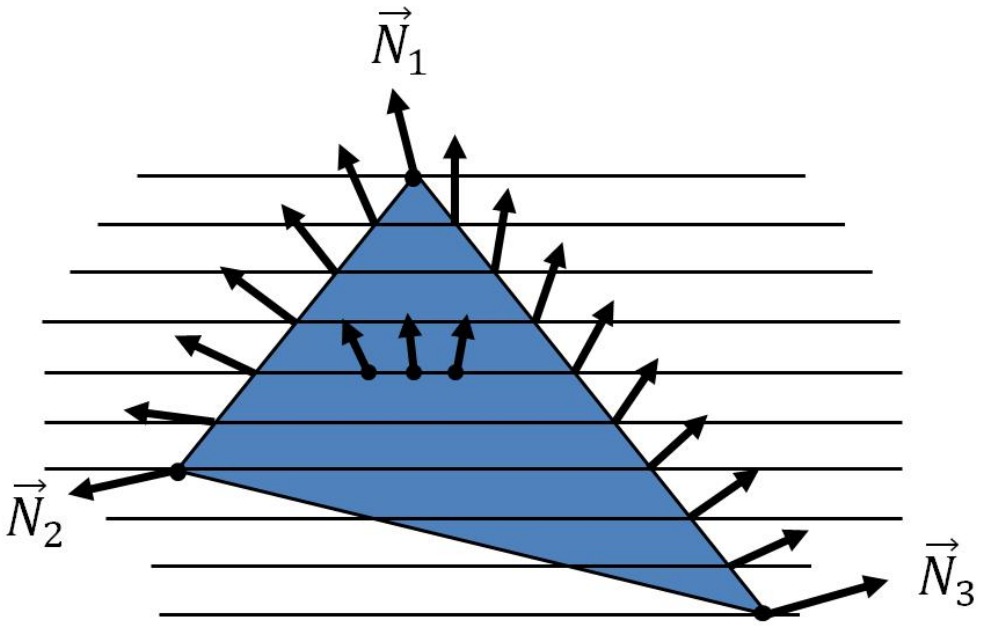
三、理论模型

1. 基础光照

（1）环境光照（Ambient Lighting）

概念: 模拟光源的间接效果，如光线从表面反射到环境中。

OpenGL 实现: 使用环境光照颜色和材质属性。

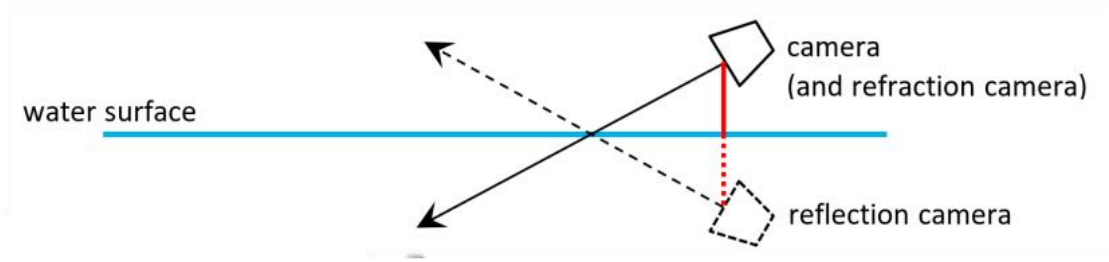


（2）漫反射光照（Diffuse Lighting）
概念：描述光线与表面法线（垂直于表面的线）的角度关系，实现更真实的光照效果。
OpenGL 实现：通过计算光源方向和表面法线的点积来实现。

（3）镜面反射光照（Specular Lighting）
概念：模拟光线在物体表面的高光效果。
OpenGL 实现：根据观察者位置，光源方向和表面法线计算高光。

2. 反射折射
（1）反射（Reflection）
概念：模拟光线击中表面后反弹的效果。
OpenGL 实现：使用环境映射（environment mapping）技术，例如立方体映射（cube mapping）。

（2）折射（Refraction）
概念：光线通过不同密度的介质（如水和空气）时，路径的改变。
OpenGL 实现：使用类似环境映射的技术，但需考虑介质的折射率。



3. 光线追踪
概念：一种高级渲染技术，通过模拟光线的路径来计算光照效果，可以产生非常真实的图像。

OpenGL 与光线追踪: 传统的 OpenGL 主要基于光栅化（rasterization），而不是光线追踪。但是，近年来，随着实时光线追踪技术的发展，一些高级 OpenGL 实现（如 OpenGL with RTX）开始支持光线追踪。
着色器（Shaders）: 在 OpenGL 中，顶点着色器和片元着色器是实现复杂光照模型的关键。
性能考虑: 高级光照模型和效果可能对性能有较大影响，特别是在实时渲染应用中。
实时渲染与离线渲染: OpenGL 通常用于实时渲染，这要求算法既要快速又要有效。

4. 水面流动
基本概念:模拟水面波动通常涉及到波形的生成，这可以通过多种方法实现，如正弦波函数、傅里叶变换或者粒子系统。
OpenGL 实现:
使用顶点着色器来修改水面顶点的位置，根据波形函数生成波纹效果。
通过纹理滚动和扭曲技术来模拟水流动的视觉效果。
实时波动效果可能需要结合时间变量来更新波纹状态。

5. 菲涅尔现象
基本概念:菲涅尔效应是描述光在不同角度入射到物体表面时反射强度变化的现象。通常在水面或其他光滑表面更为明显。
OpenGL 实现:
在片元着色器中，使用菲涅尔方程来计算基于视角和表面法线的反射率。
结合环境映射（如立方体映射）来实现反射效果。

6. 焦散
基本概念:焦散是指光通过透明或半透明物体，如水，被折射或反射并在另一个表面上形成明亮的光斑的现象。
OpenGL 实现:
一种方法是使用预先计算的焦散贴图，并在水下物体的着色器中应用这些贴图。更高级的实现可能涉及到光线追踪或实时模拟光线的折射路径，但这在实时渲染中可能非常计算密集。
性能优化: 这些效果可能需要大量计算资源，因此在实时应用中应注意性能优化。
着色器编程: 实现这些效果通常需要在 OpenGL 的着色器程序中编写复杂的 GLSL 代码。

物理精确性与视觉效果: 在实际应用中，可能需要在物理精确性和渲染效率之间找到平衡。
纹理和光照: 结合适当的纹理和光照模型可以显著提升水面和其他效果的真实感。

四、具体实现

1. 基础光照

（1）光源和材质属性定义：

光源属性：全局环境光(globalAmbient)，光源环境光(lightAmbient)，漫反射(lightDiffuse)，镜面反射(lightSpecular)定义在全局变量中。

材料属性：材料环境光(matAmb)，材料漫反射(matDif)，材料镜面反射(matSpe)，材料高光(matShi)也定义在全局变量中。

（2）光照定义：

installLights 函数：这个函数利用之前定义的光源和材料属性来设置着色器中的相应 uniform 变量。函数内部的 glProgramUniform4fv 和 glProgramUniform3fv 调用负责将这些属性传递给着色器。

2. 反射折射

（1）反射缓冲区设置：

代码创建了一个新的帧缓冲区(reflectFrameBuffer)并为其附加了一个纹理(reflectTextureId)，用于存储渲染的反射效果。

（2）折射缓冲区设置：

类似地，为折射效果创建了一个帧缓冲区(refractFrameBuffer)和纹理(refractTextureId)。

这两个缓冲区允许程序单独渲染反射和折射效果，而不直接渲染到主屏幕上。

3. 水面流动

（1）噪声纹理生成：

turbulence 函数：用于生成 3D 噪声纹理，这在 fillDataArray 函数中被调用，进而在

buildNoiseTexture 函数中创建真正的 3D 噪声纹理(noiseTexture)。

（2）动态波浪效果：
display 函数中，depthLookup 变量随着时间的推移而更新，模拟了波浪的动态变化。
display 函数中还处理了反射和折射帧缓冲区的渲染，以及最终的场景渲染。

（3）着色器应用：
在着色器中，噪声纹理和时间变量被用于动态计算水面顶点的位置和颜色，实现波浪效果。通过这种方式，代码结合了噪声纹理的不规则性和时间的动态变化来模拟水面波浪的自然流动，同时通过反射和折射效果增强了水面的真实感。整体上，本段代码展示了在 OpenGL 中实现复杂渲染效果的先进技术。

4. 菲涅尔现象（Fresnel Effect）
菲涅尔效应描述的是光波在不同介质界面上的反射率随入射角变化的现象，光线在不同角度下穿过材质边界（例如水面）时，反射和透射（折射）的比例会发生变化。在水面上，这意味着光线在与水面近乎垂直时主要透射，而在接近平行时主要反射。在水面这样的光滑表面，观察角度越平行，反射越强。
在 OpenGL 中，菲涅尔效应通常通过改变物体表面的反射率来模拟。具体实现通常涉及计算视线向量与表面法线向量之间的角度，然后根据这个角度调整反射率，菲涅尔效应是通过以下几个步骤实现的：
表面法线计算：代码中有法线向量的传递和处理，这是计算菲涅尔效应的基础。
角度计算：使用视线向量和法线向量计算角度。
反射率调整：根据计算出的角度，动态调整表面的反射率。
在本代码中，菲涅尔效应通过两个帧缓冲区（Frame Buffers）来实现：reflectFrameBuffer 和 refractFrameBuffer。reflectFrameBuffer 负责渲染水面上方的反射场景，而 refractFrameBuffer 用于渲染水面下方的透射（或折射）场景。

5. 焦散现象 (Caustics)
焦散现象指的是光通过波动的水面时产生的光斑和光纹。这在计算机图形学中通常通过特定的纹理或者算法来模拟。
直接使用噪声图，在片段着色器中生成一种焦散图案。噪声值在-1 到 1 之间循环，曲线在 0 附近坡度更陡峭，在±1 附近更平缓。使用可微调的变量 strength 作为指数求幂，进行调整，得到一个通常输出值接近 0，偶尔在小片区域输出值接近 1 的函数，将函数用于给定全局坐标处顶部水面的色彩值，它会产出随着顶部表面波浪弯曲的周期图案，然后就可以将其整合进底部表面色彩中。

```
if (isAbove != 1)
{     float causticColor = getCausticValue(tc.s, depthOffset, tc.t);
      float colorR = clamp(color.x + causticColor, 0.0, 1.0);
      float colorG = clamp(color.y + causticColor, 0.0, 1.0);
      float colorB = clamp(color.z + causticColor, 0.0, 1.0);
      color = vec4(colorR, colorG, colorB, 1.0);
}
```

五、实验分析方法与讨论

本实验使用 OpenGL、GLFW 以及其他图形库的 C++程序，目的是模拟水面的视觉效果。这个模拟包括光线反射、折射以及产生水面波动的 3D 噪声纹理。整个代码可以分为几个关键部分：初始化和设置、噪声生成、渲染水面以及渲染环境（如天空盒）。下面是对这些关键部分的详细分析。

（1）初始化和设置 (init 函数)

加载着色器：使用 Utils::createShaderProgram 函数来加载和编译顶点和片段着色器。这些着色器用于后续的渲染工作。

设置顶点数据（setupVertices 函数）：定义水面和天空盒的顶点数据，并存储到顶点缓冲区（VBO）。

创建反射和折射帧缓冲区：用于渲染水面上下的不同视图。

生成 3D 噪声：用于模拟水面波动的效果。

（2）3D 噪声生成

generateNoise 和 turbulence 函数用来生成 3D 噪声纹理。这种噪声纹理是水面波动效果的关键，通过随机数据产生的纹理模拟水面自然的波纹。

（3）渲染流程

渲染天空盒 (prepForSkyBoxRender 函数)：使用立方体贴图（Cube Map）来渲染周围环境。这为水面提供了一个自然的背景环境。

渲染水面 (prepForTopSurfaceRender 和 prepForFloorRender 函数)：这部分代码负责渲染水面的上层和下层。

为了模拟光线在水面上的行为，本代码采用了一种技术，它结合使用反射和折射帧缓冲区。这种方法不仅能精确地捕捉光线在水面上的反射和折射效果，还能通过之前生成的 3D 噪声纹理为水面添加逼真的波动效果。此外，使用了不同的着色器程序来分别处理水面的上层和下层，以更好地模拟光线在这两个不同环境中的交互。为了增强整体的视觉效果，还特别在照明和材质设置中加入了一个 installLights 函数，该函数为场景中的对象（如水面）设置了光照效果，包括全局环境光、漫反射光和镜面反射光，从而为整个场景带来更加真实和动态的光照效果。

（4）主循环 (main 函数)

这是程序的入口点，负责初始化 GLFW 窗口、GLEW，以及调用 init 函数来设置 OpenGL 环境。

主循环中调用 display 函数来不断渲染场景，实现动态的水面效果。

（5）关键 OpenGL 技术

帧缓冲区（Framebuffers）：用于离屏渲染，特别是在模拟水面的反射和折射时。

纹理（Textures）：3D 噪声纹理用于创建水面波纹效果，立方体贴图用于天空盒。

顶点缓冲区（Vertex Buffer Objects, VBOs）：存储顶点数据，用于渲染水面和天空盒。

着色器（Shaders）：顶点和片段着色器用于定义水面和天空的渲染效果。

本程序通过综合使用 OpenGL 的多种功能和技术，实现了一个相对真实的水面模拟。它通过 3D 噪声纹理创建波纹效果，同时利用反射和折射帧缓冲区模拟水面上的光线行为。此外，天空盒提供了自然的环境背景，增强了整体的视觉效果。

六、参数分析

1. 代码重要参数分析

（1）double smoothNoise(double zoom, double x1, double y1, double z1)

目的: 该函数用于平滑处理噪声数据。它通过插值的方式混合周围点的噪声值，以生成更平滑的结果。

参数:

zoom: 这个参数可能用于控制噪声的缩放级别。

x1, y1, z1: 这些参数是坐标点，用于指定在哪里计算噪声值。

（2）double turbulence(double x, double y, double z, double maxZoom)

目的: 这个函数用于创建更复杂、更自然的噪声图案，通常称为湍流。

参数:

x, y, z: 这些是坐标点，用于计算湍流噪声。

maxZoom: 控制噪声的最大缩放级别。

（3）void fillDataArray(GLubyte data[])

目的: 该函数用于填充一个数据数组，该数组可能代表纹理或图像数据。

参数:data[]: 一个字节数组，用于存储计算出的噪声值。

（4）GLuint buildNoiseTexture()

目的: 这个函数创建并返回一个 3D 纹理，该纹理基于生成的噪声数据。

返回值: 返回一个纹理 ID，用于在 OpenGL 中引用生成的噪声纹理。

（5）void generateNoise()

目的: 这个函数生成原始噪声数据，它填充了一个三维数组，该数组用于其他噪声处理函数。

2. 底面片段着色器分析

（1）输入和输出变量:

in vec3 varyingNormal;：输入的法线向量。

in vec3 varyingLightDir;：输入的光线方向。

in vec3 varyingVertPos;：输入的顶点位置。

in vec2 tc;：输入的纹理坐标。

out vec4 color;：输出的颜色。

（2）统一变量 (uniform):

sampler3D noiseTex;：3D 噪声纹理，用于计算波纹和光泽。

PositionalLight light; 和 Material material;：定义了光源和材料的属性。

mat4 mv_matrix, proj_matrix, norm_matrix;：模型视图、投影和法线变换矩阵。

int isAbove;：一个判断变量，可能用来检测对象是否在某个平面之上。

float depthOffset;：深度偏移量。

（3）函数定义:

estimateWaveNormal：估算波浪法线。
checkerboard：生成棋盘格纹理。
getCausticValue：计算水下的光斑效果（caustics）。

（4）主函数 ：
定义雾化效果的参数。
计算光照的各个成分（环境光、漫反射、镜面反射）。
通过棋盘格纹理和颜色混合生成最终颜色。
添加水下光斑效果和雾化效果。

3. 水面片段着色器分析
（1）输入和输出变量:
in vec3 varyingNormal, varyingLightDir, varyingVertPos;：输入的法线向量、光线方向和顶点位置。
in vec2 tc;：输入的纹理坐标。
in vec4 glp;：可能是用于计算反射和折射的投影坐标。
out vec4 color;：输出的颜色。

（2）纹理和结构体:
sampler2D reflectTex, refractTex;：反射和折射纹理。
sampler3D noiseTex;：3D 噪声纹理，用于计算波浪法线。
PositionalLight 和 Material 结构体：定义了光源和材料的属性。

（3）统一变量 (uniform):
mat4 mv_matrix, proj_matrix, norm_matrix;：模型视图、投影和法线变换矩阵。
int isAbove;：一个判断变量，可能用来检测对象是否在水面上方。
float depthOffset;：深度偏移量。

（4）主函数 main:
定义雾化效果的参数。
计算法线、反射向量和观察向量。
计算菲涅尔效应，用于模拟光线在不同角度下反射和折射的变化。
计算环境光、漫反射和镜面反射。
根据 isAbove 变量判断是处理水面上方还是水面以下的渲染。水面上方使用反射和折射纹理以及菲涅尔效应，水面以下则主要使用折射纹理和蓝色调。
最后，如果对象在水面以下，将颜色与雾化效果混合。

七、结论

本代码是一个利用 OpenGL、GLFW 和其他图形库实现的水面模拟项目。它通过结合多种图形渲染技术，如 3D 噪声、反射和折射效果，来创建一个动态的水面环境。

（1）优点

首先，使用了复杂的光照和材质处理，结合了环境光、漫反射光、和镜面反射光的多种光照参数，以及不同的材质属性，来增强视觉效果。接着，应用了 3D 噪声纹理技术，这种技术通过模拟噪声纹理来产生更真实和动态的水面波动效果。另外，使用了天空盒（Skybox）为场景提供环境背景，这不仅为场景添加了视觉深度，还增强了整体的沉浸感。最后，通过模拟水面的反射和折射效果，极大地提高了水面渲染的真实性。

（2）缺点

在波浪模拟和焦散模拟中，都是有规则有规律形成的纹理图案，没有使用 Perlin 噪声、分形噪声、高斯噪声，或者波浪叠加等方式，模拟不规律复杂的海面波浪。在场景方面，没有实现更加实时的变化空间，比如天气的变化，下雨场景，天空中物体的移动，水下物体的移动，等等，只局限于波浪本身和水面效果的模拟，比较单一，没有实现更加全面的整体水面场景。在光照模型，菲涅尔现象，焦散效果等光效上，也比较基础，有待改进。

（3）有待改进

在水的模拟方面，如波浪效果，涟漪效果，雨水碰撞效果等视效有待细化，让水面波浪可以高低起伏，不规律变化，更加符合真实场景，真实效果。在场景方面，可以有物体的移动，比如鸟、鱼或者岸边的树，对他们静止或运动过程中，在水面中的倒影，进行刻画，让场景更加丰富，更加体现对水模拟效果的展现。

八、参考文献

[1] 王渊.基于真实数据的逼真水面生成及动画研究[D].天津理工大学,2021.DOI:10.27360/d.cnki.gtlgy.2021.000632

[2] 张军鹏.基于FFT的海面模拟渲染方法[J].现代信息科技,2020,4(21):24-26+30.DOI:10.19850/j.cnki.2096-4706.2020.21.007

[3] 刘婵.真实感图形绘制的虚拟实验教学系统[D].中北大学,2021.DOI:10.27470/d.cnki.ghbgc.2021.000057

[4] 杨仲尧.基于粒子法的水模拟技术研究[D].天津大学,2020.DOI:10.27356/d.cnki.gtjdu.2020.003852

[5] 冯笑冰,朱登明,王兆其.基于深度图的水面重建[J].高技术通讯,2020,30(03):223-232.