# 水体渲染实现过程——4月14日到4月19日

## 概述

先介绍实现过程，这个过程分为两部分：顶点运动和表面着色。

再介绍在U3D里的实现。

## 顶点运动

参考这篇文章

[Chapter 1. Effective Water Simulation from Physical Models | NVIDIA Developer](https://developer.nvidia.com/gpugems/gpugems/part-i-natural-effects/chapter-1-effective-water-simulation-physical-models)

周期波的基础概念表示

L:波长

w:频率 w=2/L

A:振幅

S:速度 另一种表示法：φ=S·w

D:方向，水平向量，沿着波峰移动方向

我选用的是直线方向波，其中顶点高度公式为

Image

顶点法线公式推导：

任意时刻t点P的位置为

P(x, y, t)=(x, y, H)

副法线B是P点在x方向上的偏导数

B(x, y)=(1, 0, dH/dx)

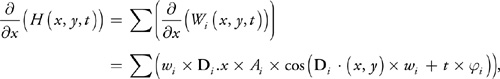
切线T是点P在y方向上的偏导数

T(x, y)=(0, 1, dH/dy)

法线N=cross(B, T)

=(-B.z, -T.z, 1) 并不是竖直向上的！

dH/dx:



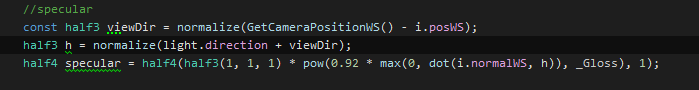
## 表面着色

水体表面着色可以根据需求制定多种方案，我为了最大程度模拟真实感，选择了

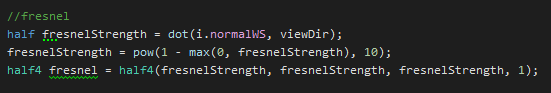
镜面反射+菲涅尔反射+折射+本体色+泡沫色的方案

由于对折射不太熟悉，花了较多时间，暂时只实现镜面反射+菲涅尔+折射，本体色应该做成根据Zbuffer做颜色插值，时间关系，暂时用临时颜色。

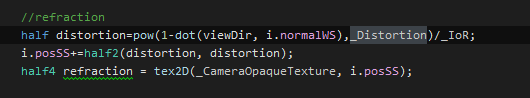
镜面反射：Blinn-Phong模型



菲涅尔反射：近似公式float fresnel = F\_base + F\_scale\*((1-v\*n)^power);



折射：光学的折射公式，做近似处理，避免三角函数计算



## U3D实现

我选择的渲染管线是URP，着色器语言为HLSL，因为SRP是未来U3D的方向，HLSL才能在SRP中使用SRP Batch。目前网上的文章多为CG语言和Shader Graph方案，我决定挑战一下。

水面的模型我用了200\*200个顶点的平面模型；

顶点运动和法线计算我选择在顶点着色器中实现，我使用了两个正弦波叠加，实际运用可根据情况调整；

水体的表面着色我选择在片段着色器中实现，处理折射时，使用了摄像机的不透明物体贴图（\_CameraOpaqueTexture）；由于\_CameraOpaqueTexture只包含Queue<=AlphaTest+50（2500）的物体，因此这个着色器的Queue>=2501，虽然处于透明物体队列，但是不进行透明度混合。

处理菲涅尔和折射时，使用近似处理以避免三角函数计算。

## 结束

以上是粗略地介绍，具体可以在项目中进行详细考察。场景是Scenes/WaterDemo.unity，各个参数可在Shader面板调整。

不足之处请耐心指教，谢谢！