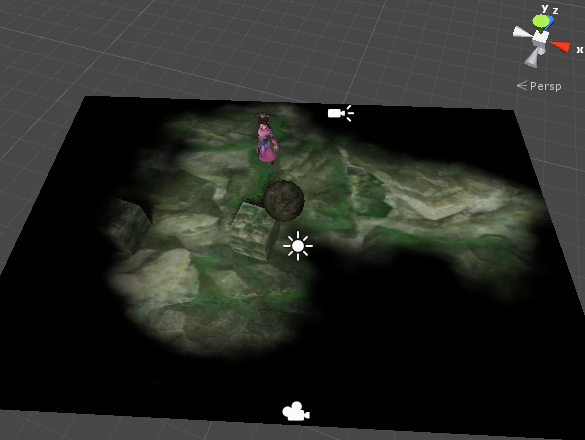
战争迷雾



1. 简介

RTS类型的游戏经常用到战争迷雾。一开始整个地形被迷雾遮蔽，只有Viewer的周围一定范围可见，随着Viewer的移动，未探索的区域变成了可见。

1. 实现方法

本人之前也没有实现过战争迷雾，但是一提到战争迷雾，我的第一反应是用RenderTexture来实现，几乎所有运算都是在屏幕空间中进行，绘制出迷雾Texture后，再用Projecter将迷雾Texture投射到整个场景，有点类似GPGPU的思想。方法的主要步骤如下：

1、创建一个RenderTexture（称为FogOfWarRTT）用于绘制（刷出）迷雾。

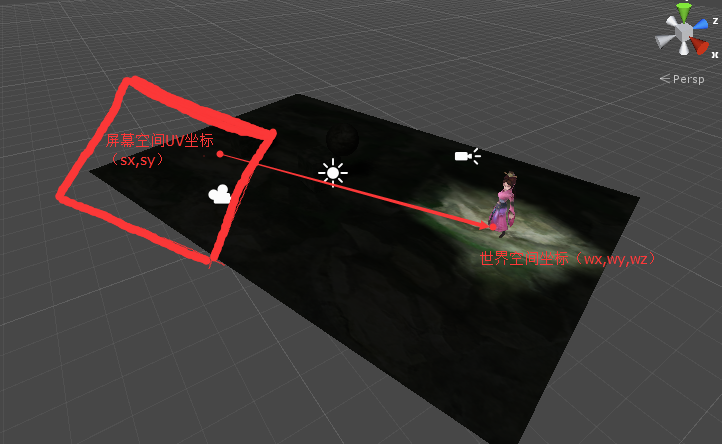
2、设置当前RenderBuffer为FogOfWarRTT，并且不清除该RenderBuffer，这就使得刷上去的迷雾不会被清除掉。

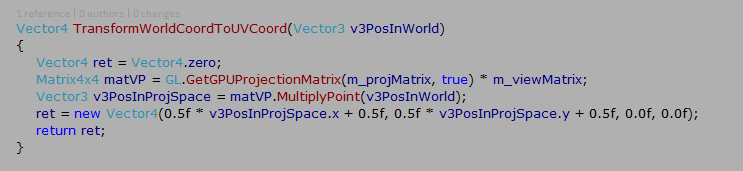
3、创建一个Projecter，用于将FogOfWarRTT投射到场景。

三、实现细节

知道大致过程后，还有一些细节问题需要讨论，如下：

1. 创建好FogOfWarRTT后，应该怎么刷上迷雾呢？换句话说，当Viewer在场景中移动时，怎么样才能擦除FogOfWarRTT上黑色的迷雾？我们只是知道Viewer的世界坐标ViewerCenter，而在FogOfWarRTT上刷迷雾时，用的是屏幕空间的UV坐标，也就是说，怎么将ViewerCenter转化为屏幕空间的UV坐标？如下：

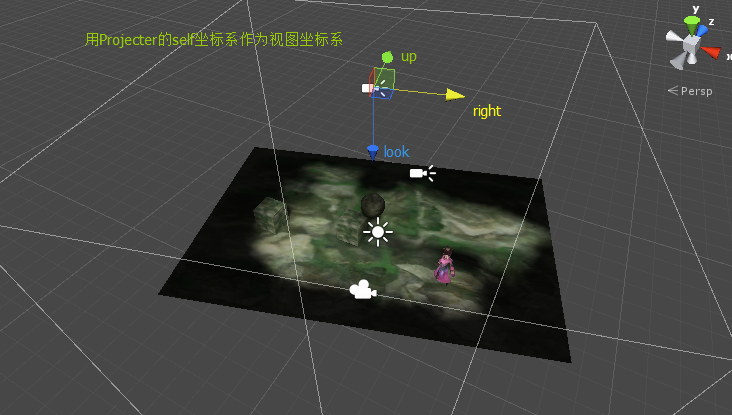


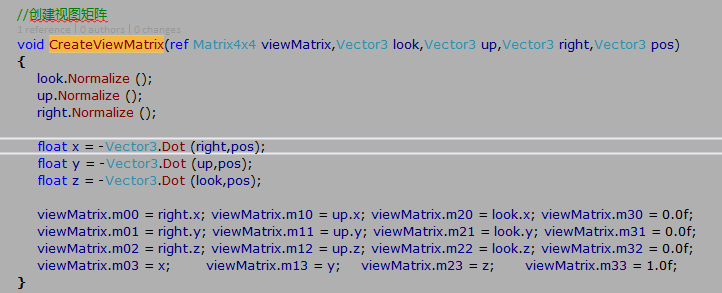


其中m\_projMatrix是投影矩阵，m\_viewMatrix是视图矩阵。这两个矩阵是怎么来的，下面再讨论。

1. 创建Projecter的视图矩阵ViewMatrix

创建视图矩阵的代码如下：

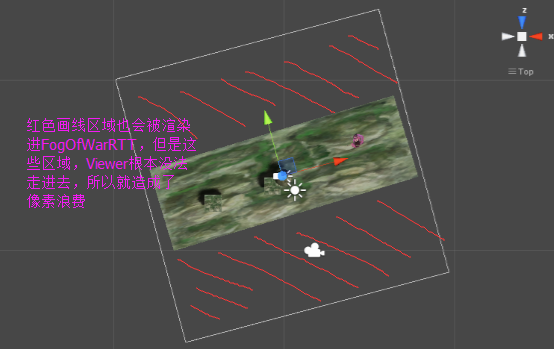


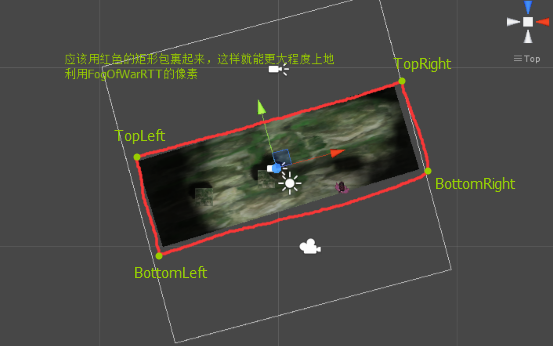


其中 look、up、right和pos都是世界坐标系下的。

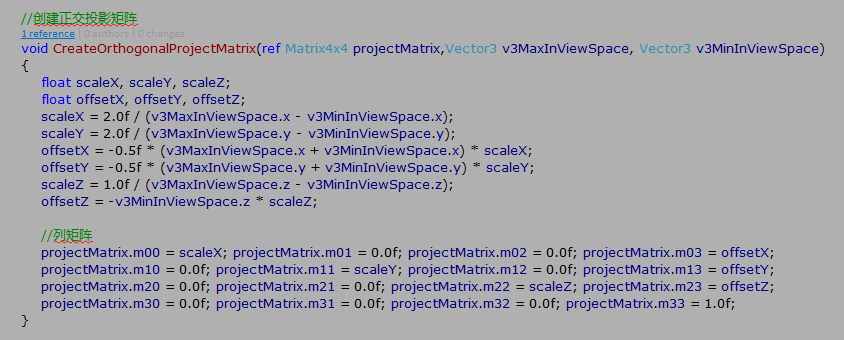
1. 创建正交投影矩阵ProjMatrix。

Projecter投射FogOfWarRTT用的是正交投影矩阵。看到这里读者可能会有疑问，Projecter不是已经有投影矩阵了吗，直接用它的就好了呀，干嘛还这么费劲再自己编写呢？ Projecter的正交投影空间是方形的，对于一些在XZ平面是比较狭长的场景，Projecter的正交投影空间包裹得就不是很严密，这就会造成FogOfWarRTT大量的像素浪费，也就是说FogOfWarRTT有相当大的一部分像素是没有用的，下面这图描述了用Projecter的投影矩阵在FogOfWarRTT上刷迷雾的缺陷。





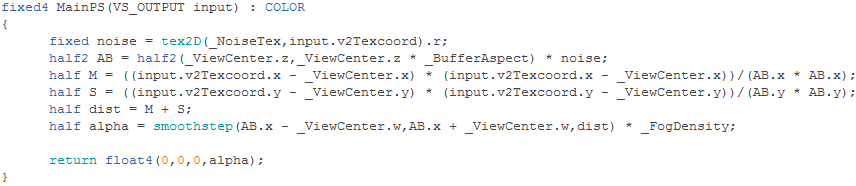
创建正交投影矩阵的代码如下：



其中v3MaxInViewSpace、v3MinInViewSpace分别是场景区域在Projecter的视图空间中的最大坐标点和最小坐标点。

1. 绘制迷雾。

有了Projecter的ViewMatrix和ProjMatrix后，最后一步就是绘制迷雾，下面是绘制迷雾的PixelShader的代码：



half4 \_ViewCenter; //x,y是Viewer的世界坐标转换成的屏幕空间UV坐标，z刷子的半径,w刷子边缘柔和程度

half \_BufferAspect; //场景矩形区域的横纵比

half \_FogDensity; //迷雾浓度

sampler2D \_NoiseTex; //Perlin噪音，对圆形的刷子进行抖动，使之变得不太规则。