Unity Shader中关于时间的内置变量

图形用户界面, 文本, 应用程序

描述已自动生成

纹理动画

1. 序列帧动画

原理：依次播放一系列关键帧图像，当播放速度达到一定数值时，看起来就是一个连续的动画。

优缺点：

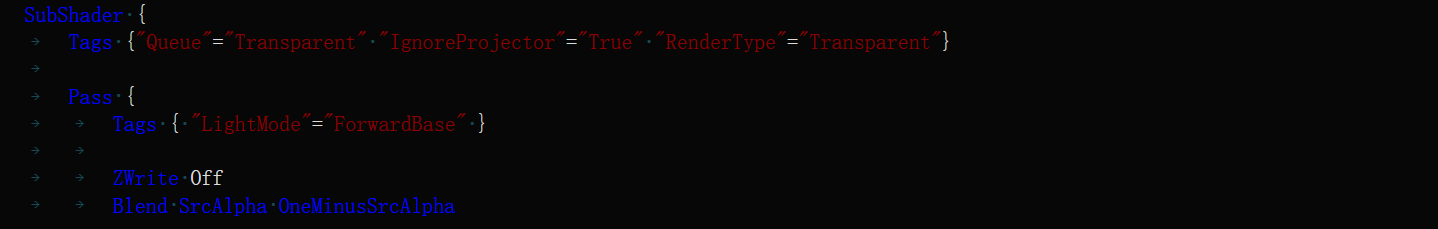
优点：灵活性很强，不需要进行任何物理计算就可以得到非常细腻的动画效果。

缺点：由于序列帧中每张关键帧图像都不一样，要制作一张出色的序列帧纹理所需要的美术工程量也比较大。

文本

描述已自动生成

\_HorizontalAmount代表水平方向上的关键帧图像数目，\_VerticalAmount代表竖直方向上关键帧图像数目，\_Speed用于控制序列帧动画的播放速度



序列帧图像通常是透明纹理，包含了透明通道，因此可以被当成一个半透明对象，因此SubShader中要设置Tag，渲染队列Queue为透明Transparent，是否受到投影器的影响IgnoreProjector设置为True。Pass中开启混合模式并关闭深度写入。

Ps：

Blend SrcAlpha OneMinusSrcAlpha：一种传统的透明混合模式，也称为标准透明混合。它使用源颜色的Alpha通道作为混合因子，将源颜色与目标颜色进行混合。使用源颜色的Alpha值作为权重，将源颜色乘以Alpha值，目标颜色乘以(1 - Alpha)值，然后将它们相加得到最终的混合结果。

Blend One One：一种预乘透明混合模式，也称为预乘混合。它使用源颜色和目标颜色的Alpha通道作为混合因子，将源颜色和目标颜色进行混合。具体来说，它将源颜色和目标颜色分别乘以它们的Alpha值，然后将它们相加得到最终的混合结果。

图片包含 图形用户界面

描述已自动生成

图片包含 形状

描述已自动生成

文本

描述已自动生成

顶点着色器把顶点从模型空间转换到裁剪空间，把顶点纹理坐标存入v2f结构体

文本

描述已自动生成

本质上播放帧动画就是计算出每个时刻需要播放的关键帧在纹理中的位置，即求出关键帧所在的行列索引数。片元着色器中首先把播放速度乘自场景加载以来的时间得到场景加载的时间，使用求出的时间time除以竖直方向上关键帧动画的数量并向下取整可以得到当前的行索引，而余数是当前的列索引。（但是如果要控制某一时刻开始播放动画且动画必须从第一个关键帧开始播放，上面这段代码是不正确的），然后把原纹理i.uv按照行列数等分，得到每个子纹理坐标范围，然后使用当前的行列数对上面uv的结果进行偏移得到当前子图像的坐标。然后把纹理归一化，采样输出。

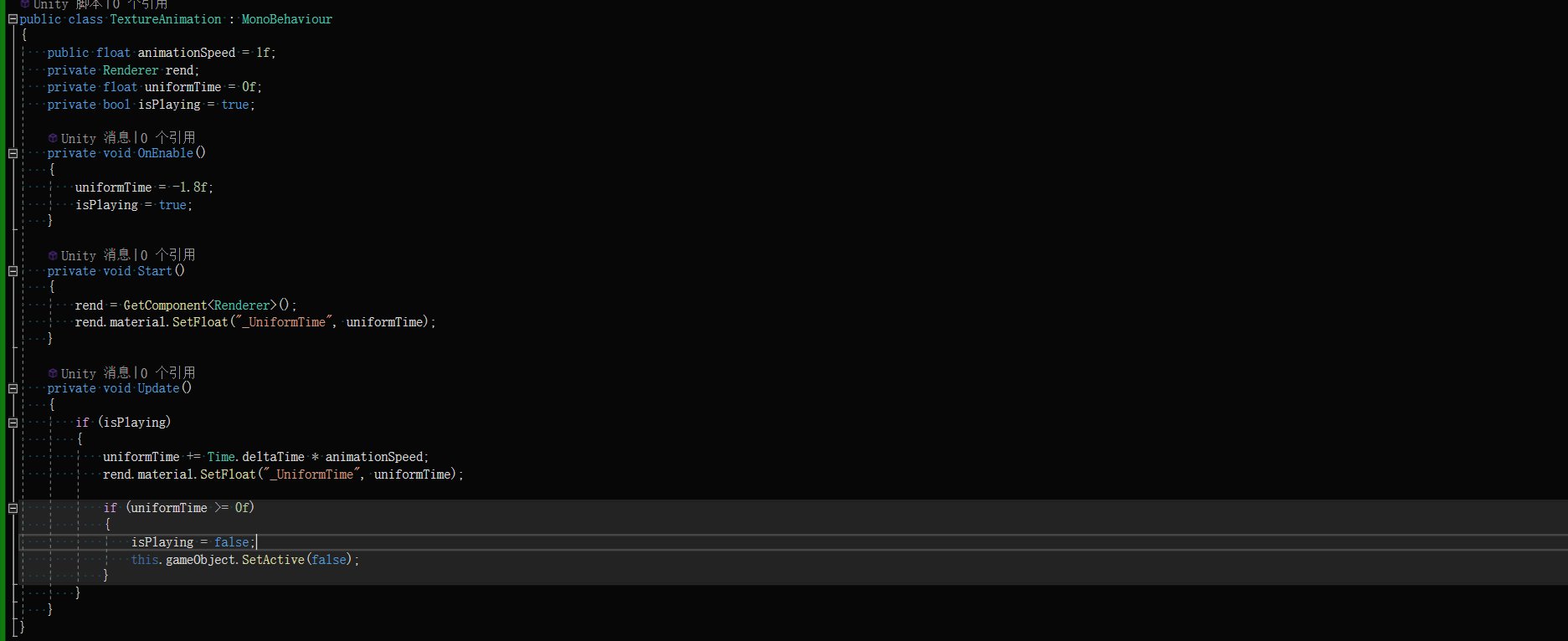
Ps：对竖直方向的坐标偏移要用减法，因为Unity中纹理坐标竖直方向的顺序是从下到上逐渐增大，与序列帧纹理中的顺序从上到下播放相反，因此需要用减法，对水平方向的坐标偏移仍是加法，Unity的纹理坐标的原点(0,0)在纹理左下角。

下面是控制本例的序列帧动画只播放一次并且在任意时刻激活物体都会从同一个关键帧开始播放动画：

文本

描述已自动生成

添加一个\_UniformTime属性用于接收脚本的传参，并用于计算关键帧索引



序列帧动画的材质挂载到的物体上挂载的脚本

图片包含 看着, 水, 小, 黑暗

描述已自动生成

1. 滚动的背景

2D游戏中用不断滚动的背景来模拟角色的穿梭，而这些背景的实现会利用纹理动画。

文本

描述已自动生成

\_MainTex和\_DetailTex分别是第一层（较远）和第二层（较近）的背景纹理，而\_Scrol1X和\_Scrol2X对应了这两层的滚动速度，\_Multiplier参数用于控制纹理的整体亮度

文本

低可信度描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

顶点着色器中把顶点从模型空间转换到裁剪空间，然后分别计算两层背景的纹理坐标，使用内置的\_Time.y变量乘水平和竖直方向偏移速度对纹理坐标进行偏移，使用frac获取相乘得到的小数部分以实现周期性平移

Ps: frac用于获取实数的小数部分。它输入一个实数，返回给定实数的小数部分。

文本

描述已自动生成

对两张背景采样，使用第二层的透明通道湖混合两张纹理，以调整亮度。

Ps：实现角色运动时背景运动，角色静止时背景静止，需要通过脚本接收角色的移动速度给\_Scrol1X和\_Scrol2X

顶点动画

河流的模拟是顶点动画最常见的应用之一。它的原理通常就是使用正弦函数等来模拟水流波动效果。

1. 流动的河流

文本

中度可信度描述已自动生成

\_Magnitude控制水流波动的幅度，\_Frequency控制波动频率，\_InvWaveLength控制波长的倒数，\_Speed控制河流纹理的移动速度

文本

中度可信度描述已自动生成

Queue="Transparent"：指定渲染队列为透明队列，这意味着该材质将在不透明物体之后进行渲染，以确保正确的混合和透明效果。

IgnoreProjector="True"：忽略投影器，这意味着该材质将不会受到投影器的影响。

RenderType="Transparent"：指定渲染类型为透明，这将告诉渲染管线将该材质视为透明材质，并相应地进行渲染处理。

DisableBatching="True"：指定禁用批处理，这将阻止渲染管线对使用该材质的物体进行批处理优化。禁用批处理可能会导致性能下降，但在某些情况下可能是必要的，例如，当使用动态纹理或需要单独渲染的物体时。

开启并设置混合模式，关闭深度写入，关闭剔除功能

Ps: 剔除（Culling）是一种在渲染过程中排除不可见物体或面的技术。剔除功能可以提高渲染性能，减少不必要的渲染开销。

手机屏幕的截图

描述已自动生成

电脑萤幕画面

中度可信度描述已自动生成

顶点着色器中首先计算顶点的位移量，本例中只希望对顶点的x方向进行位移，因此把yzw的位移量设置为0（下图中可以清楚的看到波浪是在x方向产生的，即只对x方向进行位移）。认为顶点在坐标轴上向着坐标轴正方向移动，\_Frequency \* \_Time.y得到顶点当前所在的轴上的位置；v.vertex.x \* \_InvWaveLength + v.vertex.y \* \_InvWaveLength + v.vertex.z \* \_InvWaveLength 的结果使得渲染过程中形成了波浪效果。

图表, 折线图

描述已自动生成

因为模型空间坐标系是这样的，所以v.vertex.z \* \_InvWaveLength导致波浪的形成，v.vertex.x \* \_InvWaveLength使得波浪有粗处也有细处，就像下面这样

图表, 表面图

描述已自动生成  
而v.vertex.y \* \_InvWaveLength实际上在本例中不起任何作用，删掉也不影响波浪表现

文本

描述已自动生成

图片包含 水, 橙子, 海, 蛋糕

描述已自动生成

广告牌

广告牌技术会根据视角方向旋转一个被纹理着色的多边形，使得多边形看起来好总是面对摄像机。广告牌技术常应用于渲染烟雾、云朵，闪光等效果。

本质是构建旋转矩阵。使用的基向量通常是表面法线（normal）、指向上的方向（up）、指向右的方向（right）以及锚点（anchor location）。难点在于如何构建3个相互正交的基向量。首先会通过初始计算得到目标的表面法线和指向上的方向，两者往往是不垂直的，但两者其中之一是固定的。通过这可以计算出指向右的方向：Right=Up x Normal。例如模拟例子效果时，法线总是指向视角方向， 计算出Right后，在叉积计算出修改后向上的方向：Up=Normal x Right。

文本

描述已自动生成

\_VerticalBillboarding用于调整固定法线或是固定指向上的方向

图片包含 文本

描述已自动生成

文本

描述已自动生成

选择模型空间的原点作为广告牌的锚点center，然后获取模型空间下的视角位置，然后计算三个正交矢量，首先根据观察位置减去锚点位置得到目标法线方向（因为广告牌技术顶点动画会永远与视角方向垂直，所以这样就可以得到表面法线方向），当约束度\_VerticalBillboarding 为1时意味着法线方向固定为视角方向，若为0则固定为向上方向(0，1，0)，然后归一化。为防止法线方向与up方向平行从而导致叉积的结果错误，需要对法线的y分量进行判断，以得到合适的up方向向量，然后根据up方向向量和表面法线方向叉积得到right方向，自此得到正交基。根据原始位置相对于锚点的偏移量以及3个正交基矢量计算得到新的顶点位置，然后把模型空间变换到裁剪空间，

文本

描述已自动生成

图片包含 徽标

描述已自动生成

3.注意事项

如果要对顶点动画物体添加阴影，需要自己完成一个ShadowCasterPass