<https://blog.csdn.net/puppet_master/article/details/52975666>

**1 基本拉伸效果**

Shader "Unlit/ScreenWaterWave"

{

Properties

{

\_MainTex ("Texture", 2D) = "white" {}

}

CGINCLUDE

#include "UnityCG.cginc"

sampler2D \_MainTex;

float4 \_MainTex\_ST;

fixed4 \_StartPosition;

float \_WaveSpeed;

float \_Distort;

struct v2f

{

float4 pos : SV\_POSITION;

float2 uv : TEXCOORD0;

};

v2f vert(appdata\_img v)

{

v2f o;

UNITY\_INITIALIZE\_OUTPUT(v2f, o);

o.pos = UnityObjectToClipPos(v.vertex);

o.uv = TRANSFORM\_TEX(v.texcoord, \_MainTex);

return o;

}

fixed4 frag(v2f i) : SV\_Target

{

fixed2 offset = i.uv - \_StartPosition.xy;

fixed4 col = tex2D(\_MainTex, i.uv + \_Distort \* normalize(offset));

return col;

}

ENDCG

SubShader

{

Tags { "RenderType"="Opaque" }

ZTest Always

ZWrite Off

Cull Off

Pass

{

CGPROGRAM

#pragma vertex vert

#pragma fragment frag

ENDCG

}

}

}



**2 波纹式拉伸**

在上面的例子中，通过像素采样时uv坐标增加一个偏移值来达到拉伸效果，我们就可以让这个偏移值作为这个三角函数的输出，这样，有的地方拉伸少，有的地方拉伸多，就形成了不同的拉伸夏鸥共，也就形成了一个波纹的感觉，那么，输入就很明显了，输入就是距离中心位置的绝对距离。

fixed4 frag(v2f i) : SV\_Target

{

fixed2 dv = \_StartPosition.xy - i.uv;

// 保证在不同的分辨率上都是圆形

dv = dv \* float2(\_ScreenParams.x / \_ScreenParams.y, 1);

fixed distance = length(dv);

// distance是小于1的，需要乘以一个比较大的数，这样才有多个波峰波谷

float sinFactor = sin(distance \* \_DistanceScale - \_Time.y \* 10);

float discardFactor0 = clamp(distance - \_CurrentWaveWidth, 0, 1);

float discardFactor1 = clamp(\_WaveWidth - distance, 0, 1);

fixed4 col = tex2D(\_MainTex, i.uv + \_Distort \* sinFactor \* discardFactor0 \* discardFactor1 \* normalize(dv));

return col;

}

