**解决物体透明穿插问题**

Posted on 2013年04月02日 by U3d / [Unity3D 基础教程](http://www.unitymanual.com/category/manual/unity3d-%e5%9f%ba%e7%a1%80%e6%95%99%e7%a8%8b)/被围观 280 次

首先，调节透明度，改变物体的透明程度，也可以做那种从无到有的渐变出现的动画，改变maincolor的alpha值就可以实现。

效果图：

[](http://www.unitymanual.com/wp-content/uploads/2013/04/111.jpg)

解决物体透明穿插问题

下面[**Unity3D脚本**](http://www.unitymanual.com/category/script)可以控制透明的shader。

Shader "Transparent/Diffuse\_zbuffer" {  
Properties {  
\_Color ("Main  
Color", Color) = (1,1,1,1)  
\_MainTex ("Base (RGB) Trans (A)", 2D) =  
"white" {}  
}

Category {  
Tags {"Queue"="Transparent" "IgnoreProjector"="True"  
"RenderType"="Transparent"}  
LOD 200  
Alphatest Greater  
0  
ZWrite On //就是这里，原来是ZWrite Off  
ColorMask  
RGB

//  
------------------------------------------------------------------  
//  
ARB fragment program

SubShader {  
// Ambient  
pass  
Pass {  
Name  
"BASE"  
Tags {"LightMode" =  
"ixelOrNone"}  
Fog { Color [\_AddFog]  
}  
Blend SrcAlpha  
OneMinusSrcAlpha  
Color  
[\_PPLAmbient]  
SetTexture [\_MainTex] {constantColor [\_Color]  
Combine texture \* primary DOUBLE, texture \*  
primary}  
}  
// Vertex lights  
Pass {

Name "BASE"  
Tags {"LightMode" =  
"Vertex"}  
Fog { Color [\_AddFog]  
}  
Blend SrcAlpha  
OneMinusSrcAlpha  
Lighting On  
Material  
{  
Diffuse  
[\_Color]  
Emission  
[\_PPLAmbient]  
}  
SetTexture [\_MainTex]  
{combine texture \* primary DOUBLE, texture \*  
primary}  
}  
// Pixel lights  
Pass  
{  
Name "L"  
Tags { "LightMode"  
= "ixel" }  
Blend SrcAlpha One  
Fog {  
Color [\_AddFog] }

CGPROGRAM  
#pragma fragment frag  
#pragma vertex vert  
#pragma  
multi\_compile\_builtin\_noshadows  
#pragma fragmentoption  
ARB\_fog\_exp2  
#pragma fragmentoption ARB\_precision\_hint\_fastest  
#include  
"UnityCG.cginc"  
#include "AutoLight.cginc"

struct v2f  
{  
V2F\_POS\_FOG;  
LIGHTING\_COORDS  
float2 uv;  
float3 normal;  
float3 lightDir;  
};

uniform float4 \_MainTex\_ST;

v2f vert (appdata\_base v)  
{  
v2f o;  
PositionFog(  
v.vertex, o.pos, o.fog );  
o.normal = v.normal;  
o.uv =  
TRANSFORM\_TEX(v.texcoord,\_MainTex);  
o.lightDir = ObjSpaceLightDir(  
v.vertex );  
TRANSFER\_VERTEX\_TO\_FRAGMENT(o);  
return o;  
}

uniform sampler2D \_MainTex;  
uniform float4 \_Color;

float4 frag (v2f i) : COLOR  
{  
half4 texcol = tex2D( \_MainTex,  
i.uv );  
half4 c = DiffuseLight( i.lightDir, i.normal, texcol,  
LIGHT\_ATTENUATION(i) );  
c.a = texcol.a \* \_Color.a;  
return  
c;  
}  
ENDCG

SetTexture [\_MainTex] {combine  
texture}  
SetTexture [\_LightTexture0] {combine  
texture}  
SetTexture [\_LightTextureB0] {combine  
texture}  
}  
}

//  
------------------------------------------------------------------  
//  
Radeon 9000

SubShader {  
// Ambient pass  
Pass  
{  
Blend SrcAlpha OneMinusSrcAlpha  
Name  
"BASE"  
Tags {"LightMode" =  
"ixelOrNone"}  
Color  
[\_PPLAmbient]  
SetTexture [\_MainTex] {constantColor [\_Color]  
Combine texture \* primary DOUBLE, texture \*  
constant}  
}  
// Vertex lights  
Pass {

Blend SrcAlpha OneMinusSrcAlpha  
Name  
"BASE"  
Tags {"LightMode" =  
"Vertex"}  
Lighting On  
Material  
{  
Diffuse  
[\_Color]  
Emission  
[\_PPLAmbient]  
}  
SetTexture [\_MainTex]  
{Combine texture \* primary DOUBLE, texture \*  
primary}  
}

// Pixel lights with 0  
light textures  
Pass {  
Blend SrcAlpha  
One  
Name "L"  
Tags {

"LightMode" = "ixel"

"LightTexCount" = "0"  
}

CGPROGRAM  
#pragma vertex vert  
#include "UnityCG.cginc"

struct v2f {  
V2F\_POS\_FOG;  
float2 uv :  
TEXCOORD0;  
float3 normal : TEXCOORD1;  
float3  
lightDir : TEXCOORD2;  
};

uniform float4 \_MainTex\_ST;

v2f vert(appdata\_base v)  
{  
v2f o;  
PositionFog( v.vertex,  
o.pos, o.fog );  
o.normal = v.normal;  
o.uv =  
TRANSFORM\_TEX(v.texcoord,\_MainTex);  
o.lightDir = ObjSpaceLightDir(  
v.vertex );  
return o;  
}  
ENDCG  
Program ""  
{  
SubProgram {  
Local  
0, [\_ModelLightColor0]  
Local 1, [\_Color]

"!!ATIfs1.0  
StartConstants;  
CONSTANT c0 =  
program.local[0];  
CONSTANT c1 = program.local[1];  
EndConstants;

StartOutputPass;  
SampleMap r0, t0.str; # main  
texture  
SampleMap r1, t2.str; # normalized light  
dir  
PassTexCoord r2, t1.str; #  
normal

DOT3 r5.sat, r2, r1.2x.bias; # R5 = diffuse  
(N.L)

MUL r0.rgb, r0, r5;  
MUL r0.rgb.2x, r0,  
c0;  
MUL r0.a, r0, c1;  
EndPass;

"  
}  
}  
SetTexture[\_MainTex]  
{combine texture}  
SetTexture[\_CubeNormalize] {combine  
texture}  
}

// Pixel lights with 1  
light texture  
Pass {  
Blend SrcAlpha  
One  
Name "L"  
Tags {

"LightMode" = "ixel"

"LightTexCount" = "1"  
}

CGPROGRAM  
#pragma vertex vert  
#include "UnityCG.cginc"

uniform float4 \_MainTex\_ST;  
uniform float4x4  
\_SpotlightProjectionMatrix0;

struct v2f {  
V2F\_POS\_FOG;  
float2 uv :  
TEXCOORD0;  
float3 normal : TEXCOORD1;  
float3  
lightDir : TEXCOORD2;  
float4 LightCoord0 : TEXCOORD3;  
};

v2f vert(appdata\_tan v)  
{  
v2f o;  
PositionFog( v.vertex,  
o.pos, o.fog );  
o.normal = v.normal;  
o.uv =  
TRANSFORM\_TEX(v.texcoord,\_MainTex);  
o.lightDir = ObjSpaceLightDir(  
v.vertex );

o.LightCoord0 = mul(\_SpotlightProjectionMatrix0,  
v.vertex);

return o;

}  
ENDCG  
Program ""  
{  
SubProgram {  
Local  
0, [\_ModelLightColor0]  
Local 1, [\_Color]

"!!ATIfs1.0  
StartConstants;  
CONSTANT c0 =  
program.local[0];  
CONSTANT c1 = program.local[1];  
EndConstants;

StartOutputPass;  
SampleMap r0, t0.str; # main  
texture  
SampleMap r1, t2.str; # normalized light  
dir  
PassTexCoord r4, t1.str; # normal  
SampleMap r2,  
t3.str; # a = attenuation

DOT3 r5.sat, r4,  
r1.2x.bias; # R5 = diffuse (N.L)

MUL r0.rgb, r0,  
r5;  
MUL r0.rgb.2x, r0, c0;  
MUL r0.rgb, r0,  
r2.a; # attenuate  
MUL r0.a, r0, c1;  
EndPass;

"  
}  
}  
SetTexture[\_MainTex]  
{combine texture}  
SetTexture[\_CubeNormalize] {combine  
texture}  
SetTexture[\_LightTexture0] {combine  
texture}  
}

// Pixel lights with 2  
light textures  
Pass {  
Blend SrcAlpha  
One  
Name "L"  
Tags  
{  
"LightMode" =  
"ixel"  
"LightTexCount" =  
"2"  
}  
CGPROGRAM  
#pragma vertex vert  
#include  
"UnityCG.cginc"

uniform float4 \_MainTex\_ST;  
uniform float4x4  
\_SpotlightProjectionMatrix0;  
uniform float4x4  
\_SpotlightProjectionMatrixB0;

struct v2f {  
V2F\_POS\_FOG;  
float2 uv :  
TEXCOORD0;  
float3 normal : TEXCOORD1;  
float3  
lightDir : TEXCOORD2;  
float4 LightCoord0 :  
TEXCOORD3;  
float4 LightCoordB0 : TEXCOORD4;  
};

v2f vert(appdata\_tan v)  
{  
v2f o;  
PositionFog( v.vertex,  
o.pos, o.fog );  
o.normal = v.normal;  
o.uv =  
TRANSFORM\_TEX(v.texcoord,\_MainTex);  
o.lightDir = ObjSpaceLightDir(  
v.vertex );

o.LightCoord0 = mul(\_SpotlightProjectionMatrix0,  
v.vertex);  
o.LightCoordB0 = mul(\_SpotlightProjectionMatrixB0,  
v.vertex);

return o;

}  
ENDCG  
Program ""  
{  
SubProgram {  
Local  
0, [\_ModelLightColor0]  
Local 1, [\_Color]

"!!ATIfs1.0  
StartConstants;  
CONSTANT c0 =  
program.local[0];  
CONSTANT c1 = program.local[1];  
EndConstants;

StartOutputPass;  
SampleMap r0, t0.str; # main  
texture  
SampleMap r1, t2.str; # normalized light  
dir  
PassTexCoord r4, t1.str; # normal  
SampleMap r2,  
t3.stq\_dq; # a = attenuation 1  
SampleMap r3,  
t4.stq\_dq; # a = attenuation 2

DOT3 r5.sat, r4,  
r1.2x.bias; # R5 = diffuse (N.L)

MUL r0.rgb, r0,  
r5;  
MUL r0.rgb.2x, r0, c0;  
MUL r0.rgb, r0,  
r2.a; # attenuate  
MUL r0.rgb, r0, r3.a;  
MUL  
r0.a, r0, c1;  
EndPass;

"  
}  
}  
SetTexture[\_MainTex]  
{combine texture}  
SetTexture[\_CubeNormalize] {combine  
texture}  
SetTexture[\_LightTexture0] {combine  
texture}  
SetTexture[\_LightTextureB0] {combine  
texture}  
}  
}

//  
------------------------------------------------------------------  
//  
Radeon 7000

SubShader {  
Material  
{  
Diffuse [\_Color]  
Emission  
[\_PPLAmbient]  
}  
Lighting On  
Fog {  
Color [\_AddFog] }  
Pass {  
Blend SrcAlpha  
OneMinusSrcAlpha  
Name "BASE"  
Tags  
{"LightMode" = "ixelOrNone"}  
Color  
[\_PPLAmbient]  
Lighting Off  
SetTexture  
[\_MainTex] {Combine texture \* primary DOUBLE}  
SetTexture  
[\_MainTex] {Combine texture \* primary DOUBLE}  
SetTexture  
[\_MainTex] {Combine texture \* primary DOUBLE, primary \*  
texture}  
}  
Pass {

Blend SrcAlpha  
OneMinusSrcAlpha  
Name "BASE"  
Tags  
{"LightMode" = "Vertex"}  
SetTexture [\_MainTex] {Combine  
texture \* primary DOUBLE, primary \*  
texture}  
}  
Pass {  
Blend  
SrcAlpha One  
Name "L"  
Tags  
{  
"LightMode" =  
"ixel"  
"LightTexCount" =  
"2"  
}  
SetTexture [\_LightTexture0]  
{ combine previous \* texture alpha, previous  
}  
SetTexture  
[\_LightTextureB0] {  
combine previous \* texture  
alpha + constant, previous  
constantColor  
[\_PPLAmbient]  
}  
SetTexture [\_MainTex]  
{ combine previous \* texture DOUBLE, primary \*  
texture}  
}  
Pass {  
Blend  
SrcAlpha One  
Name "L"  
Tags  
{  
"LightMode" =  
"ixel"  
"LightTexCount" =  
"1"  
}  
SetTexture [\_LightTexture0]  
{  
combine previous \* texture alpha + constant,  
previous  
constantColor  
[\_PPLAmbient]  
}  
SetTexture [\_MainTex]  
{ combine previous \* texture DOUBLE, primary \*  
texture}  
}  
Pass {  
Blend  
SrcAlpha One  
Name "L"  
Tags  
{  
"LightMode" =  
"ixel"  
"LightTexCount" =  
"0"  
}  
SetTexture [\_MainTex] {  
combine previous \* texture DOUBLE, primary \*  
texture}  
}  
}  
}

// Fallback to Alpha Vertex Lit  
Fallback "Transparent/VertexLit", 2

}