

【翻译】第五章节：剖面模型（关于片段擦除和三角形面剔除）

2014-11-26 08:30:00 2211 人阅读 [Unity3D](#) [cg](#) [剖面模型](#)

A⁻ A⁺

文章内容	例子源码	网友评论	最后编辑：2014-12-21 17:54:54
本文永久地址： http://www.omuying.com/article/93.aspx ，【 文章转载请注明出处！ 】			

原文链接：http://en.wikibooks.org/wiki/Cg_Programming/Unity/Cutaways

这个教程覆盖了表面擦除（discarding fragments）、前脸剔除（front-face）和后脸剔除（back-face）。本篇文章假定你熟悉《[RGB 立方体](#)》章节。

性能消耗非常低的剖面

本篇教程的主题是切去渲染网格中三角形或者片段。这么做主要有两个原因：我们想看穿一个三角形或者片段，另一情况是三角形始终是不可见的。同时我们不处理不可见的元素还可以节省一些性能。这两种裁剪方式 GPU 都支持，我们将要讨论他们。

下面的着色器可以用非常低的开销来剔除网格，当坐标 y 为正（大于零）时，片段会被裁剪掉。着色器的代码如下：

```
01 Shader "Cg shader using discard"
02 {
03     SubShader
04     {
05         Pass
06         {
07             Cull Off
08             // turn off triangle culling, alternatives are:
09             // Cull Back (or nothing): cull only back faces
10             // Cull Front : cull only front faces
11
12             CGPROGRAM
13
14             #pragma vertex vert
15             #pragma fragment frag
16
17             struct vertexInput
18             {
19                 float4 vertex : POSITION;
20             };
21             struct vertexOutput
22             {
23                 float4 pos : SV_POSITION;
24                 float4 posInObjectCoords : TEXCOORD0;
25             };
26
27             vertexOutput vert(vertexInput input)
28             {
29                 vertexOutput output;
30
31                 output.pos = mul(UNITY_MATRIX_MVP, input.vertex);
32                 output.posInObjectCoords = input.vertex;
33
34                 return output;
35             }
36
37             float4 frag(vertexOutput input) : COLOR
38             {
39                 if (input.posInObjectCoords.y > 0.0)
40                 {
41                     discard; // drop the fragment if y coordinate > 0
42                 }
43                 return float4(0.0, 1.0, 0.0, 1.0); // green
44             }
45             ENDCG
46         }
47     }
48 }
```



游戏开发学习



比基尼除毛



伪装微型摄像机



unity3d摄像机



微型摄像机



无线监控摄像机



室内设计师




unity3d移动

最新文章




暂无图片

【原创】C# 基础之 Lambda表达式 - 907 次阅读




暂无图片

【原创】C#基础之 IEnumerable和 IEnumerator - 792 次阅读




暂无图片

【原创】C#基础之事件 - 886 次阅读



暂无图片

【原创】C#基础之委托 - 912 次阅读



暂无图片

【原创】C#基础之委托的使用 - 856 次阅读



伪装微型摄像机



超高速摄像机



上海猎头公司



防爆摄像机



猎头公司



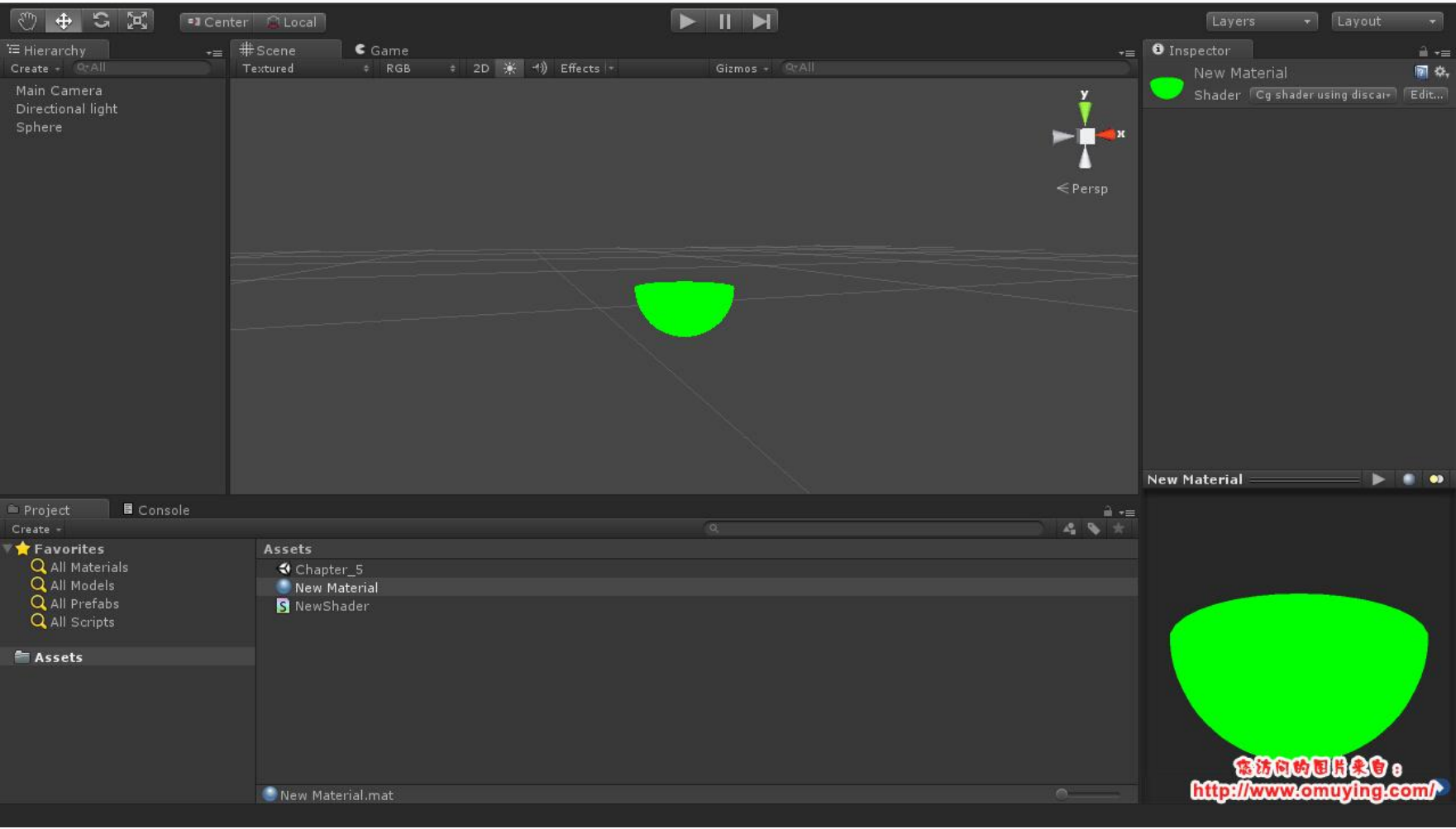
猎头公司排名

游戏开发学习

防爆红外摄像机

随机阅读

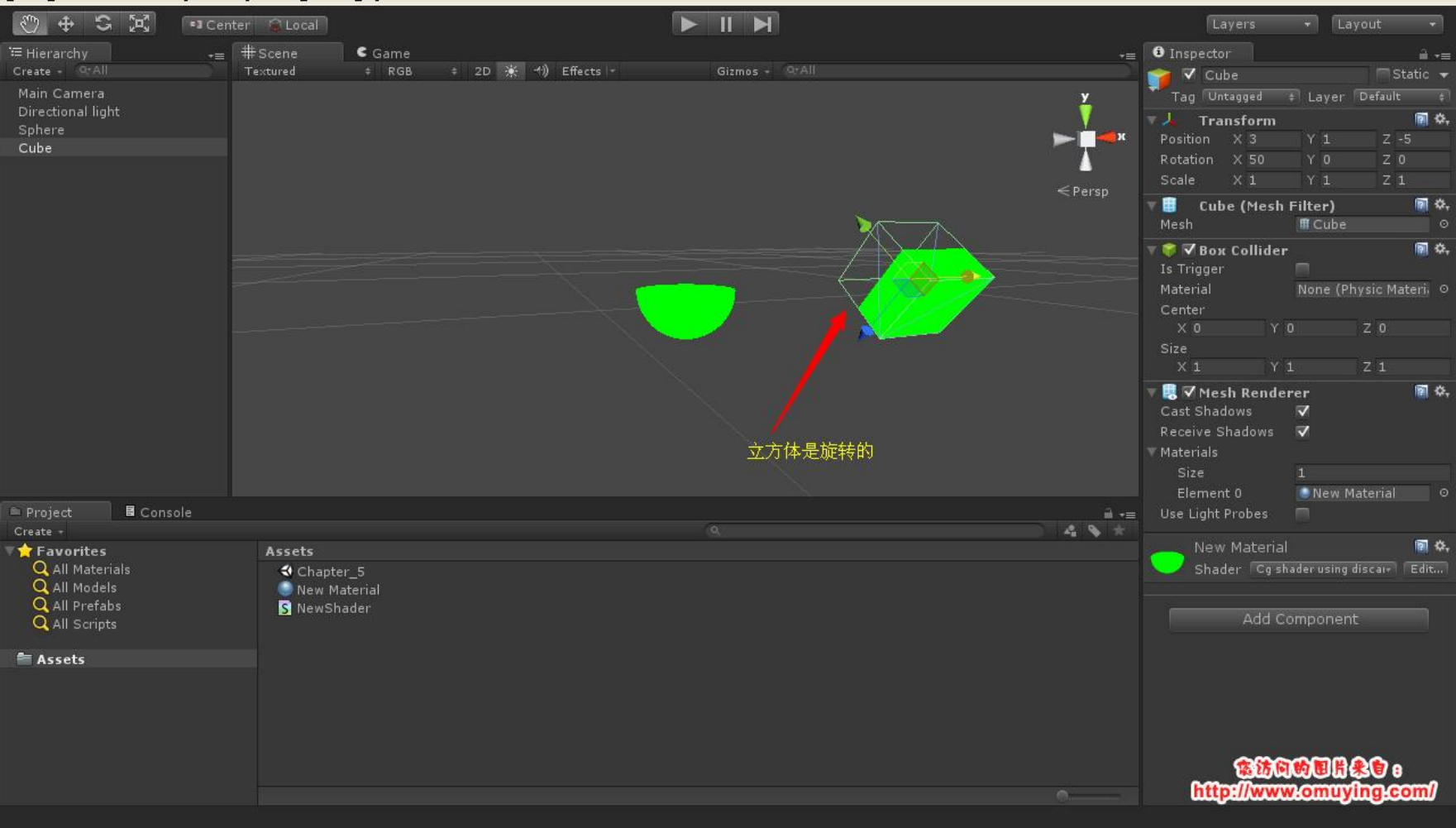
当把这个着色器应用于一个默认的对象时，着色器会把他们切去一半。如图：



片段剔除

在着色器中，我们首先关注擦除指令（discard instruction）。基本上这个指令只擦除片段（在早期的着色器语言中，这被称为“kill”，不过我更喜欢“discard”）。取决于硬件条件，如果一个着色器包含擦除指令，性能可能会非常糟糕（无论擦除多少片段，都会对性能有影响）。因此你应该避免使用这个指令，尤其是在遇到性能相关的问题时。

还要说明一点：片段 discard 的条件只包含在一个对象坐标系中。所以任何旋转或者移动中的对象也都可以被裁剪，如图：



更好的剖面

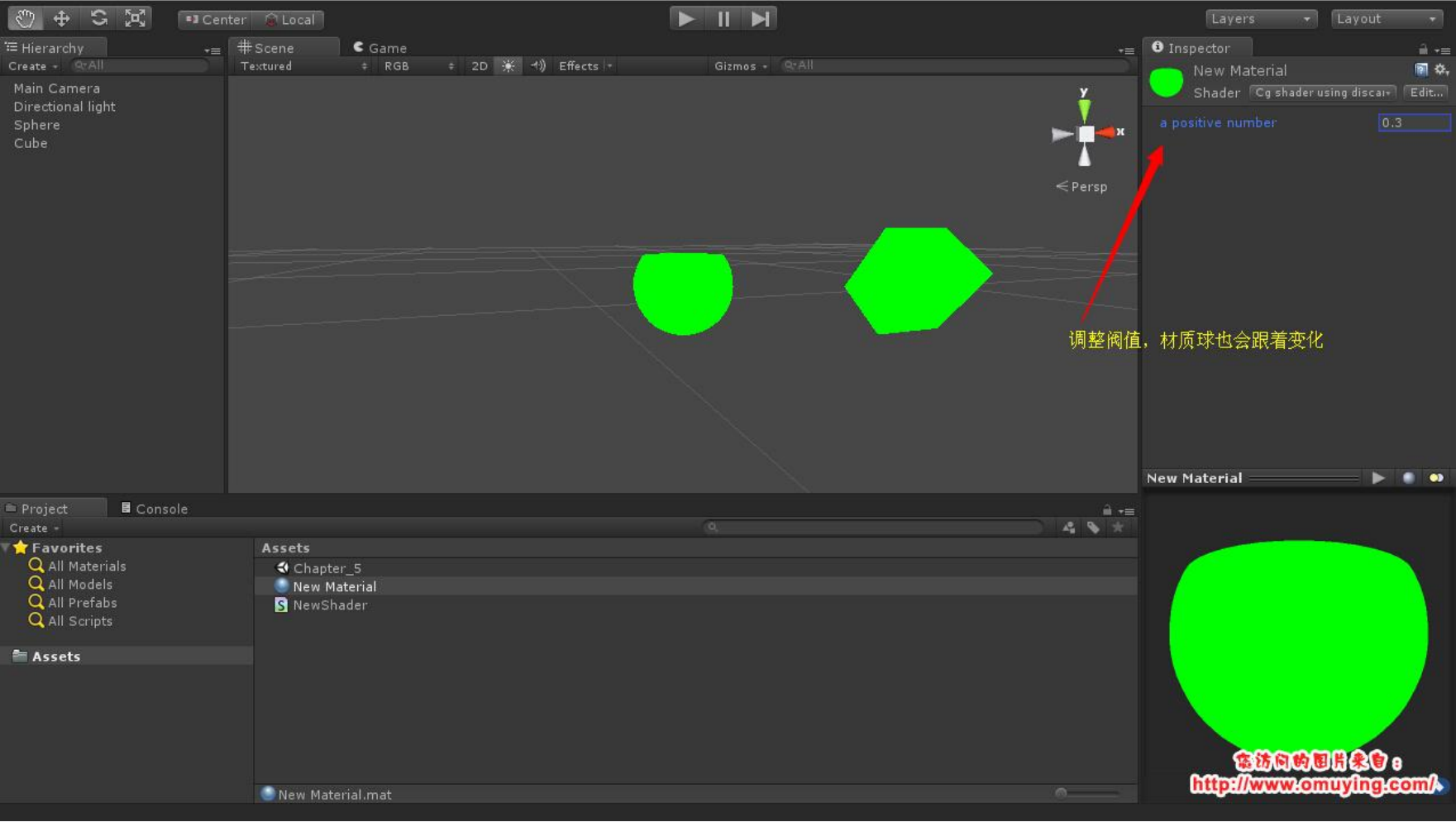
另外你还可以给着色器添加一个自定义的属性，然后通过判断坐标 y 的值是否大于自定义属性的值，来决定是否擦除片段，着色器的代码如下：

```
01 Shader "Cg shader using discard"
02 {
03     Properties
04     {
05         _Threshold ("a positive number", Float) = 0.0
06     }
07
08     SubShader
09     {
10         Pass
11         {
12             Cull Off
13         }
14     }
15 }
```

- 【翻译】第十一章节：双面表面（关于双面每顶点光照） - 1187 次阅读
- 【原创】Shader 内置 Shader 之 Diffuse Detail 学习 - 1460 次阅读
- 【原创】Shader 表面着色器语法 - 2660 次阅读
- 【原创】Shader 内置 Shader 之 Normal-Diffuse 学习 - 1479 次阅读
- 【翻译】第九章节：漫反射（关于每顶点漫反射和多光源漫反射） - 1946 次阅读

```
14 CGPROGRAM
15
16 #pragma vertex vert
17 #pragma fragment frag
18
19 uniform float _Threshold;
20
21 struct vertexInput
22 {
23     float4 vertex : POSITION;
24 };
25 struct vertexOutput
26 {
27     float4 pos : SV_POSITION;
28     float4 posInObjectCoords : TEXCOORD0;
29 };
30
31 vertexOutput vert(vertexInput input)
32 {
33     vertexOutput output;
34
35     output.pos = mul(UNITY_MATRIX_MVP, input.vertex);
36     output.posInObjectCoords = input.vertex;
37
38     return output;
39 }
40
41 float4 frag(vertexOutput input) : COLOR
42 {
43     if (input.posInObjectCoords.y > _Threshold)
44     {
45         discard; // drop the fragment if y coordinate > 0
46     }
47     return float4(0.0, 1.0, 0.0, 1.0); // green
48 }
49 ENDCG
50 }
51 }
52 }
```

通过改变阈值，我们可以看到材质球的效果跟着变化，如图：



如果你熟悉 Unity 的脚本，你可以编写一个脚本并挂载到一个对象上，接着在脚本中定义一个球体变量并且设置好引用，然后把这个球体对象的逆模型矩阵（`renderer.worldToLocalMatrix`）分配（使用 `renderer.sharedMaterial.SetMatrix()`）给着色器中的一个 `float4x4` uniform 参数，在着色器中，计算片段的世界坐标位置并且将球体的逆模型矩阵应用到这个片段位置，现在你在球体对象的本地坐标系的也有片段位置，因为在 Unity 的球体坐标系统中，球的半径是 0.5，所以你很容易判断片段是否在球体中，如果在球体的内部，需要擦除片段，这样最后生成的脚本和着色器可以根据球的位置来切除任何对象表面的点，着色器的代码如下：

```
01 Shader "Cg shader using discard"
02 {
03     SubShader
04     {
05         Pass
06         {
07             Cull Off
08
09             CGPROGRAM
10
```

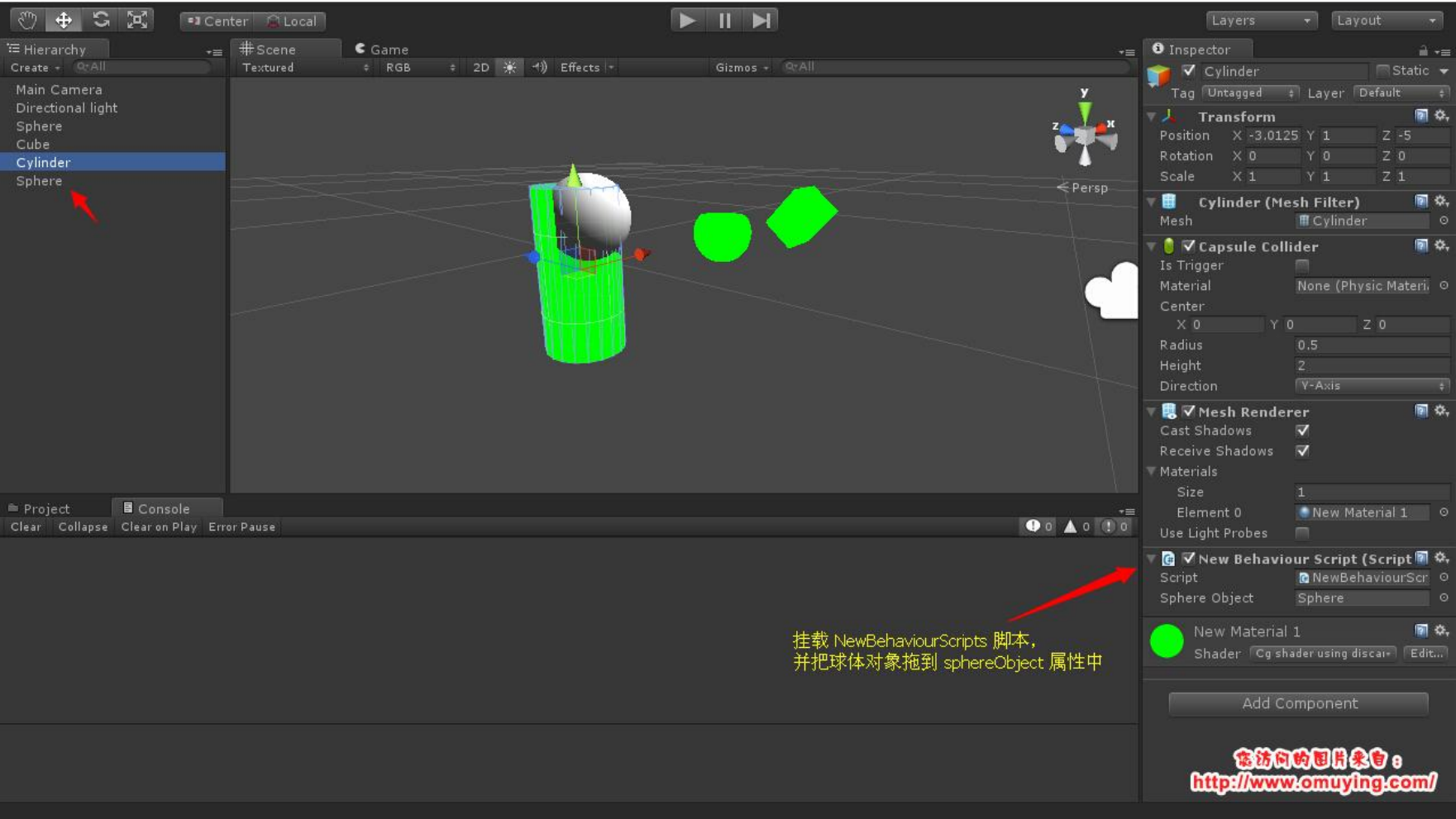


```
11      #pragma vertex vert
12      #pragma fragment frag
13
14      uniform float4x4 _SphereMatrix; // 球体矩阵
15
16      struct vertexInput
17      {
18          float4 vertex : POSITION;
19      };
20      struct vertexOutput
21      {
22          float4 pos : SV_POSITION;
23          float4 posInObjectCoords : TEXCOORD0;
24      };
25
26      vertexOutput vert(vertexInput input)
27      {
28          vertexOutput output;
29
30          output.pos = mul(UNITY_MATRIX_MVP, input.vertex);
31          output.posInObjectCoords = mul(_SphereMatrix, mul(_Object2World,
input.vertex)); //把顶点坐标映射到球体坐标系中
32
33          return output;
34      }
35
36      float4 frag(vertexOutput input) : COLOR
37      {
38          // 判断是否在球体内
39          if (abs(input.posInObjectCoords.x) <= 0.5 &&
abs(input.posInObjectCoords.y) <= 0.5 && abs(input.posInObjectCoords.z) <=
0.5)
40          {
41              discard; // drop the fragment if y coordinate > 0
42          }
43          return float4(0.0, 1.0, 0.0, 1.0); // green
44      }
45      ENDCG
46  }
47 }
48 }
```

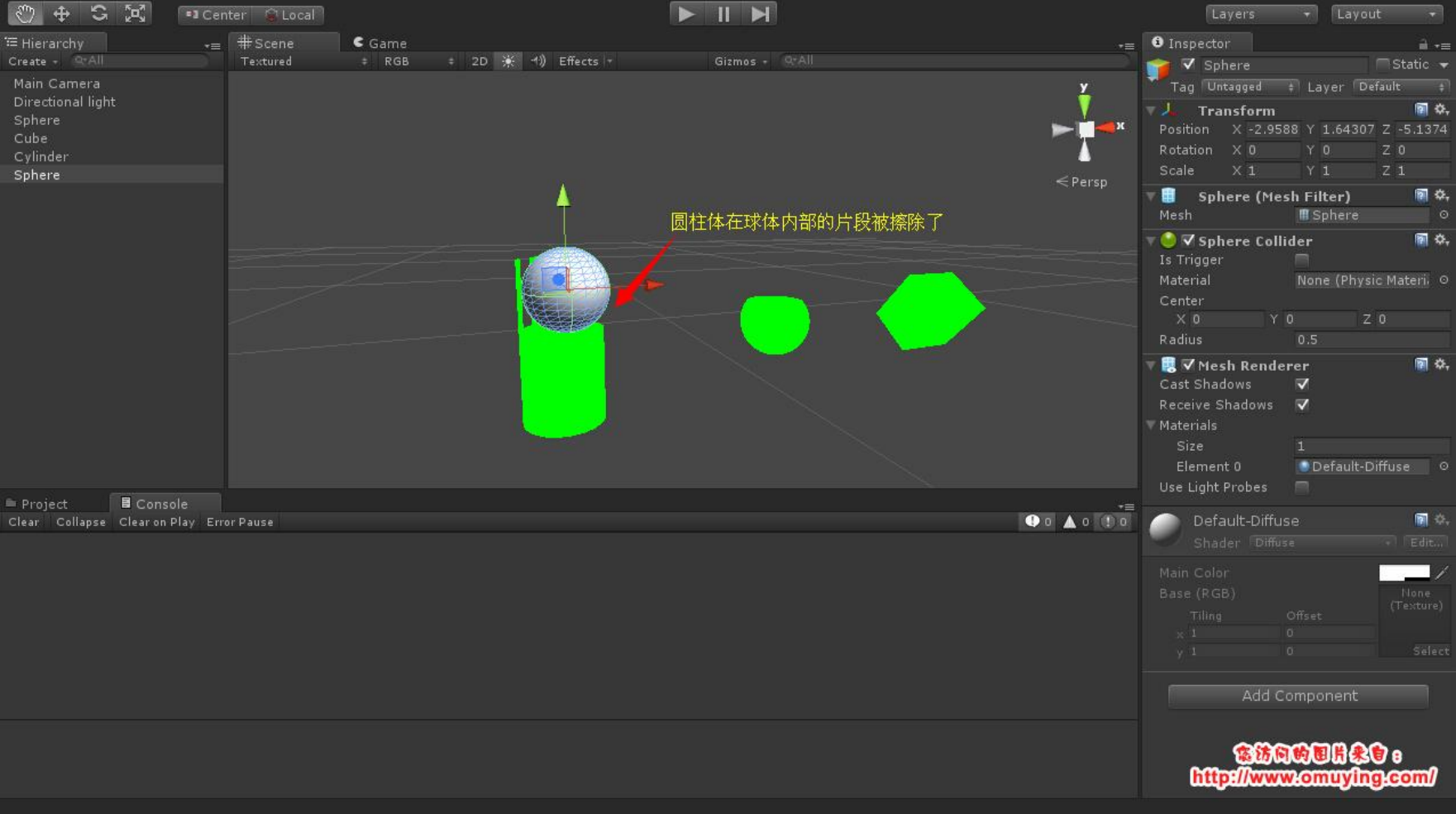
新建一个 C# 脚本，代码如下：

```
01 using UnityEngine;
02 using System.Collections;
03
04 [ExecuteInEditMode]
05 public class NewBehaviourScript : MonoBehaviour
06 {
07     public GameObject sphereObject;
08
09     void Update()
10     {
11         this.renderer.sharedMaterial.SetMatrix("_SphereMatrix",
this.sphereObject.transform.worldToLocalMatrix);
12     }
13 }
```

然后把脚本挂载到一个对象上，并且设置 sphereObject 属性为一个球体对象，如图：



拖动球体在挂载脚本对象的周围运行，观察效果，在球体内部的片段会被擦除，如图：



前脸和后脸剔除

最后，第一个着色器中有一条语句是 Cull Off，这行在 CGPROGRAM 标记之前，因为 Cg 中不存在。实际上，这是 Unity ShaderLab 关闭三角形剔除命令。因为在 Unity 中，默认的剔除命令是 Cull Back。其实可以指定 Cull Front 命令。Cull Back 命令是剔除背面三角形，因为对象的内部是不可见的。因此背面剔除可以避免栅格化这些三角形并且提高程序的性能。因为在例子中我们擦除了一些片段是为了能够使用着色器可以看到对象的内部，因此我们需要停用背面剔除。

剔除是工作方式是什么？

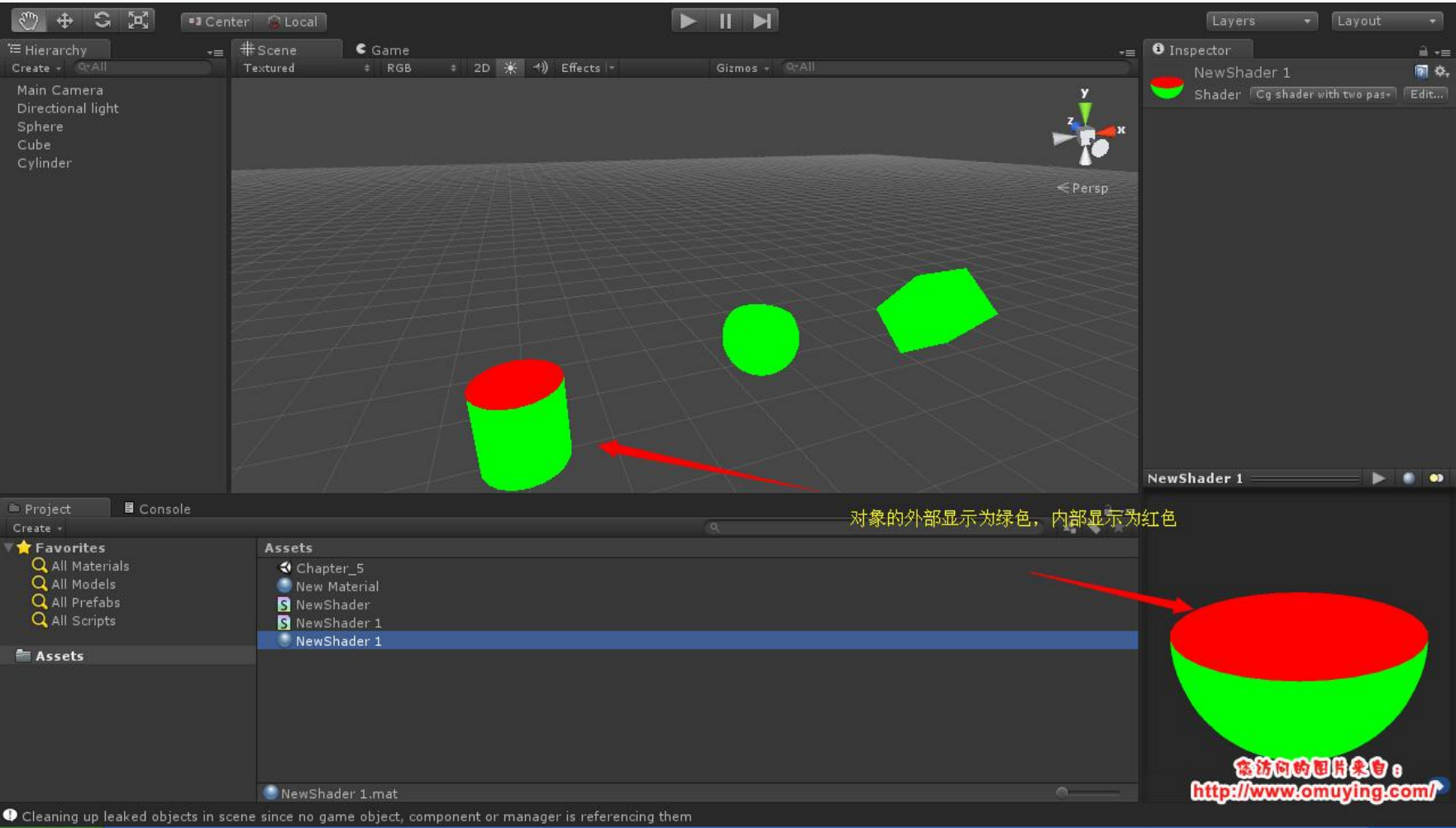
着色器像往常一样处理三角形和顶点，然后 viewport 把顶点信息转换到屏幕坐标中（详情可以查看《Vertex Transformations》），之后图形处理器确定三角形的顶点是顺时针还是逆时针出现在屏幕中，基于这个原理，每个三角形都被标记为前脸或者后脸三角形，如果是前脸三角形并且剔除前脸三角形被启动，它将停止栅格化并且被剔除，然后用类似的方式来剔除后脸三角形，否则，会像往常一样处理该三角形。

我们可以用剔除做什么，一个应用是用不同的着色器来显示对象的外部（前脸）与内部（后脸），下面的着色器使用了两个 pass，第一个 pass 只有前脸被剔除，剩余的片段呈现红色，第二个 pass 只有后脸被剔除，剩余的片段呈现绿色，着色器的代码如下：

```
01 Shader "Cg shader with two passes using discard"
02 {
03     SubShader
04     {
05         // first pass (is executed before the second pass)
06         Pass
07         {
08             Cull Front // cull only front faces
09
10             CGPROGRAM
11
12             #pragma vertex vert
13             #pragma fragment frag
14
15             struct vertexInput
16             {
17                 float4 vertex : POSITION;
18             };
19             struct vertexOutput
20             {
21                 float4 pos : SV_POSITION;
22                 float4 posInObjectCoords : TEXCOORD0;
23             };
24
25             vertexOutput vert(vertexInput input)
26             {
27                 vertexOutput output;
28
29                 output.pos = mul(UNITY_MATRIX_MVP, input.vertex);
30                 output.posInObjectCoords = input.vertex;
31
32                 return output;
33             }
34
35             float4 frag(vertexOutput input) : COLOR
36             {
```

```
37         if (input.posInObjectCoords.y > 0.0)
38         {
39             discard; // drop the fragment if y coordinate > 0
40         }
41         return float4(1.0, 0.0, 0.0, 1.0); // red
42     }
43     ENDCG
44 }
45 // second pass (is executed after the first pass)
46 Pass
47 {
48     Cull Back // cull only back faces
49
50     CGPROGRAM
51
52     #pragma vertex vert
53     #pragma fragment frag
54
55     struct vertexInput
56     {
57         float4 vertex : POSITION;
58     };
59     struct vertexOutput
60     {
61         float4 pos : SV_POSITION;
62         float4 posInObjectCoords : TEXCOORD0;
63     };
64
65     vertexOutput vert(vertexInput input)
66     {
67         vertexOutput output;
68
69         output.pos = mul(UNITY_MATRIX_MVP, input.vertex);
70         output.posInObjectCoords = input.vertex;
71
72         return output;
73     }
74
75     float4 frag(vertexOutput input) : COLOR
76     {
77         if (input.posInObjectCoords.y > 0.0)
78         {
79             discard; // drop the fragment if y coordinate > 0
80         }
81         return float4(0.0, 1.0, 0.0, 1.0); // green
82     }
83     ENDCG
84 }
85 }
86 }
```

把着色器附加到一个材质上，然后把材质应用到一个对象中，如图：



请记住，着色器只会执行一个 subshader，但是每个 subshader 中的 pass 都会被执行。

原则上，Cg 中也有其他的办法来区分前脸与后脸三角形，但是在 Unity 中，他们都不适用。

恭喜你，在本章节中，你学习了：

- 1、如果擦除片段。

- 2、如何指定剔除前脸还是后脸。
- 3、怎样在着色器的 subshader 中使用两个 Pass 来剔除里面和外面的网格。

资源下载地址：[点击下载](#)，共下载 28 次。

前一篇：[第四章：世界空间中的着色器（关于 uniforms）](#)

后一篇：[第六章：透明度（关于混合）](#)



赞

13 人



打酱油

1 人



呵呵

0 人



鄙视

0 人



正能量

0 人



8条评论

最新 最早 最热



changok88

有个疑问还请大神指点，第一个Pass剔除前面，第二个Pass剔除后面，那不是前面后面都没了么？

2015年10月14日 [← 回复](#) [♥ 顶](#) [→ 转发](#)



沉浮

有贴图的怎么办？

2015年11月5日 [← 回复](#) [♥ 顶](#) [→ 转发](#)



.最终幻想.

回复 changok88: 是什么都没有了，所以才能看到里面的红色部分啊！否则如果只剔除内部，我们是看不到的，具体你可以改一下参数试一下！

2015年11月5日 [← 回复](#) [♥ 顶](#) [→ 转发](#)



.最终幻想.

回复 沉浮: 有贴图也无所谓，反正都是 RGB 像素，看逻辑是怎么写的了

2015年11月5日 [← 回复](#) [♥ 顶](#) [→ 转发](#)



ron

请问 output.posInObjectCoords = mul(_SphereMatrix, mul(_Object2World, input.vertex)); 这句怎么理解？🐼

2015年12月3日 [← 回复](#) [♥ 顶](#) [→ 转发](#)



海风

谢谢作者的无私奉献，提点小小的建议："前脸和后脸剔除"应该翻译成"正面剔除和背面剔除"吧？

2015年12月10日 [← 回复](#) [♥ 顶](#) [→ 转发](#)



杨彬

请问一下 discard和clip有什么区别啊？

3月4日 [← 回复](#) [♥ 顶](#) [→ 转发](#)




GOPSv

万 部 A 片高清 国产日韩 hTTp://T.CN/Rt8jUdO




8月24日 [← 回复](#) [♥ 顶](#) [→ 转发](#)

社交帐号登录: [微信](#) [微博](#) [QQ](#) [人人](#) [更多»](#)



说点什么吧...




发布

最终幻想正在使用多说




8条评论

最新 最早 最热



changok88

有个疑问还请大神指点，第一个Pass剔除前面，第二个Pass剔除后面，那不是前面后面都没了么？


2015年10月14日  回复  顶  转发




沉浮




有贴图的怎么办？

2015年11月5日  回复  顶  转发



 .最终幻想.

回复 changok88: 是什么都没有了，所以才能看到里面的红色部分啊！否则如果只剔除内部，我们是看不到的，具体你可以改一下参数试一下！


2015年11月5日  回复  顶  转发



 .最终幻想.




回复 沉浮: 有贴图也无所谓，反正都是 RGB 像素，看逻辑是怎么写的了


2015年11月5日  回复  顶  转发



ron




请问 output.posInObjectCoords = mul(_SphereMatrix, mul(_Object2World, input.vertex)); 这句怎么理解？🐼

2015年12月3日  回复  顶  转发



海风

谢谢作者的无私奉献，提点小小的建议："前脸和后脸剔除"应该翻译成"正面剔除和背面剔除"吧？


2015年12月10日  回复  顶  转发



杨彬


请问一下 discard和clip有什么区别啊？




3月4日  回复  顶  转发








GOPsv

万 部 A 片高清 国产日韩 hTtp://T.CN/Rt8jUdO




8月24日  回复  顶  转发

社交帐号登录:  微信  微博  QQ  人人 [更多»](#)



说点什么吧...



发布

最终幻想正在使用多说