

卡通渲染

风格和影视化探索



卡通

写实



赛璐璐



立绘质感



混合PBR质感

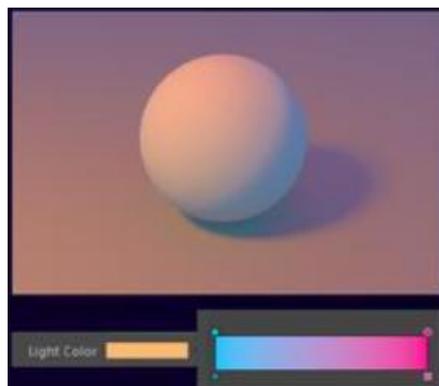


真实质感

构成卡通感的几个要素



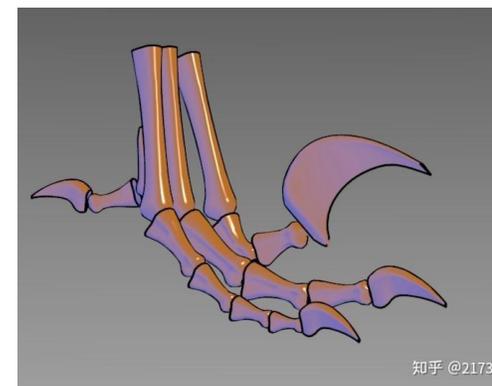
减少色阶的数量



明暗色调分离



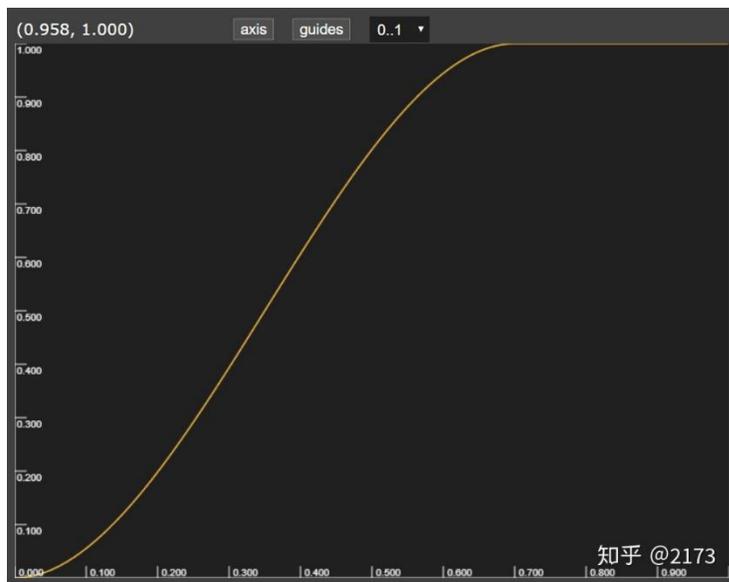
阴影手绘控制



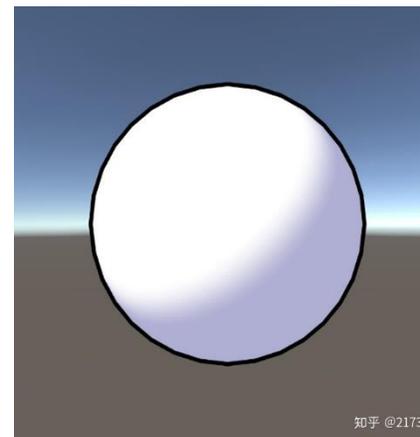
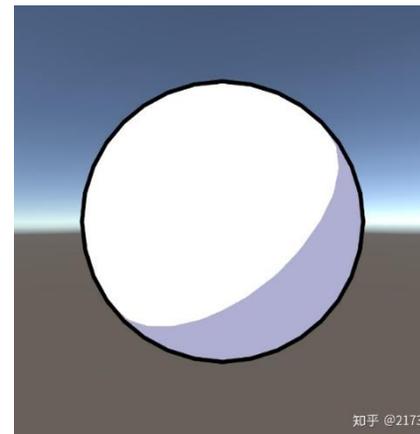
描边

色阶控制

```
half halfLambert = dot(worldNormal, worldLightDir) * 0.5 + 0.5;  
half ramp = smoothstep(0, _ShadowSmooth, halfLambert - _ShadowRange);  
half3 diffuse = lerp(_ShadowColor, _MainColor, ramp);
```



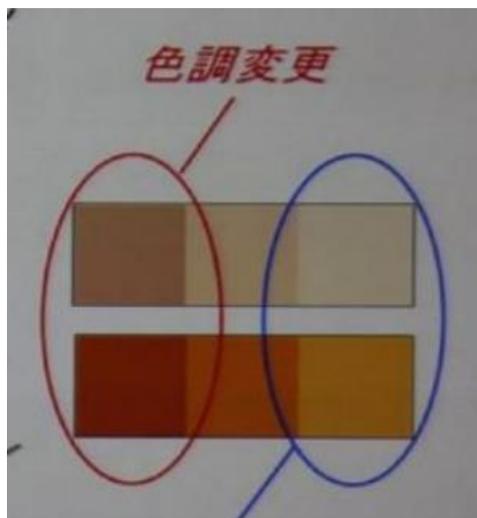
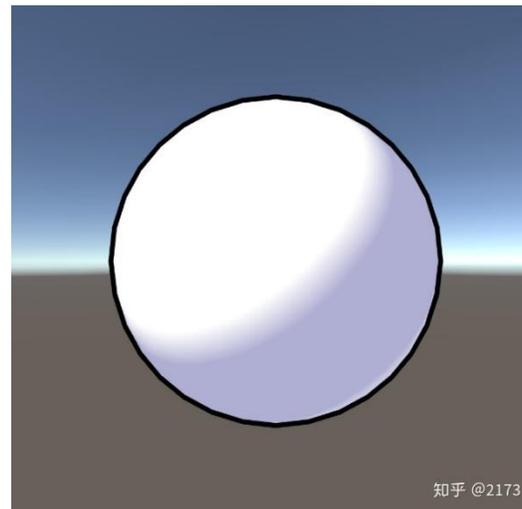
使用Smoothstep函数将光照结果二值化



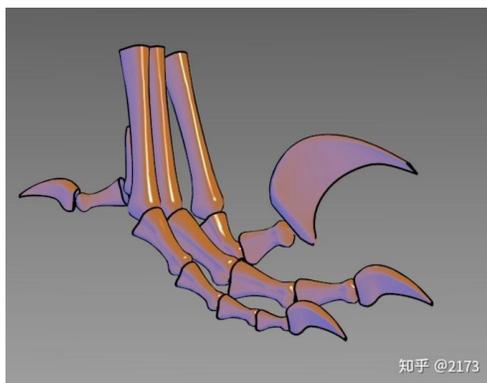
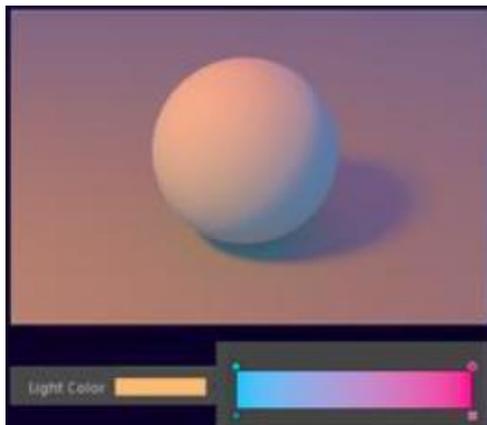
色阶控制



使用ramp贴图减少色阶数量并调整色调

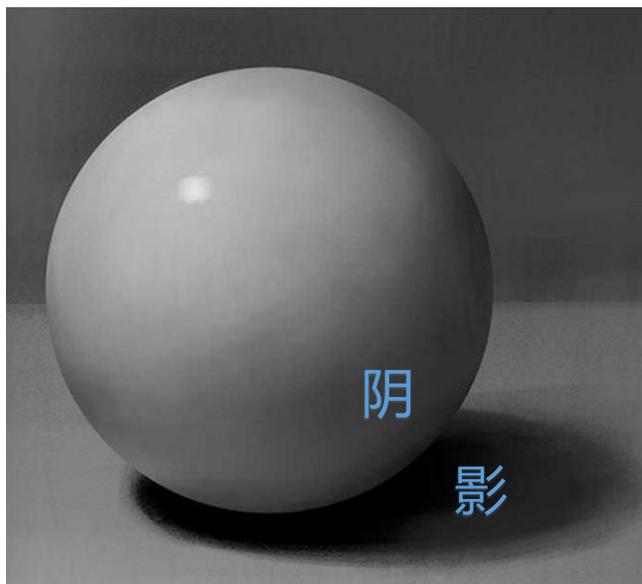


明暗色调控制

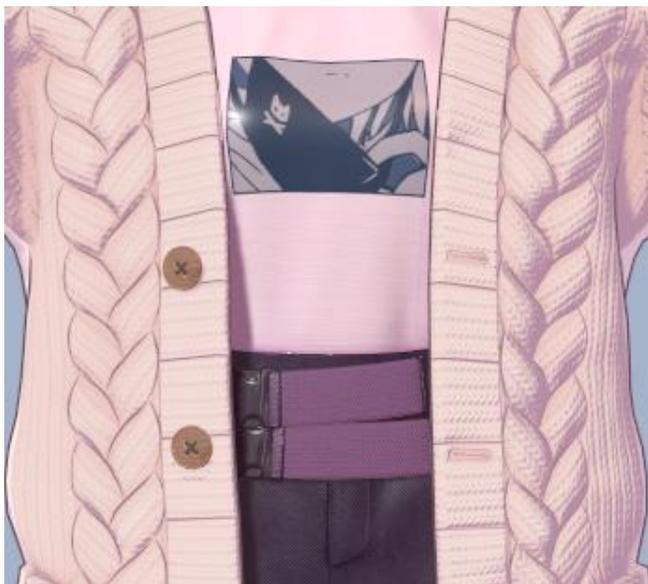


亮面使用偏暖的色调，暗面使用偏冷的色调。
色调的变化使得材质有卡通感

阴影颜色统一



“阴”和“影”颜色统一



阴和影颜色一致



阴和影颜色不一致

阴影区域控制

A Breakdown of Cel-shading



The diagram illustrates the concept of cel-shading. At the top left, a sun icon emits three vertical yellow arrows representing light rays. Below this, a circle is shown with its top half white and bottom half black, representing a lit and shadowed surface. To the right of the circle, a coordinate system shows a vertical axis and a horizontal axis, with an arc indicating a 90-degree angle between them.

- Generic step(Threshold , $\text{dot}(\text{Light}, \text{Normal})$) can decide if the surface is lit or not
- Simply put, if the surface **Normal** is facing the **Light** source, it's lit. If it's facing away more than 90° (the **Threshold**), it's not

Only Three components matter

- **Threshold**
- **Light vector**
- **Normal vector**

Control them ALL!! 知乎 @2173

影响阴影的因素

1.灯光方向

2.阈值

3.法线方向

灯光方向控制

角色单独打光

光照方向加偏移

```
Light mainLight = GetMainLight(input.shadowCoord);  
half3 worldLightDir = mainLight.direction;  
worldLightDir.x += _LightOffsetX;  
worldLightDir.y += _LightOffsetY;  
worldLightDir = normalize(worldLightDir);
```

阴影区域控制



绘制阴影阈值贴图控制阴影形状

法线控制



编辑法线控制阴影形状



通过法线传递功能，传递平滑法线到模型上

描边



外描边

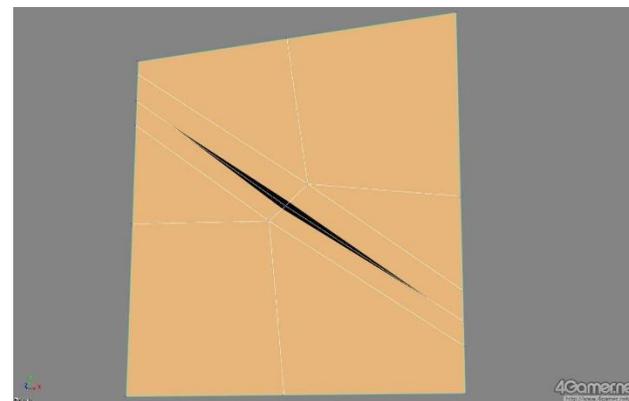
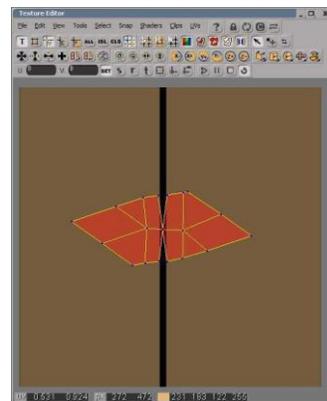
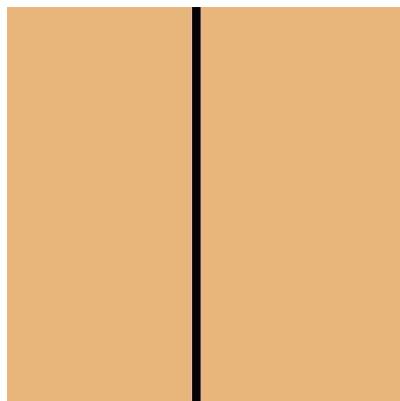


外描边+内描边

内描边

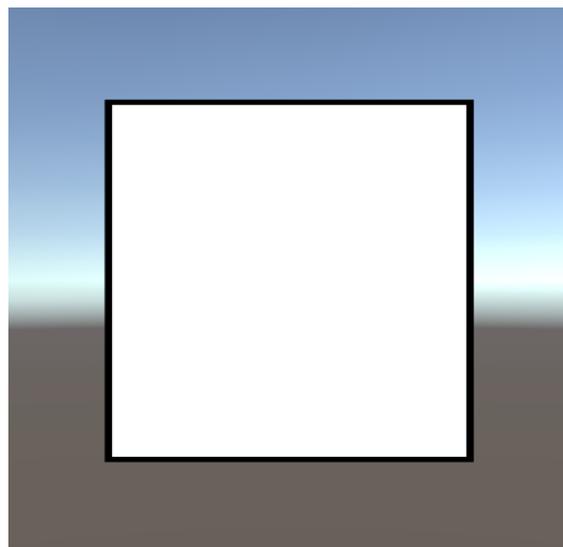
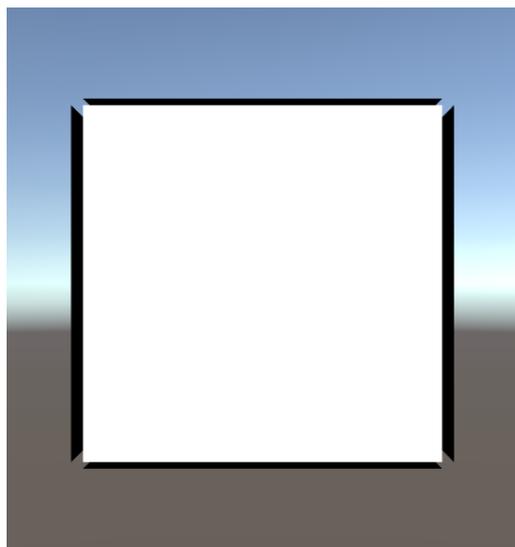


贴图放大后锯齿

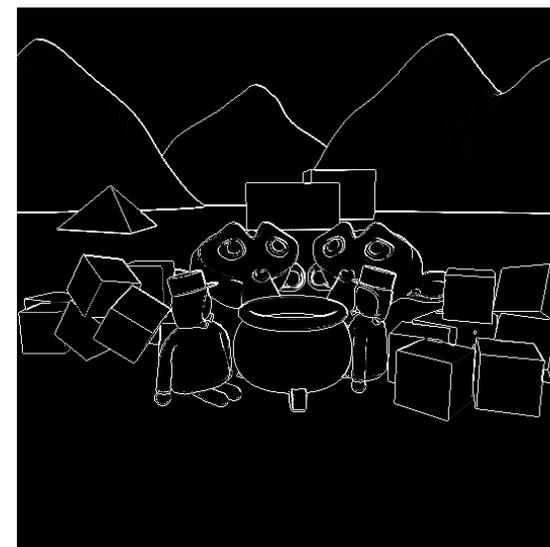


本村线

外描边

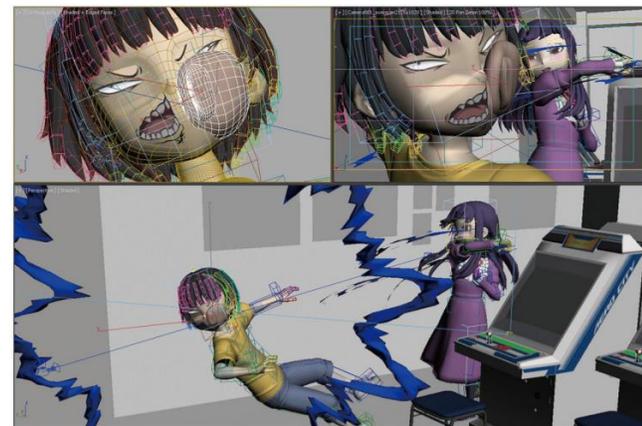
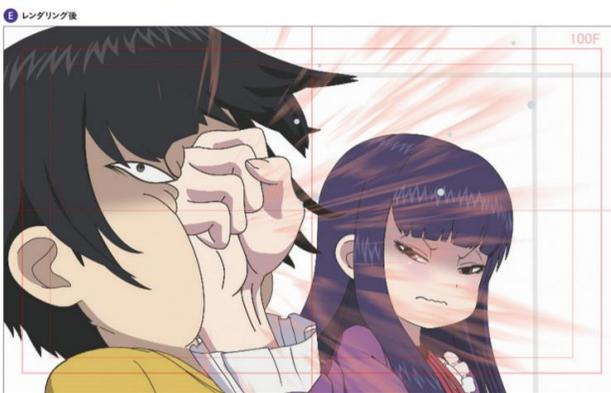


Back Facing两次绘制

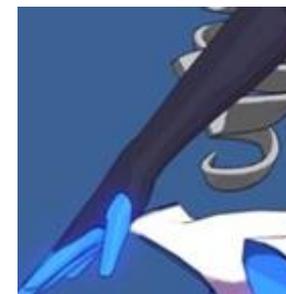


Sobel算子边缘检测

赛璐璐的优势



赛璐璐的材质表达



清漆光 (N·V)

赛璐璐表达皮肤、布料、金属和皮革的材质。基本可以满足人物需要

赛璐璐材质表达的缺陷

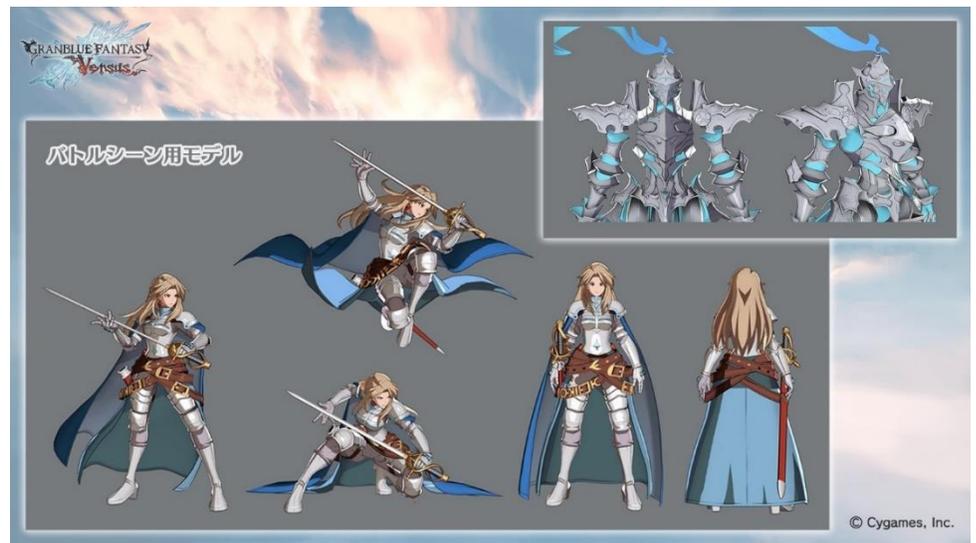


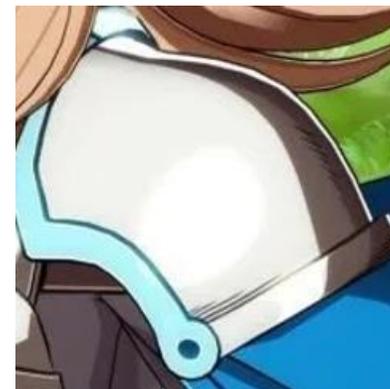
卡通

写实



立绘质感





更加丰富的材质表达



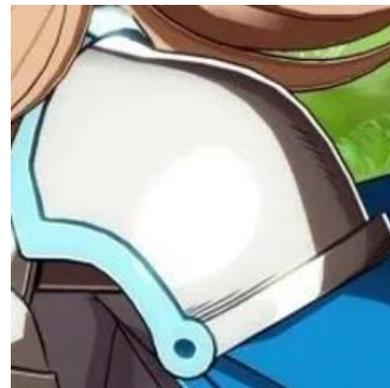
圆柱金属 清漆光 (N·V)



皮肤，单层锐利阴影



胸甲，高光Mask



肩甲，两层锐利阴影



布料，单层柔和阴影



不同材质通过组合不同的漫反射、高光、边缘光。实现了比传统赛璐璐更多的材质效果。角色更加接近立绘插画的表现。



卡通



写实



混合PBR质感

卡通渲染混合PBR



分析材质特性

哪些适合用NPR材质

哪些适合用PBR材质

通过蒙版区分不同材质



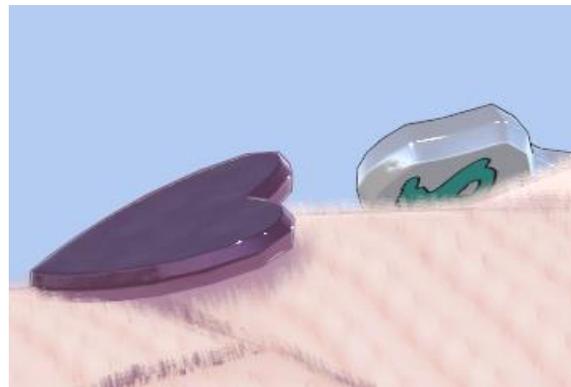
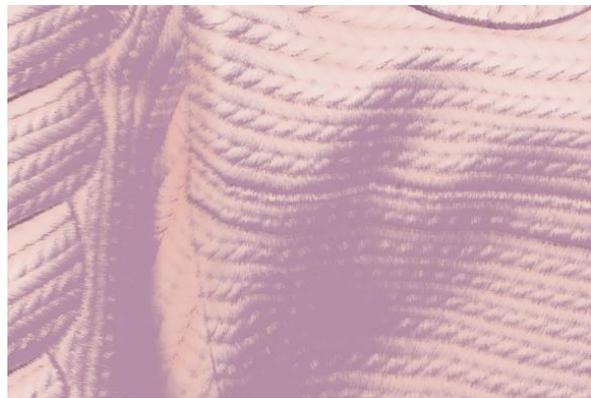
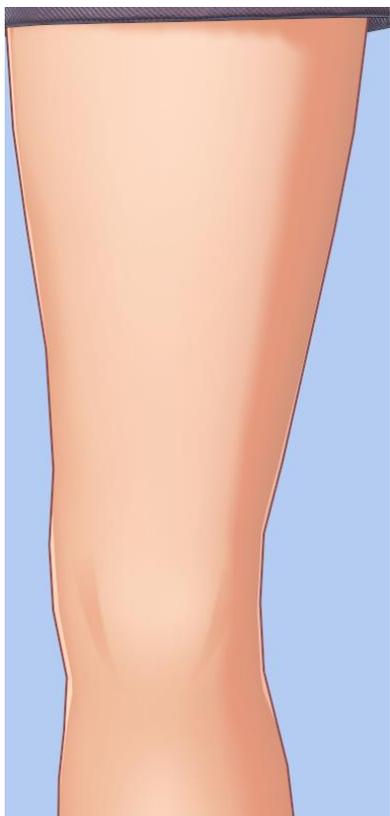
根据材质属性用蒙版图拆分

NPR部分

毛线、皮肤

PBR部分

金属、扣子



NPR材质

PBR材质

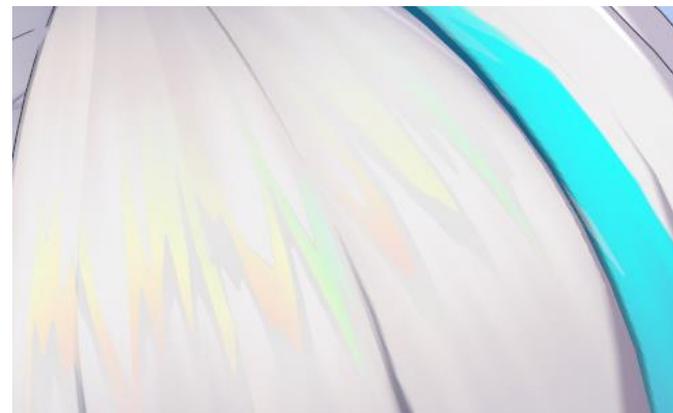
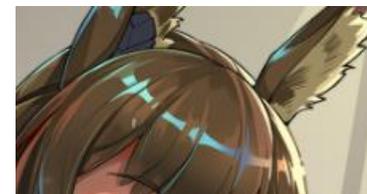
其他特殊材质



深度偏移刘海投影



MatCap眼睛



Kajiya-kay彩虹高光



构建一个卡通的光照模型。既能表现卡通感，又能表现PBR的材质感

漫反射

高光反射

边缘光

环境光

构成光照计算的4种光。其中高光对材质质感的影响最大。所以我们可以将高光部分替换为PBR的计算。其他光照保持原先卡通的光照计算。



影视化后期



ToneMapping



未应用ToneMapping



应用ACES ToneMapping

目前主流的ACES ToneMapping通常导致越亮地方的颜色越接近白色

ToneMapping



未应用ToneMapping



应用ACES ToneMapping

ACES ToneMapping也导致画面饱和度降低，和高饱和度的卡通渲染不符。

Tips : 如何修改Unity引用的Package包

Unity引用的Package包放在Library\PackageCache路径下

- ✦ com.autodesk.fbx@4.1.1
- ✦ com.unity.burst@1.5.6
- ✦ com.unity.collab-proxy@1.3.9
- ✦ com.unity.collections@1.0.0-pre.5
- ✦ com.unity.ext.nunit@1.0.6
- com.unity.formats.fbx@4.1.1
- com.unity.ide.rider@2.0.7
- com.unity.ide.visualstudio@2.0.11
- com.unity.ide.vscode@1.2.4
- com.unity.jobs@0.11.0-preview.6
- com.unity.mathematics@1.2.4
- com.unity.recorder@2.5.5
- com.unity.searcher@4.3.2
- com.unity.shadergraph@10.5.1
- com.unity.test-framework@1.1.30
- com.unity.textmeshpro@3.0.6
- com.unity.timeline@1.5.6

Tips : 如何修改Unity引用的Package包

- Assets
- com.unity.postprocessing@3.1.1
- com.unity.render-pipelines.core@10.5.1
- com.unity.render-pipelines.universal@10.5.1

修改Packages\manifest.json文件

```
"com.unity.jobs": "0.11.0-preview.6",  
"com.unity.mathematics": "1.2.4",  
"com.unity.postprocessing": "file:../com.unity.postprocessing@3.1.1",  
"com.unity.recorder": "2.5.5",  
"com.unity.render-pipelines.core" : "file:../com.unity.render-pipelines.core@10.5.1",  
"com.unity.render-pipelines.universal" : "file:../com.unity.render-pipelines.universal@10.5.1",  
"com.unity.test-framework": "1.1.30",  
"com.unity.textmeshpro": "3.0.6",  
"com.unity.timeline": "1.5.6".
```

修改SRP Core中的ACES.hisl文件

```
static const half RRT_SAT_FACTOR = 0.96 + 0.2;
```

```
static const half ODT_SAT_FACTOR = 0.93 + 0.2;
```



修改前



修改后的ACES ToneMapping



修改前



修改后的ACES ToneMapping

ToneMapping

这里对ToneMapping的修改方式不一定正确。只是说明卡通渲染对Tonemapping有一些特殊的要求。为了满足这些需求可以尝试下面的方法。

自己调整ToneMapping曲线

使用其他的ToneMapping算法，例如Filmic ToneMapping

动画摄影

在现在日本动画制作过程中。会加入很多后期效果。例如Flare和Para效果提升角色和场景的光感。这个流程也被称为动画摄影

③ 加入摄影效果

③ 摄影效果を加える



フィルター効果-1 光感
滤镜效果-1 光感



フィルター効果-2 空気感
滤镜效果-2 空气感



効果処理前



効果処理後

④ 完成画面



撮影効果を加えて完成。

Flare



原始影像



加入Flare

Flare模拟正上方的光照。让场景和角色泛白，有一种受光的感觉。可以理解成叠加了一层发光的遮罩，或者对原始颜色相加。

Flare

赛马娘中的Flare效果



加入Flare效果前



加入Flare效果后

Para



原始影像



加入Para

Flare模拟阴影。让场景和角色显黑。可以认为叠加了一层深色的遮罩，或者对原始颜色相乘。

Para



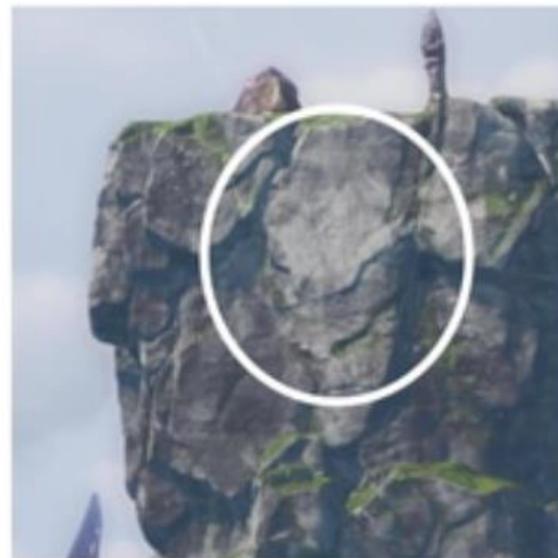
《蓝色协议》中的Flare+Para效果

SNN

SNN（对称近邻平滑滤波器）。用于去除图像中的噪声（高频信息）。使得画面的色阶减少，看起来更加卡通。在《蓝色协议》和《破晓传说》中都使用SNN滤波器来处理场景。有类似效果的还有Kuwahara滤波器



ENV用フィルター





SNN Filter



Kuwahara Filter



SNN Filter



Kuwahara Filter



总结

卡通渲染的设计规则

目的：还原美术原画效果

分析原画包含那些材质，进行分类

针对材质种类的不同，设计实现方案

综合考虑性能

卡通渲染可以考虑推进的几个方向

1.赛璐璐可以更加往2D动画的方向考虑。忽略3D结构，进行更加夸张的处理，更加还原2D动画效果

2.结合PBR材质表现的卡通渲染

3.加入动画摄影类的后期效果

作者专栏

知乎：https://www.zhihu.com/column/c_1215952152252121088

Unity社区：<https://developer.unity.cn/projects/02-cong-ling-kai-shi-de-qia-tong-xuan-ran-zhao-se-pian-1>

引用的贴图和文章

「GUILTY GEAR Xrd -SIGN-」中实现的「纯卡通动画的实时3D图形」的秘密

腾讯游戏主美：二次元卡通渲染有哪些黑科技？ TGDC 2020

动画基础知识大百科

动画摄影后期流程解密——Graphinica摄影杨晓牧采访

「BLUE PROTOCOL」は“劇場アニメクオリティ”。壮大で精緻なアニメ表現はどのように制作されているのか

『Tales of ARISE』におけるレンダリング技術と高速化

谢谢大家