色彩后期

名词解释

- **ICC Profile** 色彩配置文件。与输入输出设备有关的颜色配置文件。
- **色彩校正** 就是重新定义其ICC文件
- **色彩空间** 人为指定颜色的集合，在标准色度图上，指定三个基点，即RGB，就可以表示出三个基点内的色彩，再加上亮度的变化就能完整的表达一个色彩空间。色彩空间有无数个颜色，只是用来表明能显示的颜色的范围。
- **色域** 色域是一种对颜色进行编码的方式，如**RGBA8**，**SRGB16**等。色域是一套技术或一套系统所能产生的颜色数量的总和。从定义可以看出色域的数量可以是无数也可以是有限的。取决于这套系统或者技术再现颜色的能力。如：当你在计算机图形学领域将一幅图片定义为SRGB，8位。那能表示的颜色数量就只有1677216个。所以色域于色域之间比较颜色数量的多少是要基于条件的，如SRGB16能表示的颜色的多少就比Adobe RGB8能表示的要多。但是SRGB16能表示的范围肯定不如Adobe RGB8大。

我们即将要讨论的色彩都是色彩再现领域的，即通过一套技术将颜色表现出来。那**ICC Profile**的作用是什么呢?

它的作用就是来定义设备或者图像基于什么色域来显示，或者印刷，或者扫描。所以色彩管理实际上管理的就是色域，管理的就是ICC Profile。那么ICC profile的定义和转换，就能保证一幅图在不同设备上的视觉效果是一致的，所以无论是广色域还是标准色域的显示器，只要做好了色彩管理，他们呈现的效果就是一致的，不会有色差，或者说色差很微弱，几乎不能察觉。

色彩管理

上文说到，色彩管理就是管理ICC文件。

一，工作空间

工作空间指的是不同的颜色模型的新建文档使用的ICC配置文件。如图中的四种颜色模型：RGB,CMYK,灰色，专色。你在PS中新建一个空白文档，这个文档就使用当前颜色模型下的色域，在这个文档中的所有调色都在这个色域范围内进行。关于色彩空间的概念请看颜色那一篇。

二，色域（色彩空间）转换与位深度

通常转换色域还需要和位深度配合，当从广色域转换为标准色域时，应当先转色域，再降低位深度，这样可以保证广色域到标准色域的色彩精度，转换为标准色域后，会尽可能多的保留原本广色域和标准色域叠加区域的颜色。从标准色域转广色域同理，应该先提高位深度，再转到广色域，否则转换后的广色域会由于位深度不足，丢失颜色导致色调分离。

后期

色彩矫正（Color correction）

色彩矫正是所有颜色后期的开始，这一步需要将素材的色调与世界中的色调相匹配。使之更加准确的表达。

色彩矫正时期，可以调整曝光，对比度，白平衡。

色彩分级（color grading)

色彩分级使你的画面更有优势。 在色彩分级阶段，你可以为你影片的着色应用一种整体风格。这为你的项目注入了视觉基调，传达了你希望观众感受到的情感。在调色之前，先进行色彩矫正，以确保开始时有平衡的、自然的色彩。 在色彩矫正过后，可以尝试用色彩分级为影片奠定基调和氛围。

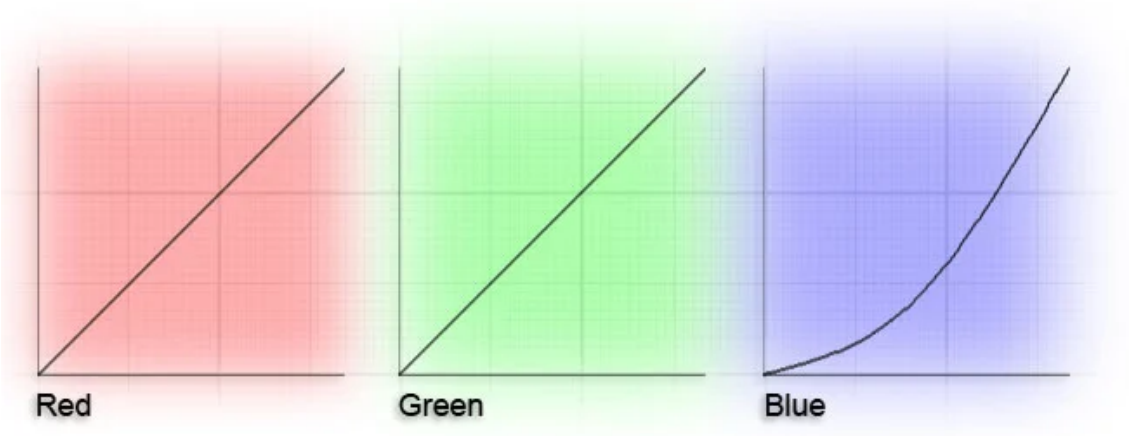
LUT

在实时渲染领域，如果每一帧都实时进行色彩矫正和色彩分级，消耗太大，所以通常使用查表的方式来进行色彩矫正和色彩分级。这个表则被成为LUT。

- 1DLUT

Red		Green		Blue	
In	Out	In	Out	In	Out
1	1	1	1	1	0.3
2	2	2	2	2	0.6
3	3	3	3	3	1.1
4	4	4	4	4	1.6
5	5	5	5	5	2.5
6	6	6	6	6	3.3
7	7	7	7	7	4.5
8	8	8	8	8	6.1
9	9	9	9	9	7.8
10	10	10	10	10	10

每个颜色值都有与之对应的输入输出，我们输入一个颜色的RGB分量，即可查找到其对应的输出的RGB的值。



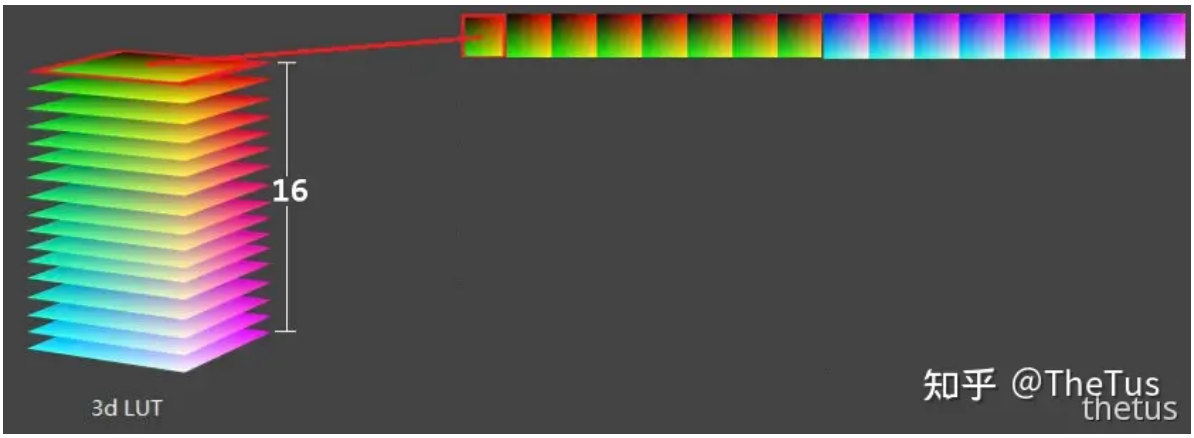
如上图所示，红色和绿色的是一条直线，这意味着对于红色和绿色来说，输入什么就会输出什么，但是蓝色是一条降低的曲线，这会让最终的图像变黄。

1DLUT犹如PS中的曲线。他不能改变诸如色调，饱和度，白平衡等内容。

- 3DLUT

原理就不写了，3DLUT可以改变色彩的饱和度，白平衡。

在实际应用中，我们不存储3D cube纹理，而是将这张cube纹理展开成一张2D纹理。（可以想象成把cube的每一层平铺为一层）



看一下Unity中如何做LUT的

在Unity中，只要开启了后处理，就会每帧产生一张Lut图，同时在UberPass中进行计算。可以在管线处设置不同的状态来调整LUT的生成格式。

```
// Color grading is always ena
{
    color = ApplyColorGrading(
}
```

生成LUT的shader有两种，第一种是**LutBuilderLdr**,第二种是**LutBuilderHdr**。而控制使用哪个进行生产的在管Post-Process下的GradingMode下。如果使用HighDynamicRange，需要勾选HDR。

颜色分级有关的设置：

- **ChannelMixer**
- **ColorAdjustments**
- **ColorCurves**
- **LifGammaGain**
- **ShadowsMidtonesHighlights**
- **SplitToning**
- **Tonemapping**
- **WhiteBalance**

LUT的Size决定了LUT图的高度，其宽度是高度的平方。

LUT图分析

LUT原图

R通道

G通道

B通道

将Unity生成的LUT图在RednerDoc打开，可以看到：**B**通道被分成了32块，每个块中，R通道从左到右为0-1，G通道从下到上为0-1。

所以如何采样一张LUT图就很明确了。首先，通过Blue确定当前块的索引。然后，计算当前快的中的Red偏移得到u，最后计算green偏移得到v。

Show code：

```
float u = floor(color.b*15.0)/15.0 * 240.0;
u = (floor(color.r*15.0)/15.0*15.0)+u;
u /= 255.0;
float v = floor(clolor.g*15.0);
v /= 15.0;
```

这样采样出来的颜色过于突兀，所以还要采样偏右边的block，然后对左边和右边的结果进行混合。采样右边的**UV**

```
u = ceil(color.b * 15.0) / 15.0 * 240.0;
    u = (ceil(color.r * 15.0) / (15.0 * 15.0)) + u;
    u /= 255.0;

    v = ceil(color.g * 15.0) / 15.0;
```

生成LUT

推导一下根据UV计算出Color值的过程，这就是Color计算Lut uv的反过程。首先写出根据color计算UV的公式。

```
float block = floor(b*(height-1));
u = block*(1/height)+r*(height-1)*(1/width)+0.5/width;
v = g*(height - 1)*(1/height)+0.5/height;
```

将其逆运算后得到：

```
 uv -= params.yz;
    real3 color;
    color.r = frac(uv.x * params.x);
    color.b = uv.x - color.r / params.x;
    color.g = uv.y;
    return color * params.w;
```

将UV转换成颜色后，再对其进行一系列的处理，如白平衡，YRGB分离，颜色过滤，色相偏移

，对比度。

参考文章

CatLikecoding Color Grading

URP ColorGradingLut 分析

什么是颜色管理