## 一、YUV定义

YUV是指亮度参量和色度参量分开表示的像素格式，其中“Y”表示明亮度（Luminance或Luma），也就是灰度值；

而“U”和“V”表示的则是色度（Chrominance或Chroma），作用是描述影像色彩及饱和度，用于指定像素的颜色。

Y: 表示明亮度(Luminance或Luma)

U和V: 色度(Chrominance或者Chroma), 作用是描述影像色彩及饱和度，用于指定像素的颜色。

YCbCr其中Y是指亮度分量，Cb指蓝色色度分量，而Cr指红色色度分量。

YCbCr 则是在世界数字组织视频标准研制过程中作为ITU - R BT.601 建议的一部分，其实是YUV经过缩放和偏移的翻版。

其中Y与YUV 中的Y含义一致，Cb，Cr 同样都指色彩，只是在表示方法上不同而已。

在YUV 家族中，YCbCr 是在计算机系统中应用最多的成员，其应用领域很广泛，JPEG、MPEG均采用此格式。

一般人们所讲的YUV大多是指YCbCr。YCbCr 有许多取样格式，如4∶4∶4，4∶2∶2，4∶1∶1 和4∶2∶0。

优点：

由于历史原因，模拟电视部分使用了YUV格式。模拟彩色电视信号能够兼容以前的黑白电视。

彩色电视信号携带着色度值（U和V）与灰度值。黑白电视忽略色度值，显示出灰度图片。

彩色电视可以提取出色度信息并将信号还原回RGB。

在今天YUV有着另一个优点。相比于亮度的变换，人眼对于色彩的变换更迟钝。那么一张图片中，可以少存放一些色度信息。

## 二、YUV采样

色度通道的采样率可以低于亮度通道，而不会显著降低感知质量：

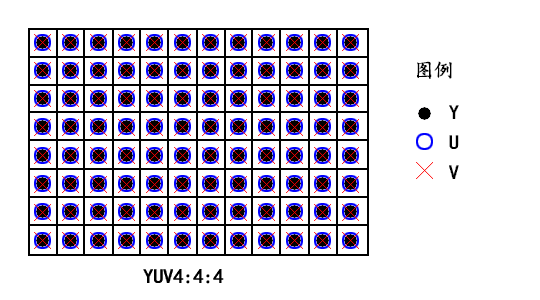
* 4:4:4 表示完全取样。
* 4:2:2 表示2:1的水平取样，垂直完全采样。
* 4:2:0 表示2:1的水平取样，垂直2:1采样。
* 4:1:1 表示4:1的水平取样，垂直完全采样。

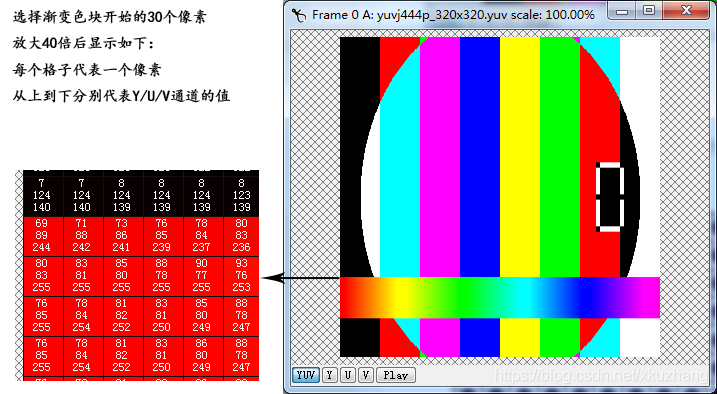
最常用Y:UV记录的比重通常1:1或2:1，Video是以YUV4:2:0的方式记录，也就是我们俗称的I420，YUV4:2:0并不是说只有U（即Cb）,V（即Cr）一定为0，而是指U：V互相援引，时见时隐，也就是说对于每一个行，只有一个U或者V分量，如果一行是4:2:0的话，下一行就是4:0:2，再下一行是4:2:0…以此类推。

其他常见的YUV格式:

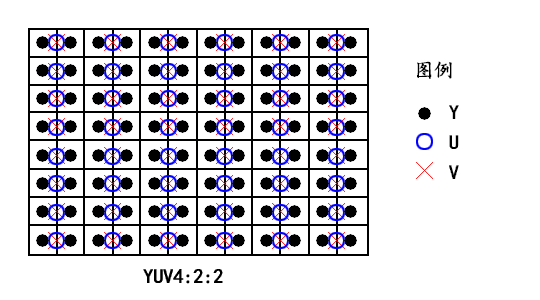
YUY2、YUYV、YVYU、UYVY、AYUV、Y41P、Y411、Y211、IF09、IYUV、YV12、YVU9、YUV411、YUV420等。

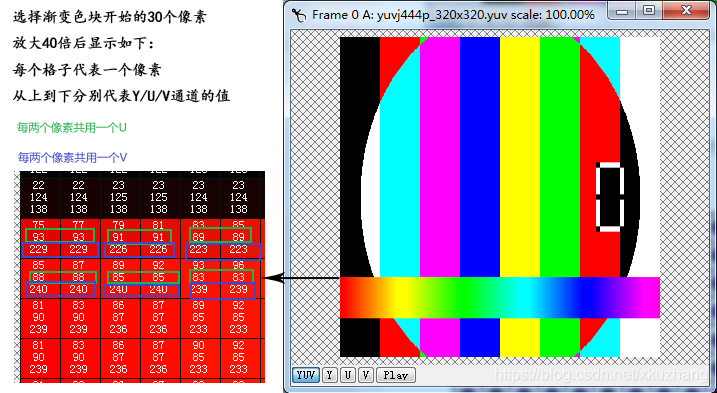
1. YUV 4:4:4，完全取样，每个像素都有独立的Y/U/V值



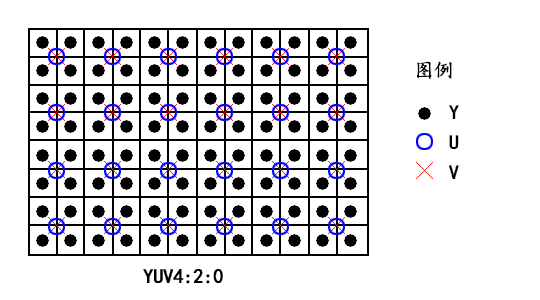


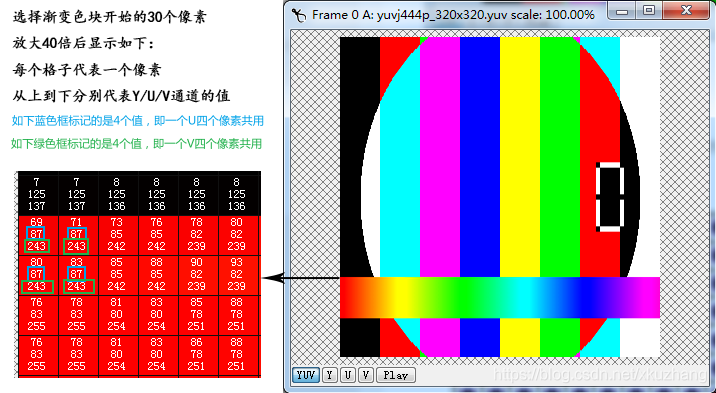
1. YUV 4:2:2，水平方向每两个Y共用一对U/V值



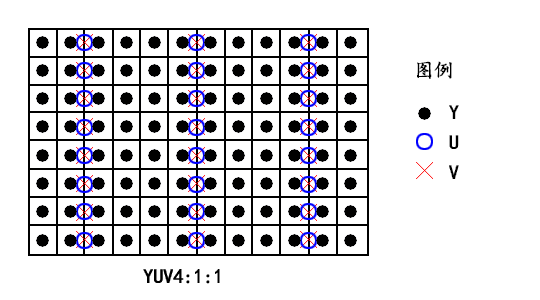


1. YUV 4:2:0，相邻的4个Y共用一对U/V





1. YUV 4:1:1，与YUV 4:2:0类似，都是四个每四个Y共用一对U/V值，区别是4:1:1是水平方向的4个Y共享一对U/V，411采样的不太常用，一般工具都不支持播放



三、planar format

可以按照 YUV 的排列方式，再次将 YUV 分成三个大类，Planar，Semi-Planar 和 Packed。

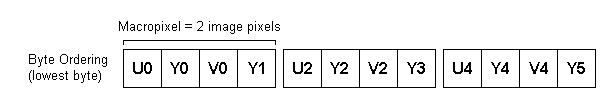
* Planar YUV 三个分量分开存放
* Semi-Planar Y 分量单独存放，UV 分量交错存放
* Packed YUV 三个分量全部交错存放

格式的名称以“ p”结尾的一般都是planar格式,如在ffmpeg 中的yuvj420p格式  
格式的名称以“ sp”结尾的一般都是semi-planar格式

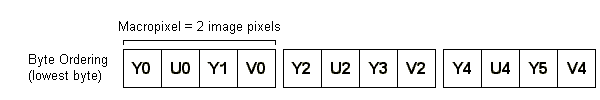
1. Packed (or Interleaved)

将YUV三个分量的素数值放在同一个阵列中，存储为单个阵列宏像素  
将每个像素点的Y/U/V连续交叉存储。  
以UYVY、YUYV为例，由两个像素的Y/U/V组成一个宏像素，依次排列

UYVY：



YUYV：

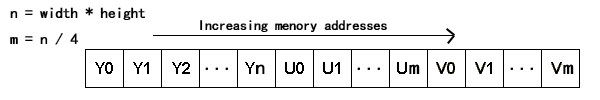


1. Planar

每个分量存储为单独的数组，最终图像是三个单独平面的整合。

以I420为例Y/U/V排列如下：

width: 图像分辨率的宽  
height: 分辨率高  
n = width \* height  
m = width \* height / 4  
Y0Y1Y2Y3Y4Y5…Yn U0U1U2U3…Um V0V1V2V3…Vm

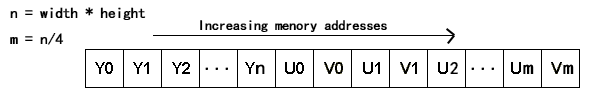


1. Semi-Planar

介于Pakced和Planar之间，即Y分量为一个阵列，UV交叉存储

NV12：

width: 图像分辨率的宽  
height: 分辨率高  
n = width \* height  
m = width \* height / 4  
Y0Y1Y2Y3Y4Y5…Yn U0V0U1V1U2V2U3V3…UmVm



* I420（属于 YUV 420 Plannar）

I420 是 YUV 420 Planar 的一种，YUV 分量分别存放，先是 w \* h 长度的 Y，后面跟 w \* h \* 0.25 长度的 U， 最后是 w \* h \* 0.25 长度的 V，总长度为 w \* h \* 1.5。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

U U U

U U U

U U U

V V V

V V V

V V V

* YV12（属于 YUV 420 Plannar）

YV12 是 YUV 420 Planar 的一种，YUV 分量分别存放，先是 w \* h 长度的 Y，后面跟 w \* h \* 0.25 长度的 V， 最后是 w \* h \* 0.25 长度的 U，总长度为 w \* h \* 1.5。与 I420 不同的是，YV12 是先 V 后 U

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

V V V

V V V

V V V

U U U

U U U

U U U

* NV12（属于 YUV 420 Semi-Planar）

NV12 是 YUV 420 Semi-Planar 的一种，Y 分量单独存放，UV 分量交错存放，UV 在排列的时候，从 U 开始。总长度为 w \* h \* 1.5。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

U V U V U V

U V U V U V

U V U V U V

* NV21（属于 YUV 420 Semi-Planar）

NV21 是 YUV 420 Semi-Planar 的一种，Y 分量单独存放，UV 分量交错存放，与 NV12 不同的是，UV 在排列的时候，从 V 开始。总长度为 w \* h \* 1.5。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

V U V U V U

V U V U V U

V U V U V U

* I422（属于 YUV 422 Plannar）

I422 是 YUV 422 Planar 的一种，YUV 分量分别存放，先是 w \* h 长度的 Y，后面跟 w \* h \* 0.5 长度的 U， 最后是 w \* h \* 0.5 长度的 V，总长度为 w \* h \* 2。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

U U U U U U

U U U U U U

U U U U U U

V V V V V V

V V V V V V

V V V V V V

* YV16（属于 YUV 422 Plannar）

YV16 是 YUV 422 Planar 的一种，YUV 分量分别存放，先是 w \* h 长度的 Y，后面跟 w \* h \* 0.5 长度的 V， 最后是 w \* h \* 0.5 长度的 U，总长度为 w \* h \* 2。与 I422 不同的是，YV16 是先 V 后 U

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

V V V V V V

V V V V V V

V V V V V V

U U U U U U

U U U U U U

U U U U U U

* NV16（属于 YUV 422 Semi-Planar）

NV16 是 YUV 422 Semi-Planar 的一种，Y 分量单独存放，UV 分量交错存放，UV 在排列的时候，从 U 开始。总长度为 w \* h \* 2。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

U V U V U V

U V U V U V

U V U V U V

U V U V U V

U V U V U V

U V U V U V

* NV61（属于 YUV 422 Semi-Planar）

NV61 是 YUV 422 Semi-Planar 的一种，Y 分量单独存放，UV 分量交错存放，UV 在排列的时候，从 V 开始。总长度为 w \* h \* 2。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

V U V U V U

V U V U V U

V U V U V U

V U V U V U

V U V U V U

V U V U V U

* YUVY（属于 YUV 422 Interleaved）

YUVY 属于 YUV 422 Interleaved 的一种。事实上，Interleaved 是属于 Packed 的，但是在 422 中，用 Interleaved 更加形象一些。在 Packed 内部，YUV 的排列顺序是 Y U V Y，两个 Y 共用一组 UV。

Y U V Y Y U V Y Y U V Y

Y U V Y Y U V Y Y U V Y

Y U V Y Y U V Y Y U V Y

Y U V Y Y U V Y Y U V Y

Y U V Y Y U V Y Y U V Y

Y U V Y Y U V Y Y U V Y

* VYUY（属于 YUV 422 Interleaved）

VYUY 属于 YUV 422 Interleaved 的一种。在 Packed 内部，YUV 的排列顺序是 VYUY，两个 Y 共用一组 UV。

V Y U Y V Y U Y V Y U Y

V Y U Y V Y U Y V Y U Y

V Y U Y V Y U Y V Y U Y

V Y U Y V Y U Y V Y U Y

V Y U Y V Y U Y V Y U Y

V Y U Y V Y U Y V Y U Y

* UYVY（属于 YUV 422 Interleaved）

UYVY 属于 YUV 422 Interleaved 的一种。在 Packed 内部，YUV 的排列顺序是 UYVY，两个 Y 共用一组 UV。

U Y V Y U Y V Y U Y V Y

U Y V Y U Y V Y U Y V Y

U Y V Y U Y V Y U Y V Y

U Y V Y U Y V Y U Y V Y

U Y V Y U Y V Y U Y V Y

U Y V Y U Y V Y U Y V Y

* I444（属于 YUV 444 Plannar）

I444 属于 YUV 444 Plannar 的一种。YUV 分量分别存放，先是 w \* h 长度的 Y，后面跟 w \* h 长度的 U， 最后是 w \* h 长度的 V，总长度为 w \* h \* 3。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

U U U U U U

U U U U U U

U U U U U U

U U U U U U

U U U U U U

U U U U U U

V V V V V V

V V V V V V

V V V V V V

V V V V V V

V V V V V V

V V V V V V

* YV24（属于 YUV 444 Plannar）

YV24 属于 YUV 444 Plannar 的一种。YUV 分量分别存放，先是 w \* h 长度的 Y，后面跟 w \* h 长度的 V， 最后是 w \* h 长度的 U，总长度为 w \* h \* 3。与 I444 不同的是，YV24 是先排列 V。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

V V V V V V

V V V V V V

V V V V V V

V V V V V V

V V V V V V

V V V V V V

U U U U U U

U U U U U U

U U U U U U

U U U U U U

U U U U U U

U U U U U U

* NV24（属于 YUV 444 Semi-Planar）

NV24 是 YUV 444 Semi-Planar 的一种，Y 分量单独存放，UV 分量交错存放，UV 在排列的时候，从 U 开始。总长度为 w \* h \* 3。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

U V U V U V U V U V U V

U V U V U V U V U V U V

U V U V U V U V U V U V

U V U V U V U V U V U V

U V U V U V U V U V U V

U V U V U V U V U V U V

* NV42（属于 YUV 444 Semi-Planar）

NV42 是 YUV 444 Semi-Planar 的一种，Y 分量单独存放，UV 分量交错存放，UV 在排列的时候，从 V 开始。总长度为 w \* h \* 3。

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

Y Y Y Y Y Y

V U V U V U V U V U V U

V U V U V U V U V U V U

V U V U V U V U V U V U

V U V U V U V U V U V U

V U V U V U V U V U V U

V U V U V U V U V U V U

* YUV 444 Packed

Y U V Y U V Y U V Y U V Y U V Y U V

Y U V Y U V Y U V Y U V Y U V Y U V

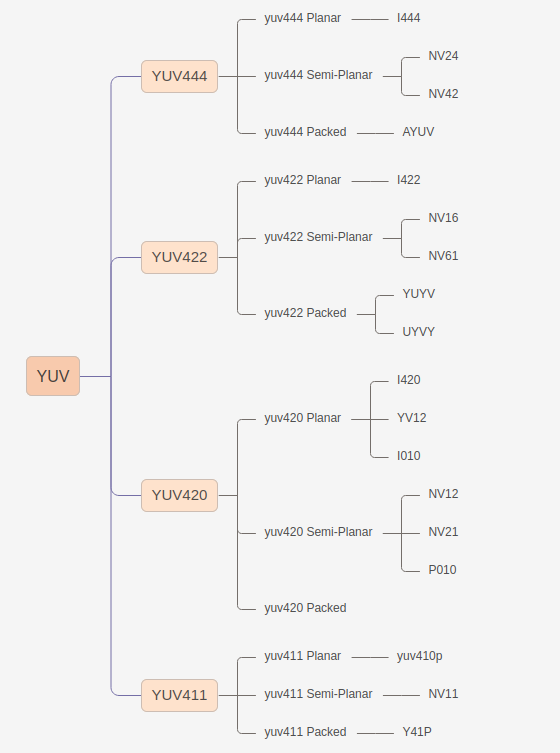
Y U V Y U V Y U V Y U V Y U V Y U V

Y U V Y U V Y U V Y U V Y U V Y U V

Y U V Y U V Y U V Y U V Y U V Y U V

Y U V Y U V Y U V Y U V Y U V Y U V

## 四、关系图



## 五、YUV和RGB转换

一般来讲，我们把YUV、Y‘UV、YCbCr, YPbPr包含在YUV颜色模型的范围内，其中Y都表示亮度，UV表示两个色度分量，但是具体的颜色模型的用途又各不相同，当然，他们与RGB的互转公式也就不相同。在电视系统发展的早期，YUV和Y’UV都是颜色信息的模拟信号编码形式，虽然Y和Y’都表示亮度，但是两种亮度的意义却天差地别，Y用来代指luminance,表示的是自然颜色的亮度，而Y‘代指luma，表示的是经过伽马压缩之后电信号的强度。在现在的计算机系统中，YUV一般用来代指YCbCr,用来表示文件的编码格式，用于数字视频的编码，而YPbPr颜色模型常常用在模拟分量视频中。所以YUV颜色模型到RGB颜色模型的转换，应该对应两种方式，分别是模拟YUV->模拟RGB、数字YUV->数字RGB。不过因为标清、高清、以及超清幅面，YUV转RGB的权重值各不相同，需要将模拟信号和数字信号再做一次幅面划分，就出现了6种转换公式。

Y、Cr、Cb信号的限制信号的限制

Y、Cr、Cb信号形式的数字编码可以表示比那些由R,、G、B信号相应范围支持的范围更大的信号值范围。正因为这样，作为电子图像生成或信号处理的结果，产生Y、CR、CB信号是可能的，尽管只是个别有效，但当转变为R、G、B时，将导致值超出范围。通过应用Y、CR、CB信号的限制，它将比在这些信号成为R、G、B形式之前一直等待来防止值超出范围更加方便和有效。限制还能在保持亮度和色调值的方法中应用，通过仅仅牺牲饱和度，来最大限度地降低主观缺陷。

计算公式推导

Kr + Kb +Kg = 1

BT.601：

Kr = 0.299

Kg = 0.587

Kb = 0.114

BT.709:

Kr = 0.2126

Kg = 0.7152

Kb = 0.0722

BT.2020:

Kr = 0.2627

Kg = 0.678

Kb = 0.0593

模拟RGB转YUV的推导公式：

Kr + Kb +Kg = 1  
Umax = Vmax = 0.5

Y = Kr \* R + Kg \* G + Kb \* B  
Pb = Umax \* (B - Y) / (1 - Kb)  
Pr = Vmax \* (R - Y) / (1 - Kr)

数字RGB转YUV：

Kr + Kb +Kg = 1

Umax = Vmax = 0.5

Y = 16 + 219 \* (Kr \* R + Kg \* G + Kb \* B) / 255

Cb = 128 + 224 \* (Umax \* (B - Y) / (1 - Kb)) / 255

Cr = 128 + 224 \* (Vmax \* (R - Y) / (1 - Kr)) / 255

模拟YUV→模拟RGB

BT.601(标清国际定义)

# RGB to YUV

Y = 0.299 \* R + 0.587 \* G + 0.114 \* B

Pb =-0.169 \* R - 0.331 \* G + 0.500 \* B

Pr = 0.500 \* R - 0.439 \* G - 0.081 \* B

# YUV to RGB

R = Y + 1.402\* Pr

G = Y - 0.344 \* Pb - 0.792\* Pr

B = Y + 1.772 \* Pb

BT.709(高清)

# RGB to YUV

Y = 0.213 \* R + 0.715 \* G + 0.072 \* B

Pb =-0.115 \* R - 0.385 \* G + 0.500 \* B

Pr = 0.500 \* R - 0.454 \* G - 0.046 \* B

# YUV to RGB

R = Y + 1.402\* Cr

G = Y - 0.344 \* Cb - 0.792\* Cr

B = Y + 1.772 \* Cb

BT.2020(超高清)

# RGB to YUV

Y = 0.2627 \* R + 0.678 \* G + 0.0593 \* B

Pb = -0.1396 \* R - 0.3604 \* G + 0.5 \* B

Pr = 0.5 \* R - 0.4598 \* G - 0.0402 \* B

数字YUV→数字RGB

BT.601

# RGB2YUV

Y = 16 + 0.257 \* R + 0.504 \* G+ 0.098 \* B

Cb = 128 - 0.148 \* R - 0.291 \* G+ 0.439 \* B

Cr = 128 + 0.439 \* R - 0.368 \* G - 0.071 \* B

# YUV2RGB

R = 1.164 \*(Y - 16) + 1.596 \*(Cr - 128)

G = 1.164 \*(Y - 16) - 0.392 \*(Cb - 128) - 0.812 \*(Cr - 128)

B = 1.164 \*(Y - 16) + 2.016 \*(Cb - 128)

BT.709

# RGB2YUV

Y = 16 + 0.183 \* R + 0.614 \* G + 0.062 \* B

Cb = 128 - 0.101 \* R - 0.339 \* G+ 0.439 \* B

Cr = 128 + 0.439 \* R - 0.399 \* G- 0.040 \* B

#YUV2RGB

R = 1.164 \*(Y - 16) + 1.792 \*(Cr - 128)

G = 1.164 \*(Y - 16) - 0.213 \*(Cb - 128) - 0.534 \*(Cr - 128)

B = 1.164 \*(Y - 16) + 2.114 \*(Cb - 128)

BT.2020

Y = 16 + 219 \* (0.2627 \* R + 0.678 \* G + 0.0593 \* B) / 255

≈ \approx ≈ 16 + 0.226 \* R + 0.582 \* G + 0.051 \* B

Cb = 128 + 224 \* (0.5 \* (B - Y) / 0.9407) / 255

Cr = 128 + 224 \* (0.5 \* (R - Y) / 0.7373) / 255

不管是YUV转RGB还是RGB转YUV都会有图像质量的损失。

对于不同位深或不同采样率的转换质量损失都好理解，但对于相同位深相同采样率的转换图像质量损失就要从计算公式来理解了，如I444和RGB24都是8bit位深未经压缩处理全采样数据，无论是RGB转YUV，还是YUV转RGB都会有少量的图像质量的损失，从转换公式来看这个由于浮点计算带来的损失，可以说是微乎其微的肉眼基本上看不出来的损失。

## 六、计算

由于YUV格式是未经压缩并且只存储了Y/U/V三个通道的数据，播放或编码必须要知道分辨率和YUV具体的格式(如nv12/i420/yuvj420ple), 因此在YUV文件命名时建议名字包含YUV格式名称和分辨率信息。  
计算文件大小或者计算帧数中的采样率和位深可由YUV格式得到。

* 1. YUV计算文件大小

已知YUV采样率、分辨率、和帧数

文件大小=宽 \* 高 \* 采样率 \* 向上取整(位深/8) \* 帧数

file size(byte) = width \* height \* rate \* Ceil(bitdepth/8) \* frames

以最常用的nv12/i420等8bit位深的420为例

文件大小 = 宽 \* 高 \* 1.5 \* 帧数

* 1. YUV计算帧数

YUV帧数的计算和文件大小的计算类似，要求已知文件大小、采样率、分辨率

YUV帧数=文件大小/(宽 \* 高 \* 采样率 \* 向上取整(位深/8))

YUV frames = file size(byte) / (width \* height \* rate \* Ceil(bitdepth/8))

sample map

| sample | rate |
| --- | --- |
| 4:4:4 | 3 |
| 4:2:2 | 2 |
| 4:2:0 | 1.5 |
| 4:1:1 | 1.5 |

当位深超过8bit时存在大小端模式的情况

big-edian(大端序)： 低地址存放高位

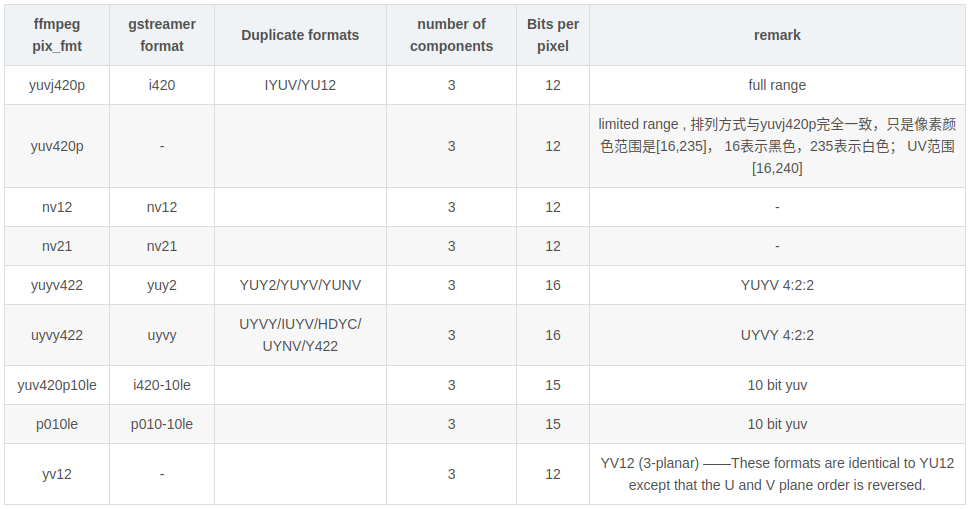
little-endian(小端序)： 低地址存放低位

格式名称以le为小端序，格式名称以be为小端序

相同名称只是后缀不同的两种格式YUV排列是完全一样的, 如yuv420p10le与yuv420p10be

## 七、一些常用格式

YUV格式名称比较多在不同的地方不同的名称有可能代表的是同一种格式，如在ffmpeg中的yuyv422与gstreamer 中的yuy2，以及其它不它的格式名称YUY2/YUYV/YUNV，都是代表同一种YUV格式。



常见的YUV格式存储排列方式

Define:

s = width \* height

n = width \* height // 4

m = width \* height // 2



其它4:1:1采样的格式一般不太常见

