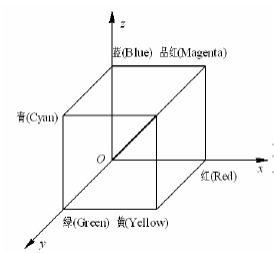
【1】RGB

    RGB 在计算机领域有着举足轻重的地位, 由于色彩显示器使用红、绿、蓝 3

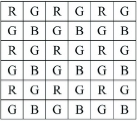
色来产生需要的颜色, 所以被广泛用于计算机图形和成像之中。然而, 在处理现实世界的图像时,RGB 并非很有效, 因为它对所有色彩都用等长像素点的 R、G、B 3 色加以合成。这就使得每个像素在 R、G、B 3 个成分上拥有相同的像素深度和显示分辨率。而且, 处理 RGB 色彩空间的图像也不是最有效的。



RGB 有一个问题是数据量太大，采集里如果每一个相素都是 3 个 BYTE 的

数据（R G B），是很没有必要的，于是有人就想去了两个相素点共用一种颜色的

算法。这个算法就是著名的 bayer 算法的初始。 R 感应红光，G 感应绿光，B 感应蓝光。在 Bayer 格式的图像中，图像一半的像素分配给 G 分量，而 R 和 B 分量则占图像的另一半像素。因为 G 分量是 R、B 分量的两倍，所以如果 G 分量采用好的插值方法，不仅可以提高 G 分量的质量，还可以提高 R 和 B 分量的质量。

Bayer 格式示意图：

【2】YUV

  在 YUV 空间中，每一个颜色有一个亮度信号 Y，和两个色度信号 U 和 V。亮度信号是强度的感觉，它和色度信号断开，这样的话强度就可以在不影响颜色的情况下改变。

  YUV 使用RGB的信息，但它从全彩色图像中产生一个黑白图像，然后提取出三个主要的颜色变成两个额外的信号来描述颜色。把这三个信号组合回来就可以产生一个全彩色图像。

  Y 通道描述 Luma 信号，它与亮度信号有一点点不同，值的范围介于亮和暗之间。 Luma 是黑白电视可以看到的信号。U (Cb) 和 V (Cr) 通道从红 (U) 和蓝 (V) 中提取亮度值来减少颜色信息量。这些值可以从新组合来决定红，绿和蓝的混合信号。

    YUV和RGB的转换:  
       Y = 0.299 R + 0.587 G + 0.114 B  
       U = -0.1687 R - 0.3313 G + 0.5 B + 128  
       V = 0.5 R - 0.4187 G - 0.0813 B + 128

       R = Y + 1.402 (V-128)

       G= Y - 0.34414 (U-128) - 0.71414 (V-128)

       B= Y + 1.772 (U-128)

【3】YCbCr

   YCbCr 是在世界数字组织视频标准研制过程中作为ITU - R BT1601 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与RGB的相互转换

　　   Y=0.299R+0.587G+0.114B

　　   Cb=0.564(B-Y)

　　   Cr=0.713(R-Y)

　   　R=Y+1.402Cr

　  　 G=Y-0.344Cb-0.714Cr

     B=Y+1.772Cb

YUV（YCbCr）采样格式：

  主要的采样格式有YCbCr 4:2:0、YCbCr 4:2:2、YCbCr 4:1:1和 YCbCr 4:4:4。其中YCbCr 4:1:1 比较常用，其含义为：每个点保存一个 8bit 的亮度值(也就是Y值), 每 2 x 2 个点保存一个 Cr和Cb值, 图像在肉眼中的感觉不会起太大的变化。所以, 原来用 RGB(R,G,B 都是 8bit unsigned) 模型, 每个点需要 8x3=24 bits， 而现在仅需要 8+(8/4)+(8/4)=12bits, 平均每个点占12bits。这样就把图像的数据压缩了一半。

上边仅给出了理论上的示例，在实际数据存储中是有可能是不同的，下面给出几种具体的存储形式：

（1） YUV 4:4:4

　　YUV三个信道的抽样率相同，因此在生成的图像里，每个象素的三个分量信息完整（每个分量通常8比特），经过8比特量化之后，未经压缩的每个像素占用3个字节。

　　下面的四个像素为: [Y0 U0 V0] [Y1 U1 V1] [Y2 U2 V2] [Y3 U3 V3]

　　存放的码流为: Y0 U0 V0 Y1 U1 V1 Y2 U2 V2 Y3 U3 V3

（2） YUV 4:2:2

　　每个色差信道的抽样率是亮度信道的一半，所以水平方向的色度抽样率只是4:4:4的一半。对非压缩的8比特量化的图像来说，每个由两个水平方向相邻的像素组成的宏像素需要占用4字节内存(例如下面映射出的前两个像素点只需要Y0、Y1、U0、V1四个字节)。

　　下面的四个像素为: [Y0 U0 V0] [Y1 U1 V1] [Y2 U2 V2] [Y3 U3 V3]

　　存放的码流为: Y0 U0 Y1 V1 Y2 U2 Y3 V3

　　映射出像素点为：[Y0 U0 V1] [Y1 U0 V1] [Y2 U2 V3] [Y3 U2 V3]

（3） YUV 4:1:1

　　4:1:1的色度抽样，是在水平方向上对色度进行4:1抽样。对于低端用户和消费类产品这仍然是可以接受的。对非压缩的8比特量化的视频来说，每个由4个水平方向相邻的像素组成的宏像素需要占用6字节内存

　　下面的四个像素为: [Y0 U0 V0] [Y1 U1 V1] [Y2 U2 V2] [Y3 U3 V3]

　　存放的码流为: Y0 U0 Y1 Y2 V2 Y3

　　映射出像素点为：[Y0 U0 V2] [Y1 U0 V2] [Y2 U0 V2] [Y3 U0 V2]

（4）YUV4:2:0

　　4:2:0并不意味着只有Y,Cb而没有Cr分量。它指得是对每行扫描线来说，只有一种色度分量以2:1的抽样率存储。相邻的扫描行存储不同的色度分量，也就是说，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。对每个色度分量来说，水平方向和竖直方向的抽样率都是2:1，所以可以说色度的抽样率是4:1。对非压缩的8比特量化的视频来说，每个由2x2个2行2列相邻的像素组成的宏像素需要占用6字节内存。

下面八个像素为：

  [Y0 U0 V0] [Y1 U1 V1] [Y2 U2 V2] [Y3 U3 V3]

　　[Y5 U5 V5] [Y6 U6 V6] [Y7U7 V7] [Y8 U8 V8]

存放的码流为：

  Y0 U0 Y1 Y2 U2 Y3

　　Y5 V5 Y6 Y7 V7 Y8

映射出的像素点为：

  [Y0 U0 V5] [Y1 U0 V5] [Y2 U2 V7] [Y3 U2 V7]

  [Y5 U0 V5] [Y6 U0 V5] [Y7U2 V7] [Y8 U2 V7]

  之前对于4:4:4、4:2:2、4:2:0一直没有清晰地认识，而上面的解释是我见到的最直观的，使我一目了然，豁然开朗。