Multimedia-Lecture-Two

COLOR SYSTEMS

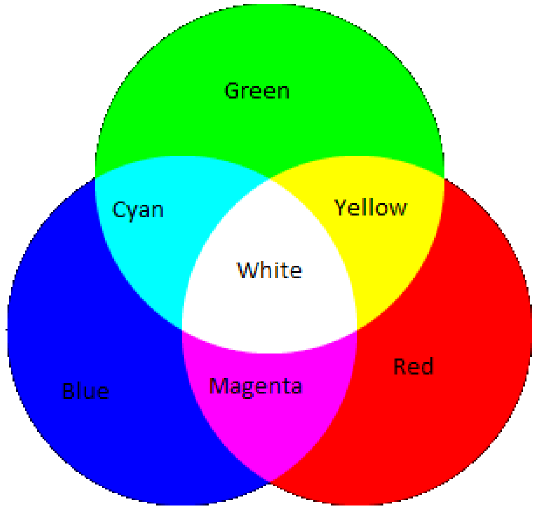
Eng. Noor AL-Hakim

Color Space

When colors need to be used in digital media like cameras and laptops, colors need to be presented in numbers. Because digital media can only understand numbers. Therefore, color space is a set of rules that allows describing colors with numbers.

RGB Color Space

The RGB color space represents images as an m-by-n-by-3 numeric array whose elements specify the intensity values of the red, green, and blue color channels.



To Extract RGB Components follow these commands:

```
// Create blank images for each components

// Extract components pixel by pixel

// Get RGB color of current pixel

// Extract components

// Create colors for each component

// Set the corresponding pixel in each component image

// Save component images
```

```
Bitmap rgbImage = new Bitmap(imagePath);
Bitmap redComponentImage = new Bitmap(rgbImage.Width, rgbImage.Height);
Bitmap greenComponentImage = new Bitmap(rgbImage.Width, rgbImage.Height);
Bitmap blueComponentImage = new Bitmap(rgbImage.Width, rgbImage.Height);
// Extract components pixel by pixel
for (int y = 0; y < rgbImage.Height; y++)
     {
       for (int x = 0; x < rgbImage.Width; x++)
         {
           // Get RGB color of current pixel
           Color pixelColor = rgbImage.GetPixel(x, y);
           // Extract components
           int redComponent = pixelColor.R;
           int greenComponent = pixelColor.G;
           int blueComponent = pixelColor.B;
           // Create colors for each component
           Color redColor = Color.FromArgb(redComponent, 0, 0);
           Color greenColor = Color.FromArgb(0, greenComponent, 0);
           Color blueColor = Color.FromArgb(0, 0, blueComponent);
// Set the corresponding pixel in each component image
redComponentImage.SetPixel(x, y, redColor);

greenComponentImage.SetPixel(x, y, greenColor);

blueComponentImage.SetPixel(x, y, blueColor);

   }
}

// Save component images
redComponentImage.Save("red_component.jpg");
greenComponentImage.Save("green_component.jpg");
blueComponentImage.Save("blue_component.jpg");
```

CMY Color Space

A CMY color space uses cyan, magenta, and yellow (CMY) as its primary colors. Red, green, and blue are the secondary colors.

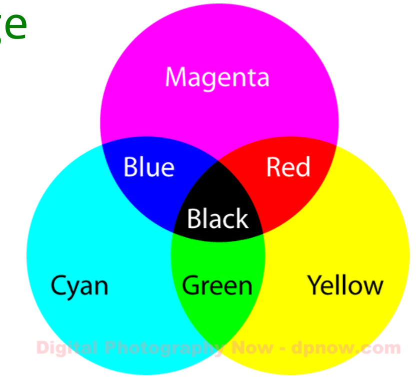
$$W = (0, 0, 0) \qquad B = (1, 1, 1)$$

Conversion from RGB to CMY

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = 1 - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

To Convert RGB Image to CMY:

```
// Create a blank image for the CMY image

// Convert RGB to CMY

// Get RGB color of current pixel

// Calculate CMY values

// Create CMY color using the calculated values

// Set the corresponding pixel in the CMY image

// Save component images
```

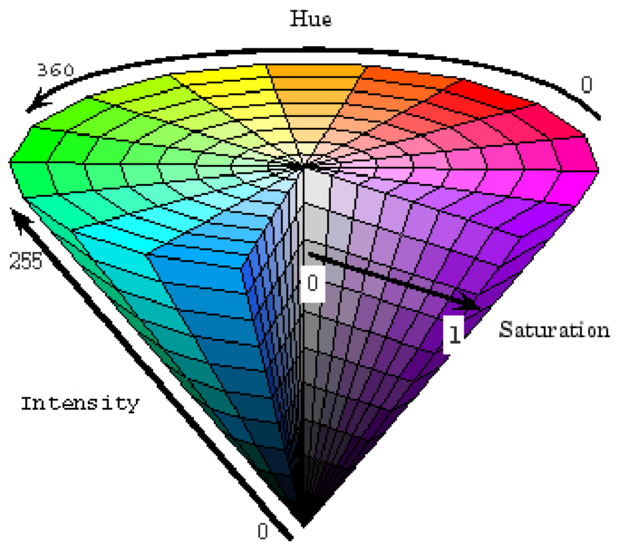


```
// Convert RGB to CMY
for (int i = 0; i < rgbImage.Height; i++)
      {
        for (int j = 0; j < rgbImage.Width; j++)
           {
             // Get RGB color of current pixel
             Color pixelColor = rgbImage.GetPixel(j, i);
             // Calculate CMY values
             int c = 255 - pixelColor.R; // Cyan
             int m = 255 - pixelColor.G; // Magenta
             int y = 255 - pixelColor.B; // Yellow
             // Create CMY color using the calculated values
             Color cmyColor = Color.FromArgb(c, m, y);
             // Set the corresponding pixel in the CMY image
             cmyImage.SetPixel(i, j, cmyColor);
                }
            }
// Save the CMY image
cmyImage.Save("cmy_image.jpg");
```

Try it using Lockbit Method

HSV Color Space

The Hue-Saturation-Value model is oriented towards the user/artist.



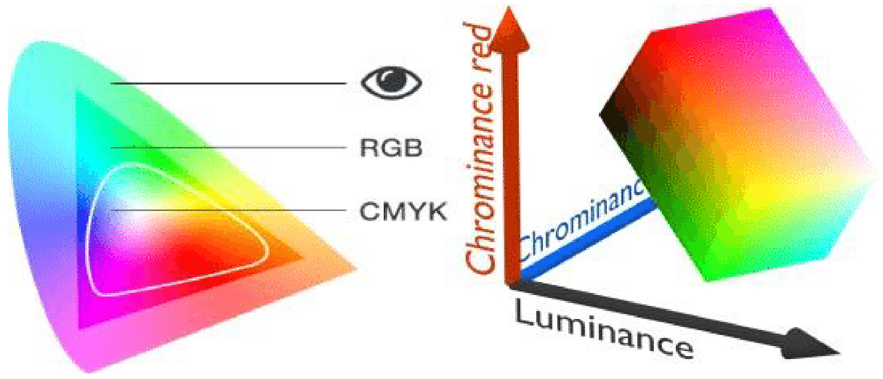
To Convert RGB Image to HSV:

```
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Structure;

// Load an RGB image
Mat rgbImage = CvInvoke.Imread("input.jpg");

// Convert RGB to HSV
Mat hsvImage = new Mat();
CvInvoke.CvtColor(rgbImage, hsvImage,
ColorConversion.Bgr2Hsv);

CvInvoke.Imshow("HSV Image", hsvImage);
CvInvoke.WaitKey(0);
```

Be Attention!!
You should use Windows Form Application when using EmguCV

YCbCr Color Space

Y is the luma component of the color. Luma component is the brightness of the color. That means the light intensity of the color. The human eye is more sensitive to this component.

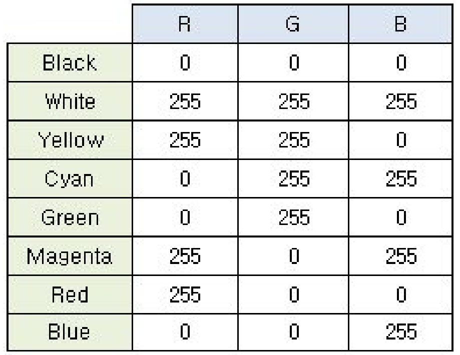


To Convert RGB Image to YCbCr:

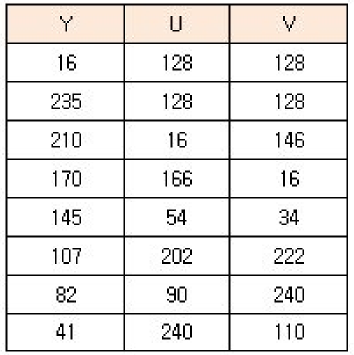
```
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Structure;

// Load an RGB image
Mat rgbImage = CvInvoke.Imread("input.jpg");

// Convert RGB to YCbCr
Mat ycbcrImage = new Mat();
CvInvoke.CvtColor(rgbImage, ycbcrImage,
ColorConversion.Bgr2YCrCb);
CvInvoke.Imshow("YCbCr Image", ycbcrImage);
CvInvoke.WaitKey(0);
```

YUV Color Space

YUV color space is a bit unusual. The Y component determines the brightnes s of the color (referred to as luminance or luma), while the U and V components determine the color itself (the chroma).



	R	G	B
Black	0	0	0
White	255	255	255
Yellow	255	255	0
Cyan	0	255	255
Green	0	255	0
Magenta	255	0	255
Red	255	0	0
Blue	0	0	255



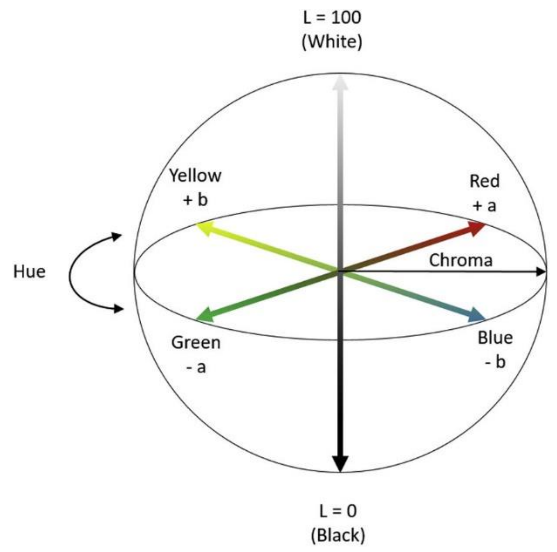
Y	U	V
16	128	128
235	128	128
210	16	146
170	166	16
145	54	34
107	202	222
82	90	240
41	240	110

To Convert RGB Image to YUV:

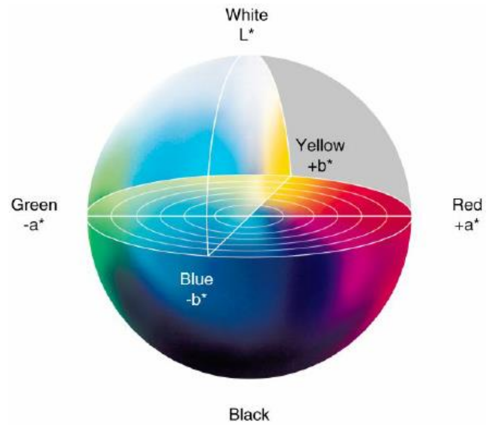
```
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Structure;

// Load an RGB image
Mat rgbImage = CvInvoke.Imread("input.jpg");

// Convert RGB to YUV
Mat yuvImage = new Mat();
CvInvoke.CvtColor(rgbImage, yuvImage,
ColorConversion.Bgr2Yuv);
CvInvoke.WaitKey(0);
```

L*a*b Color Space

A CMY color space uses cyan, magenta, and yellow (CMY) as its primary colors. Red, green, and blue are the secondary colors.



To Convert RGB Image to L*a*b:



```
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Structure;

// Load an RGB image
Mat rgbImage = CvInvoke.Imread("input.jpg");

// Convert RGB to Lab
Mat labImage = new Mat();
CvInvoke.CvtColor(rgbImage, labImage,
ColorConversion.Bgr2Lab);
CvInvoke.WaitKey(0);
```

Exercise:

I. Read an image, then:

- Delete the green matrix then merge the photo again.
- Show the merged image.

Try to Use Lockbit Method to Read and Write Pixels

That’s All