Digital Image Processing

Color image processing

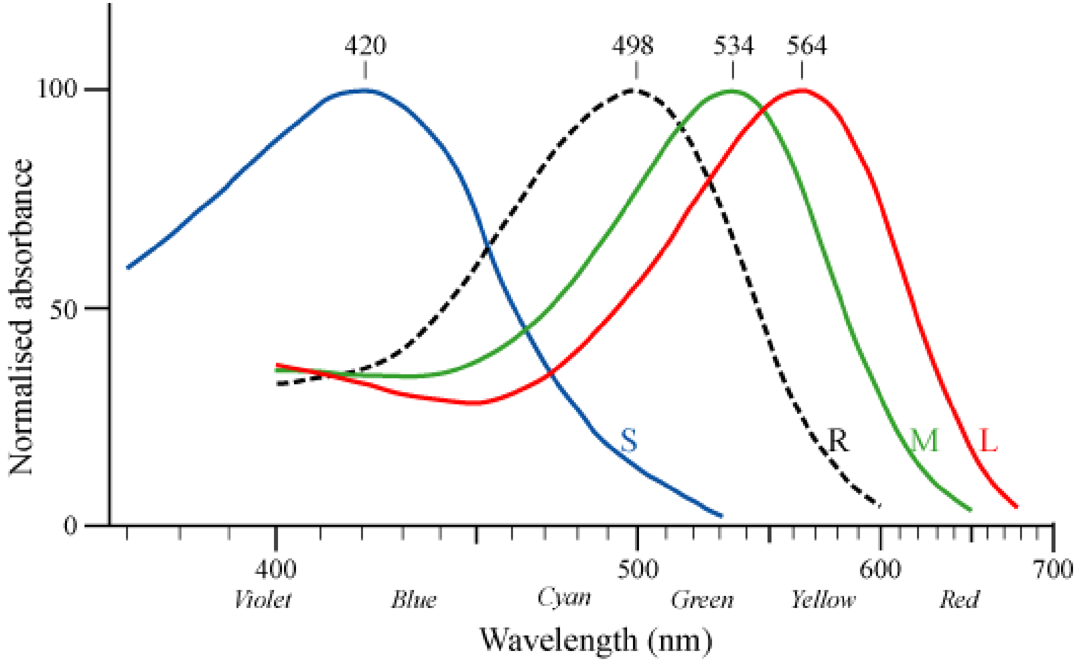
Introduction

- Color perception is a psychophysical phenomenon that combines two main components:

1. The physical properties of light sources (usually expressed by their spectral power distribution, SPD) and surfaces (e.g., their absorption and reflectance capabilities).
2. The physiological and psychological aspects of the human visual system (HVS).

Color fundamentals

- It all starts with light in the 400-700 nm range of the EM spectrum.
- As light reaches the retina, it is encoded by specialized photoreceptors (cones).

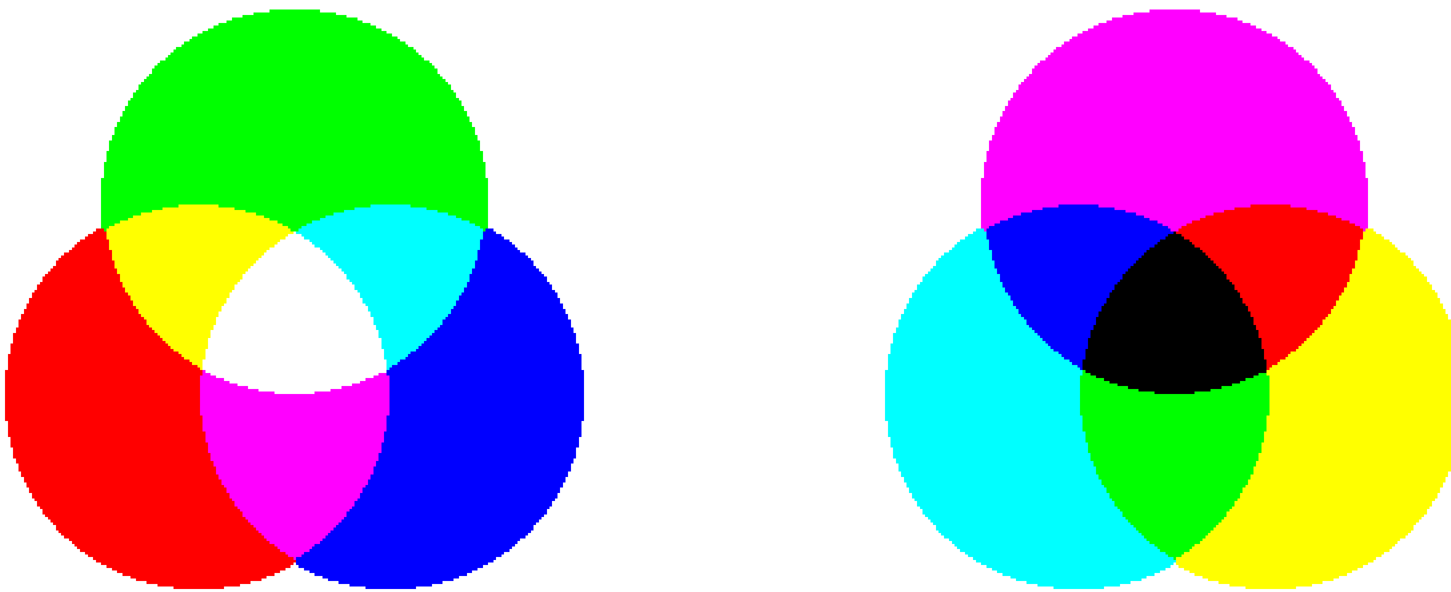


Color fundamentals

- Chromatic light can be described by:
  - **Intensity** (or radiance): the total amount of energy that flows from the light source, measured in watts (W).
  - **Luminance**: a measure of the amount of information an observer perceives from a light source, measured in lumen (lm).
  - **Brightness**: the *subjective* perception of (achromatic) luminous intensity.

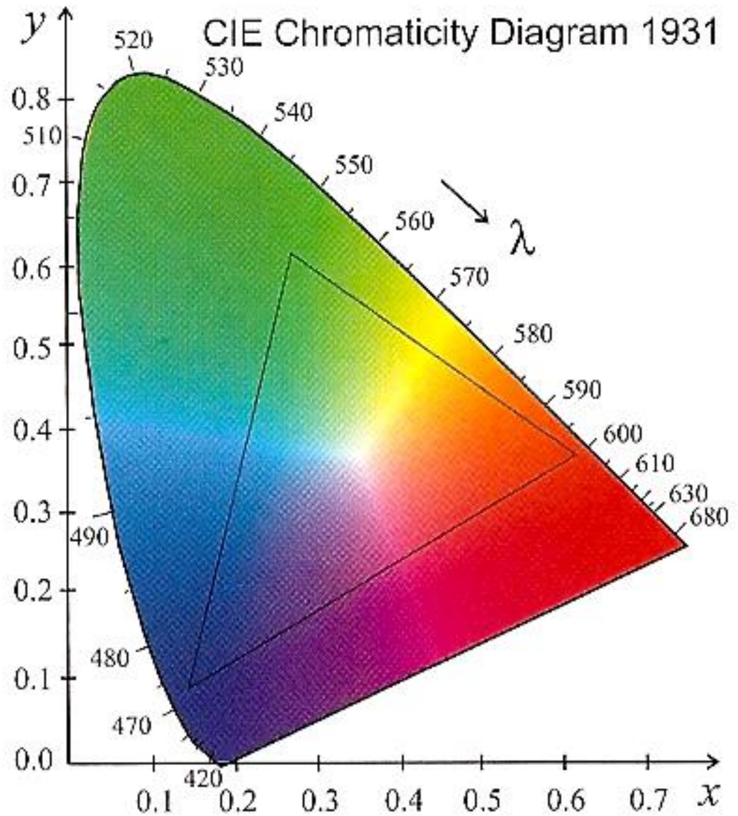
Color fundamentals

- Primary and secondary colors



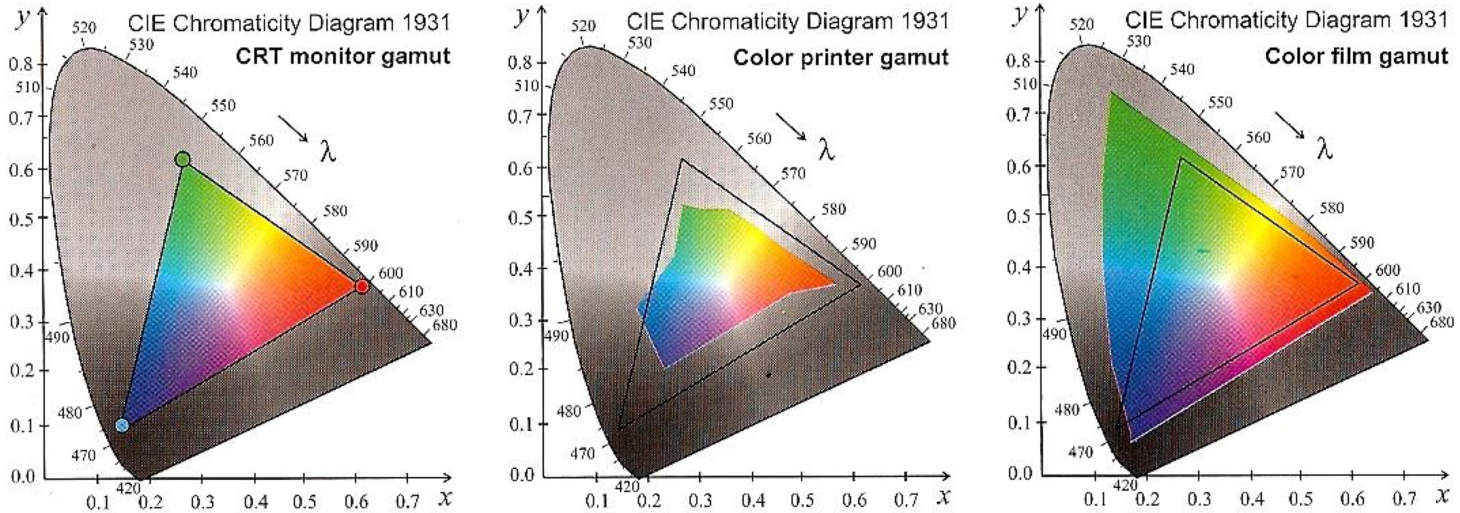
Color fundamentals

- The CIE XYZ chromaticity diagram



Color fundamentals

- Using the CIE XYZ to represent color gamut of different devices.



(a) CRT monitor (b) printer (c) film

Color models

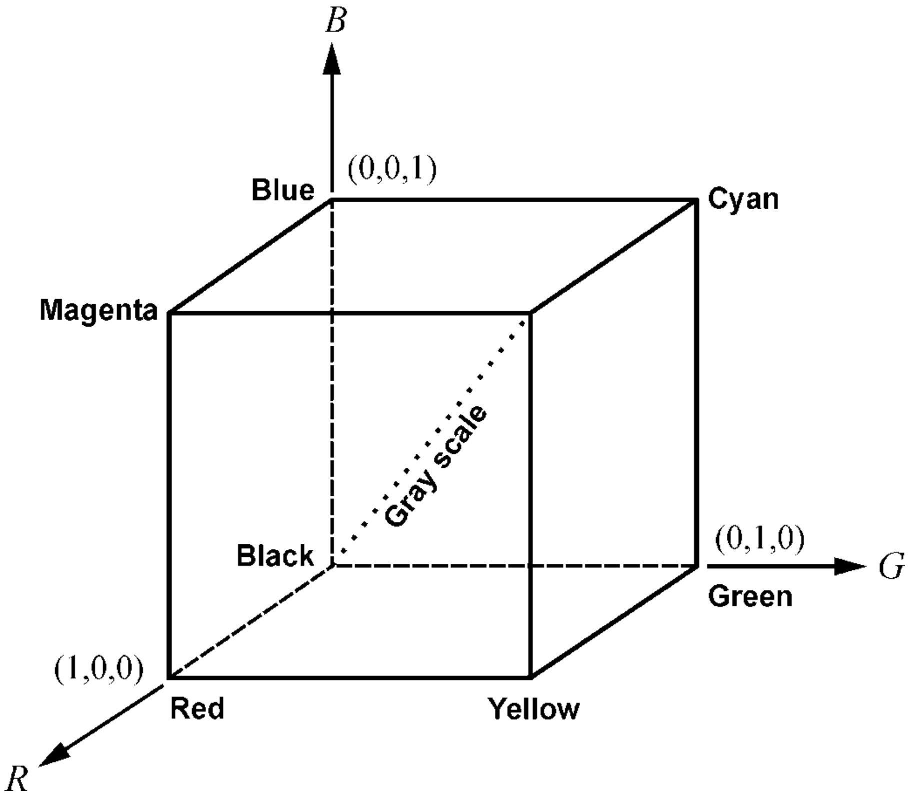
- A color model (also called color space or color system) is a specification of a coordinate system and a subspace within that system where each color is represented by a single point.

- OPTIONAL: For an interactive exploration of color models, check out the ImageJ 3D Color Inspector plugin
  - [http://rsbweb.nih.gov/ij/plugins/color-inspector.html]

Representative color spaces

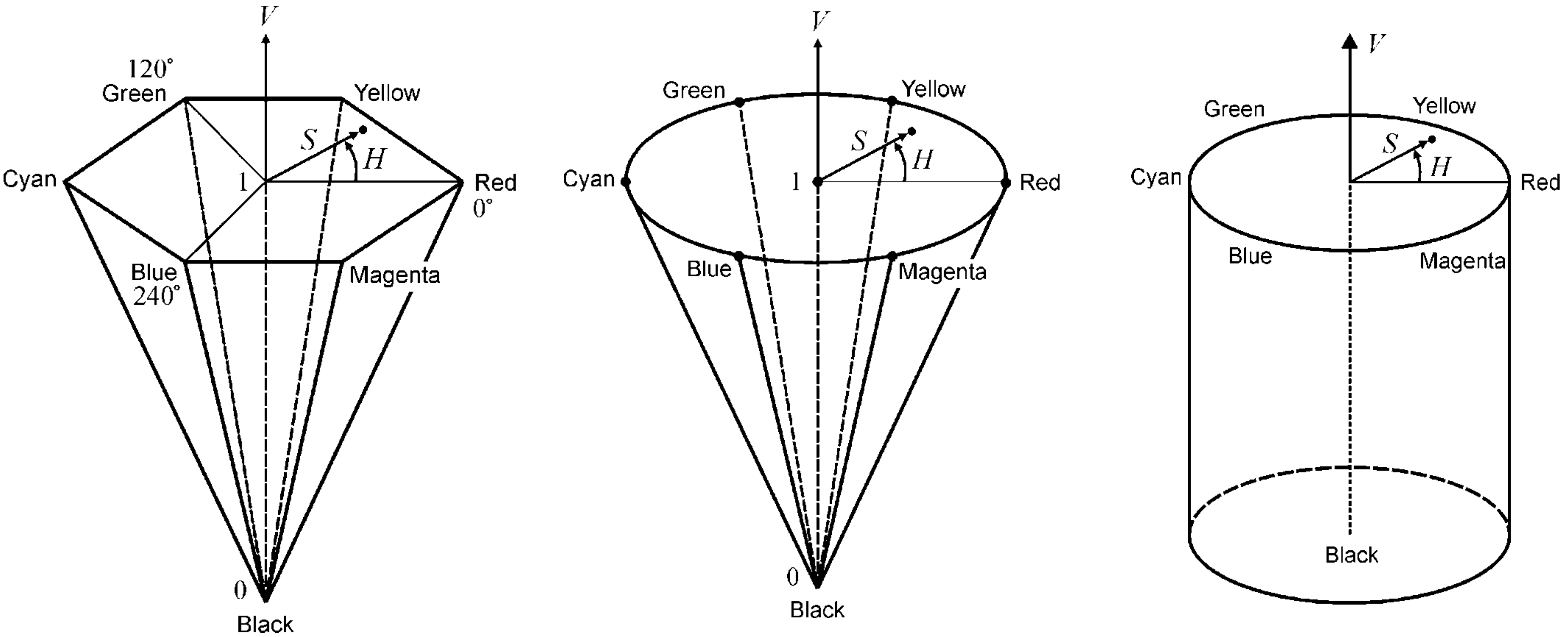
- RGB
- YUV, YIQ, YCbCr
- HSB, HSV, HSL
- HMMD
- CIEXYZ
- L*a*b* (CIELAB)
- L*u*v* (CIELUV)

RGB color space



Color name	R	G	B
Black	0	0	0
Blue	0	0	1
Green	0	1	0
Cyan	0	1	1
Red	1	0	0
Magenta	1	0	1
Yellow	1	1	0
White	1	1	1

HSV color space



In MATLAB: **`rgb2hsv`** and **`hsv2rgb`**

YIQ (NTSC) color space

- In MATLAB: **`rgb2ntsc`** and **`ntsc2rgb`**

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

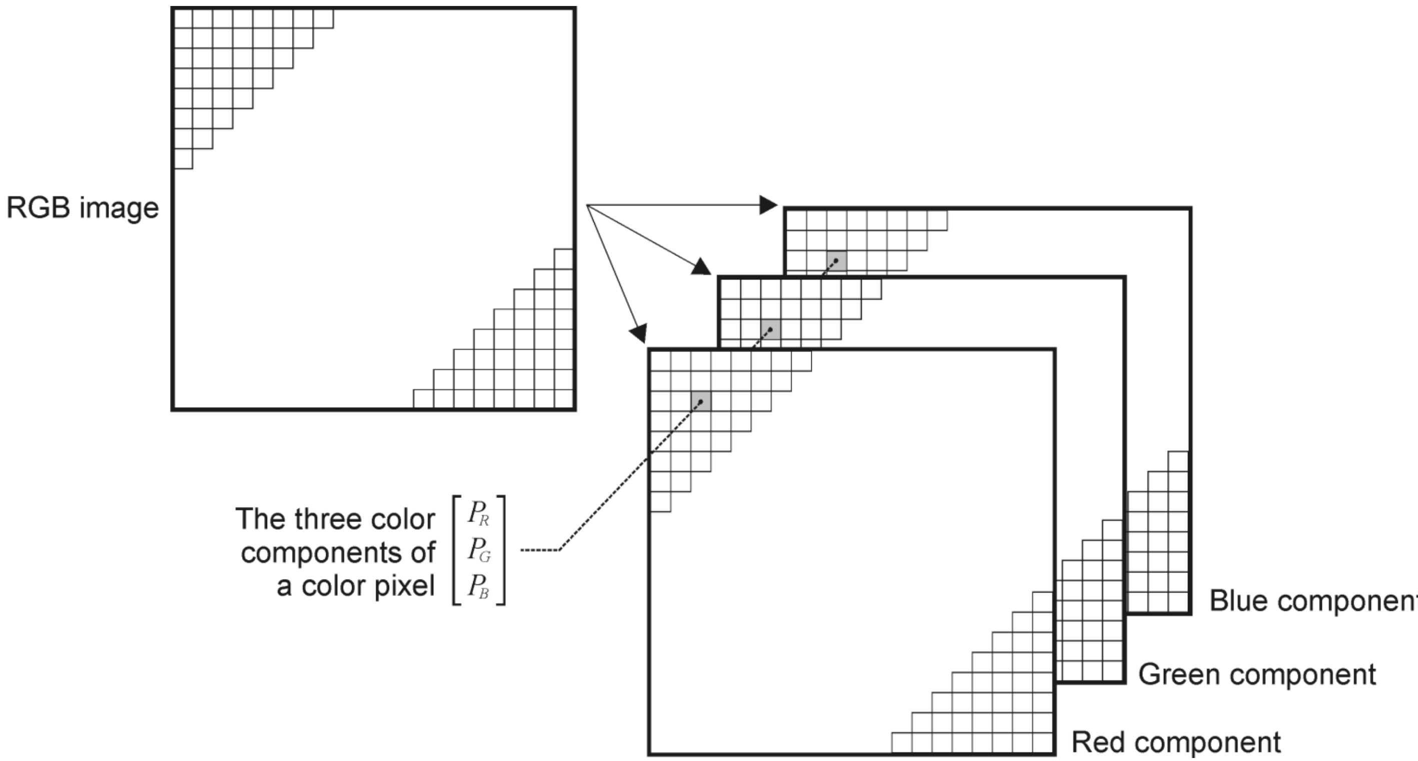
YCbCr (digital video) color space

- In MATLAB: **`rgb2ycbcr`** and **`ycbcr2rgb`**

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Color images in MATLAB

- RGB



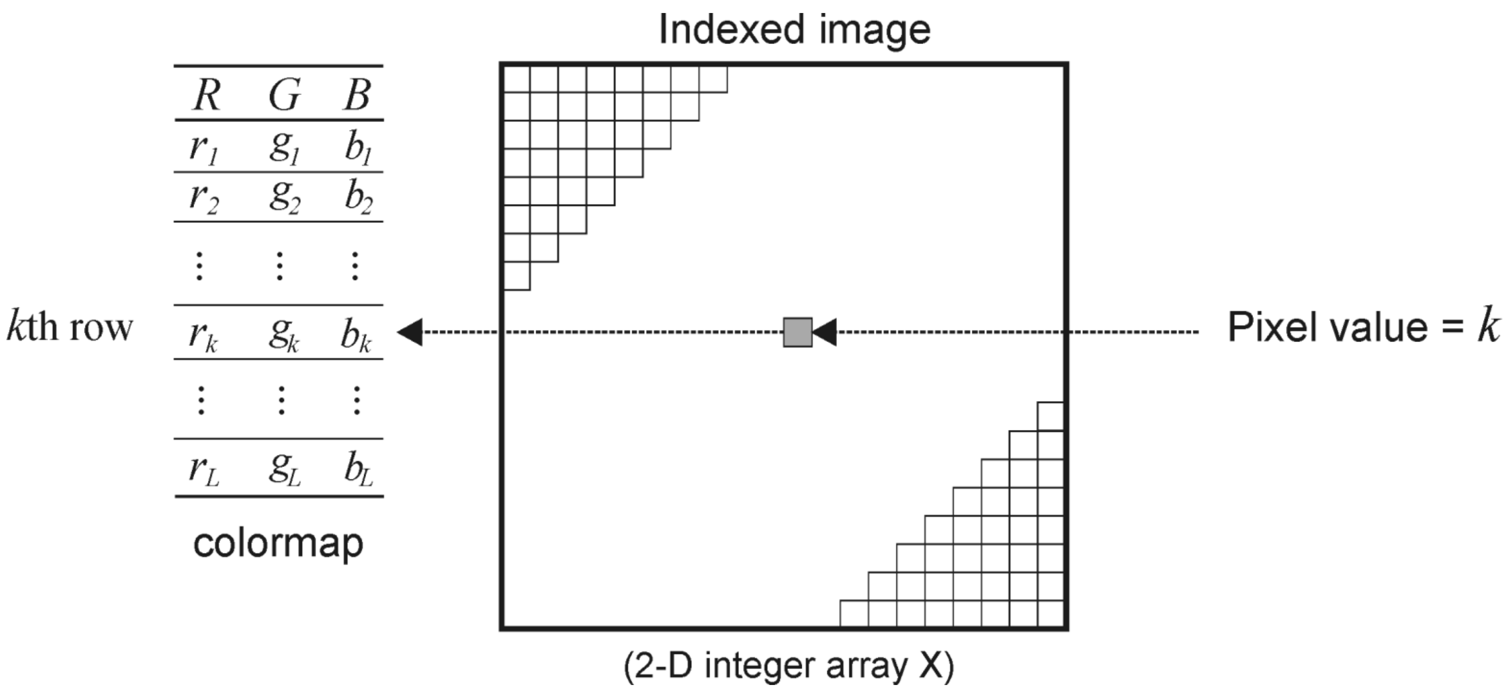
Color images in MATLAB

- RGB

```
I = imread('peppers.png');
size(I)
class(I)
subplot(2,2,1), imshow(I), title('Color image (RGB)')
subplot(2,2,2), imshow(I(:,:,1)), title('Red component')
subplot(2,2,3), imshow(I(:,:,2)), title('Green component')
subplot(2,2,4), imshow(I(:,:,3)), title('Blue component')
```

Color images in MATLAB

- Indexed



Color images in MATLAB

- Indexed

```
load clown
size(X)
class(X)
size(map)
class(map)
imshow(X,map), title('Color (Indexed)')
```

Color maps in MATLAB

Name	Description
`hsv`	Hue-saturation-value color map.
`hot`	Black-red-yellow-white color map.
`gray`	Linear gray-scale color map.
`bone`	Gray-scale with tinge of blue color map.
`copper`	Linear copper-tone color map.
`pink`	Pastel shades of pink color map.
`white`	All white color map.
`flag`	Alternating red, white, blue, and black color map.
`lines`	Color map with the line colors.
`colorcube`	Enhanced color-cube color map.
`vga`	Windows colormap for 16 colors.
`jet`	Variant of HSV.
`prism`	Prism color map.
`cool`	Shades of cyan and magenta color map.
`autumn`	Shades of red and yellow color map.
`spring`	Shades of magenta and yellow color map.
`winter`	Shades of blue and green color map.
`summer`	Shades of green and yellow color map.

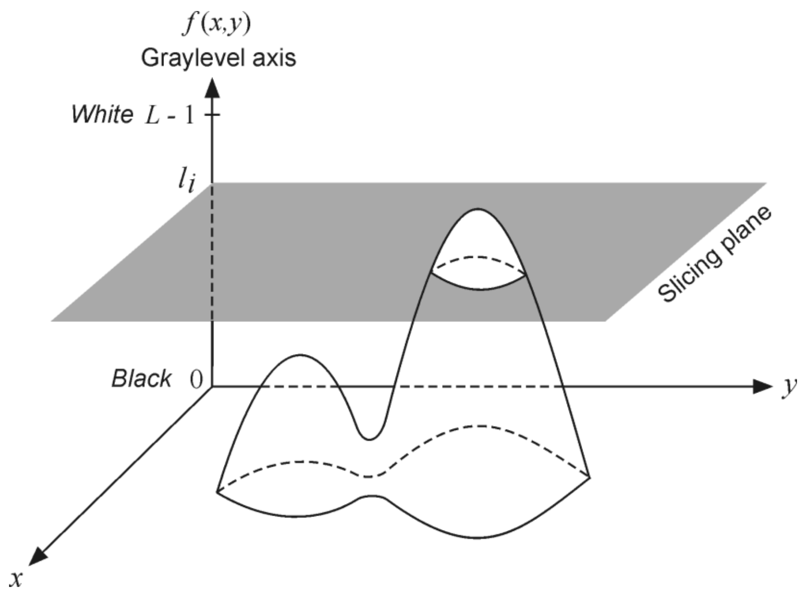
Pseudocolor image processing

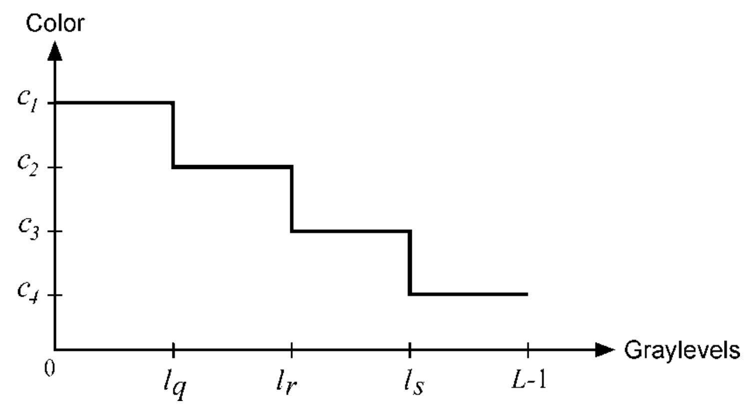
- Goal: to enhance a monochrome image for human viewing purposes.
- Rationale: subtle variations of gray levels may very often mask or hide regions of interest within an image.
- Basic idea: assigning colors to gray values based on a specified criterion.

- Simplest and best-known technique: intensity (or density) slicing.
  - This method defines slicing planes that slice the original image in different points above the xy plane. Each side of the plane will be assigned a different color.
  - The idea can be easily extended to M planes and M+1 intervals.
  - In MATLAB: **`grayslice`**

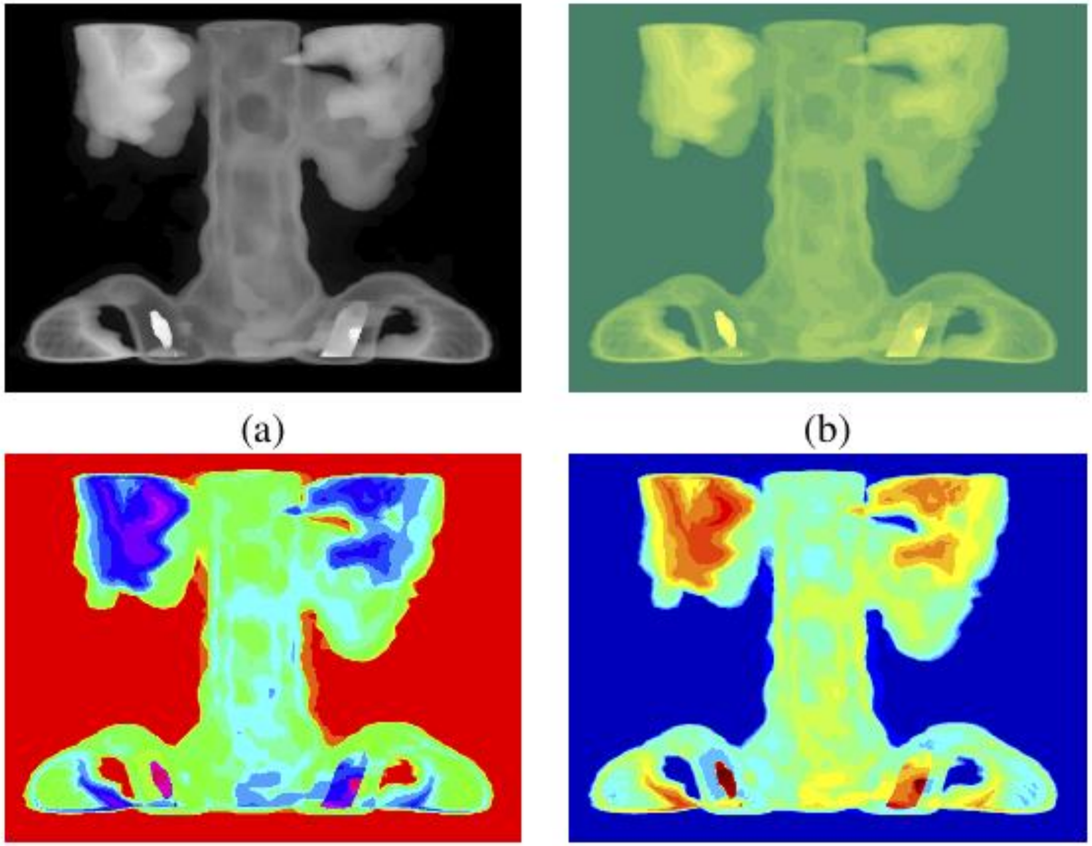
Pseudocolor image processing

- Intensity slicing





Intensity slicing example

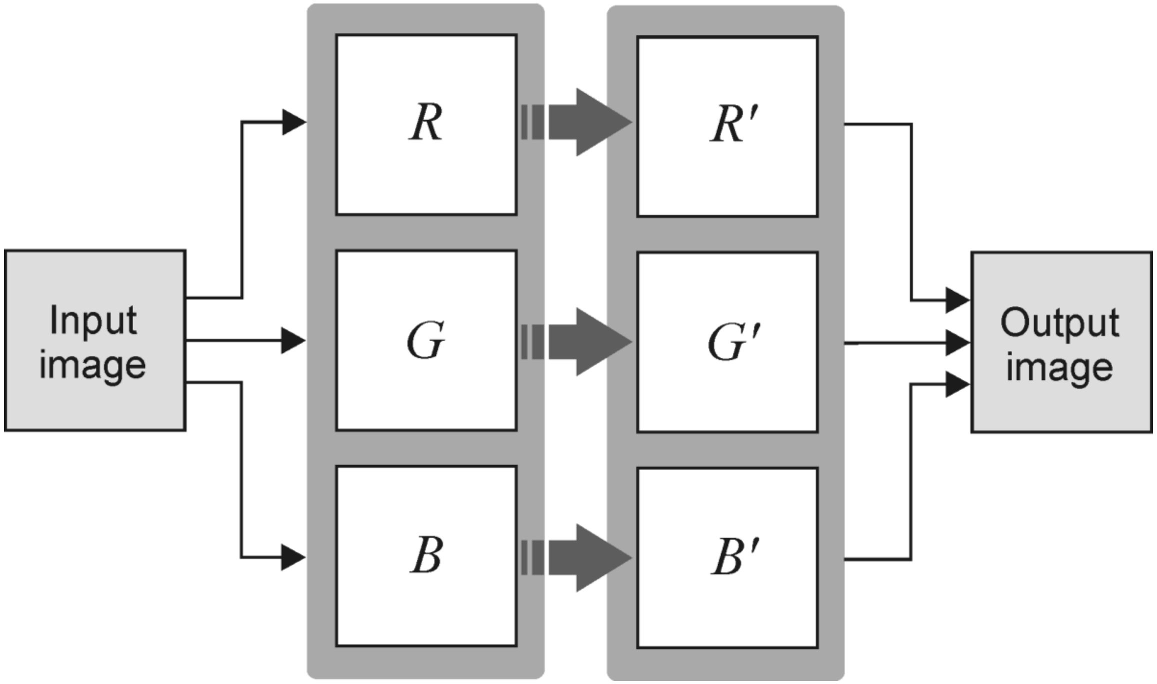


(a) (b)

(c) (d)

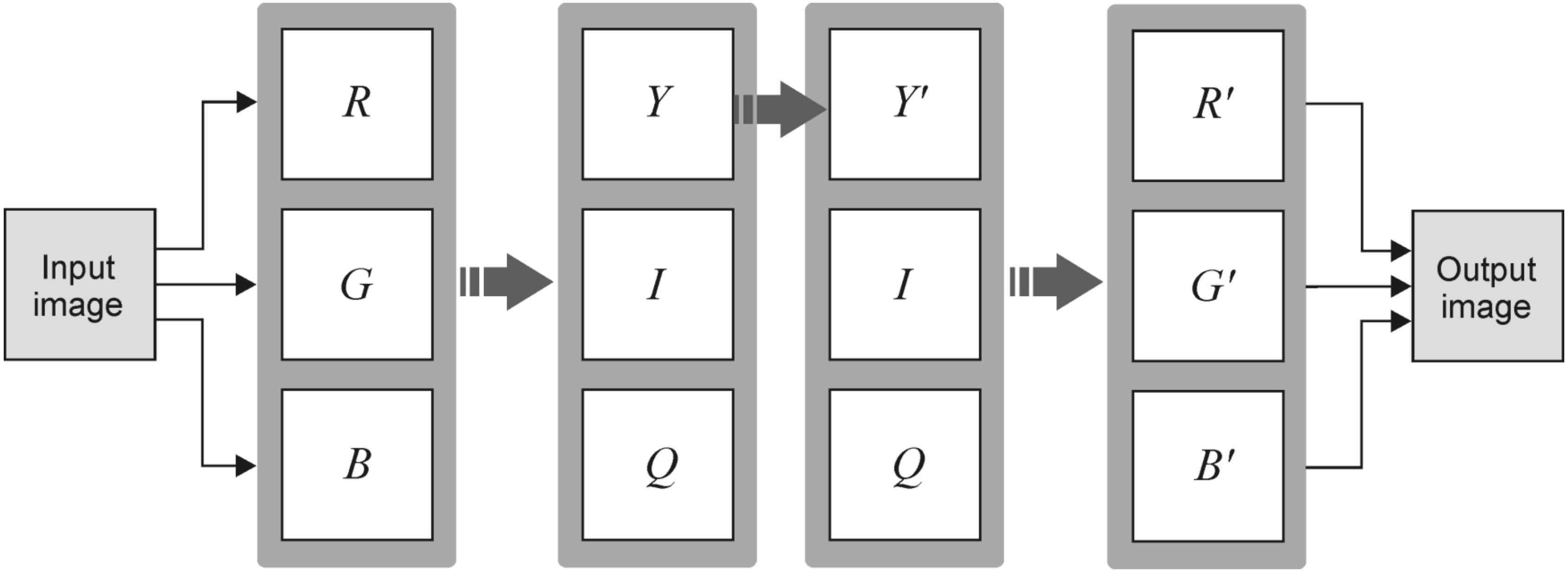
Full-color image processing

- RGB processing



Full-color image processing

- Intensity processing using RGB to YIQ color space conversions



Examples of full-color image processing

- Point transformations
- Histogram processing
- Smoothing and sharpening
- Noise reduction
- Thresholding and segmentation
- Edge detection

Examples of full-color image processing

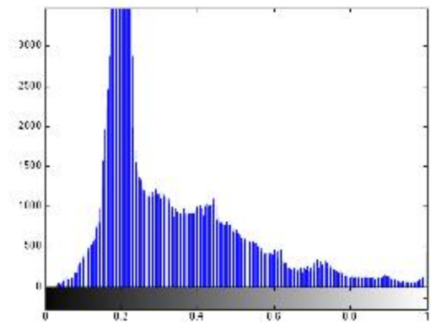
- Point transformations
  - Similar to grayscale images
  - Can be done in RGB color space

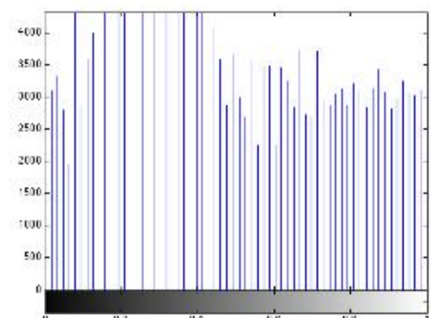
Examples of full-color image processing

- Histogram processing
  - Grayscale equivalent operations require using a color space that allows intensity to be separated from chromaticity:
    - Intensity (e.g., V, Y) is processed
    - Other components (e.g., H and S, I and Q) are left unchanged

Examples of full-color image processing

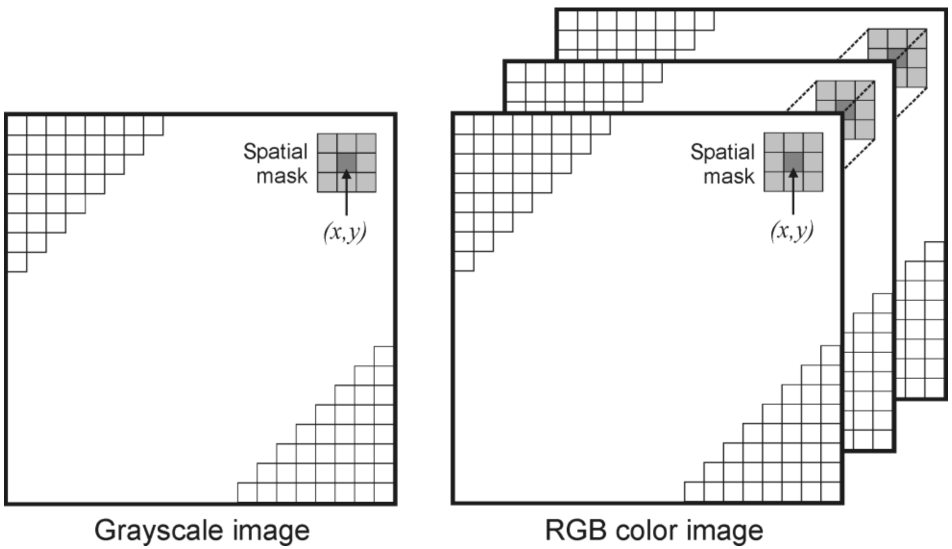
- Histogram processing example (histogram equalization)





Examples of full-color image processing

- Smoothing and sharpening



$$\bar{\mathbf{c}}(x,y) = \begin{bmatrix} \frac{1}{K}\sum_{(s,t)\in \mathbf{S}_{xy}} R(s,t) \\ \frac{1}{K}\sum_{(s,t)\in \mathbf{S}_{xy}} G(s,t) \\ \frac{1}{K}\sum_{(s,t)\in \mathbf{S}_{xy}} B(s,t) \end{bmatrix}$$

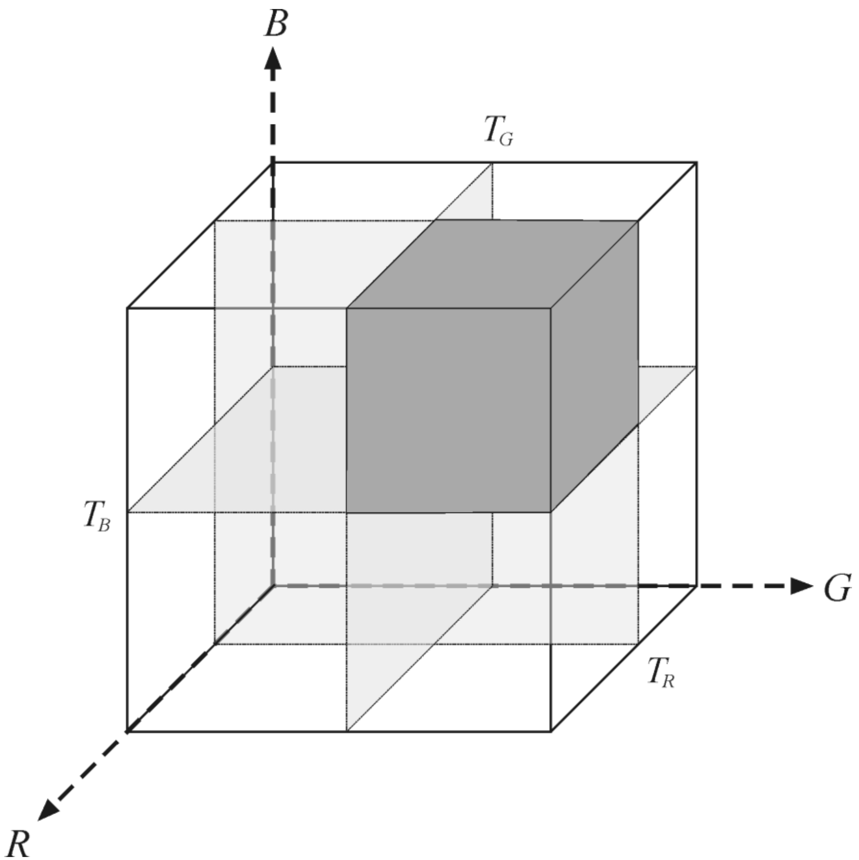
Examples of full-color image processing

- Noise reduction
  - The impact of noise on color images is *strongly dependent on the color model used*.
    - Even when only one of the R, G, or B channels is affected by noise, conversion to another color model such as HSI or YIQ will spread the noise to all components.
  - Linear noise reduction techniques (such as the mean filter) can be applied on each R, G, and B component separately with good results.

Examples of full-color image processing

- Thresholding and segmentation

  - **Basic idea**: to partition the color space into a few regions (which hopefully should correspond to meaningful objects and/or regions in the image) using appropriately chosen thresholds.



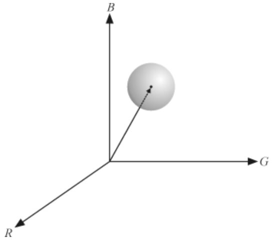
Examples of full-color image processing

- Thresholding and segmentation
  - It is also possible to specify a threshold relative to a distance between any color and a reference color in the RGB space.

$$g(x,y) = \begin{cases} 1 & d(x,y) \le d_{max} \\ 0 & d(x,y) > d_{max} \end{cases} \quad (16.17)$$

where:

$$d(x,y) = \sqrt{[f_R(x,y) - R_0]^2 + [f_G(x,y) - G_0]^2 + [f_B(x,y) - B_0]^2} \quad (16.18)$$

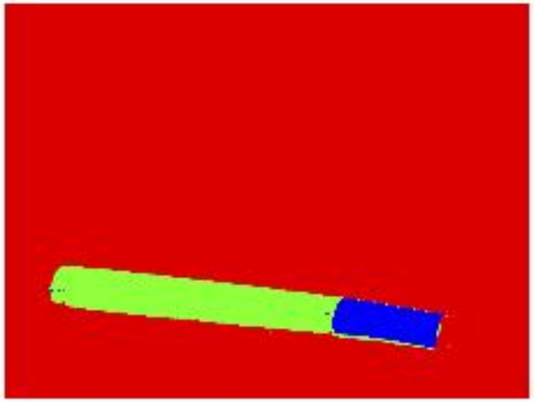


Examples of full-color image processing

- Thresholding and segmentation using **`rgb2ind`**

```
I = imread('marker.png');
n = 3;
[I2,map2] = rgb2ind(I,n,'nodither');
imshow(I)
figure, imshow(I2,map2)
figure, imshow(I2,hsv(3))
```





Examples of full-color image processing

- Thresholding and segmentation using **rgb2ind**

```
I = imread('flower_klu_small.png');
[I2,map2] = rgb2ind(I,2,'nodither');
[I4,map4] = rgb2ind(I,5,'nodither');
figure, imshow(I4,map4)
figure, imshow(I2,map2)
```

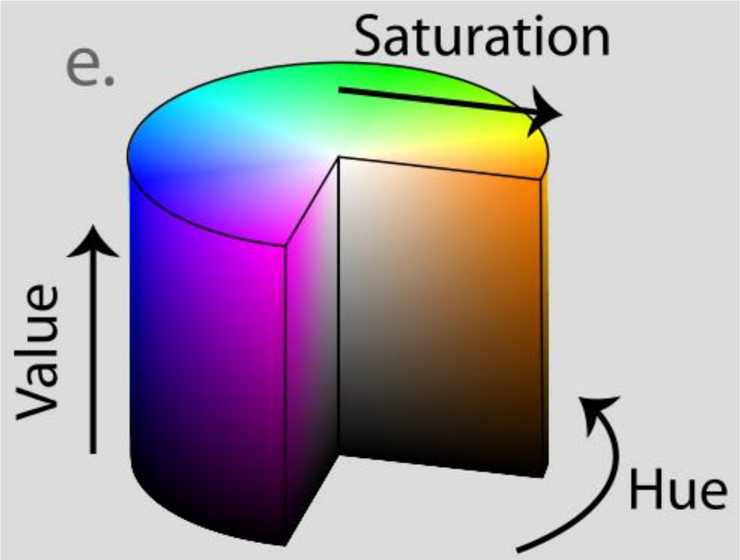
Examples of full-color image processing

- Edge detection
  - No universal definition
  - Usual methods:
    - Detect edges on luminance channel;
    - Detect edges on each channel and OR the results;
    - Detect edges on each channel and add up the results.

RGB para HSV

ISO/IEC 15938-3, 2001 (para RGB 0…1)

```
Max = max(R, G, B); Min = min(R, G, B);
Value = Max;
if(Max  ==  0) then
    Saturation = 0;
else
    Saturation = (Max-Min)/Max;
if( Max == Min ) Hue = 0; /* achromatic */
otherwise:
if( Max == R && G >= B )
    Hue = 60*(G-B)/(Max-Min)
else if( Max == R && G < B )
    Hue = 360 + 60*(G-B)/(Max-Min)
else if( G == Max )
    Hue = 60*(2.0 + (B-R)/(Max-Min))
else
    Hue = 60*(4.0 + (R-G)/(Max-Min))
```



http://en.wikipedia.org/wiki/HSL_and_HSV

Outras versões:

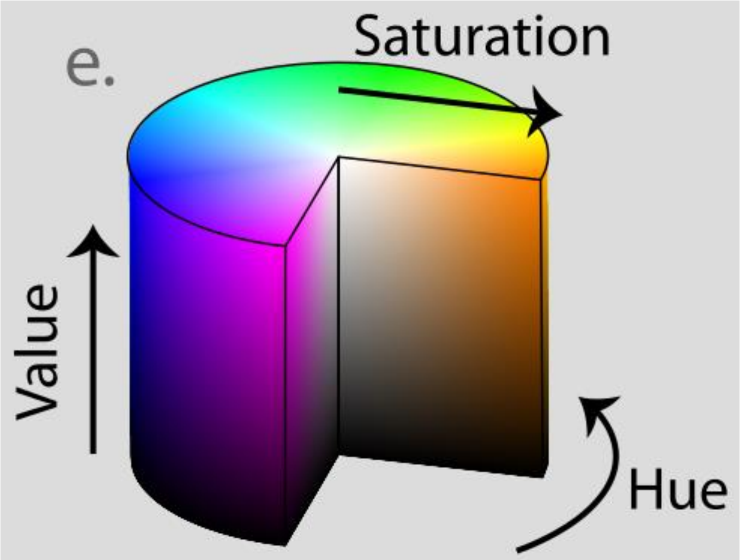
- http://www.easyrgb.com/index.php?X=MATH&H=20#text20
- Gonzalez
- http://en.wikipedia.org/wiki/HSL_and_HSV
- …

(gustavo)

HSV

http://colorizer.org/

- **H**ue : the color type (such as red, blue, or yellow).
  - Ranges from 0 to 360 in most applications (each value corresponds to one color : 0 is red, 45 is a shade of orange and 55 is a shade of yellow).
- **S**aturation :
  - Ranges from 0 to 100% (0 means no color, that is a shade of grey between black and white [pq vai depender do V]; 100 means intense color).
  - Also sometimes called the "purity“ (or how much it has been diluted in white).
- **V**alue: the brightness of the color.
  - Ranges from 0 to 100% (0 is always black; depending on the saturation, 100 may be white or a more or less saturated color).



http://en.wikipedia.org/wiki/HSL_and_HSV

(gustavo)