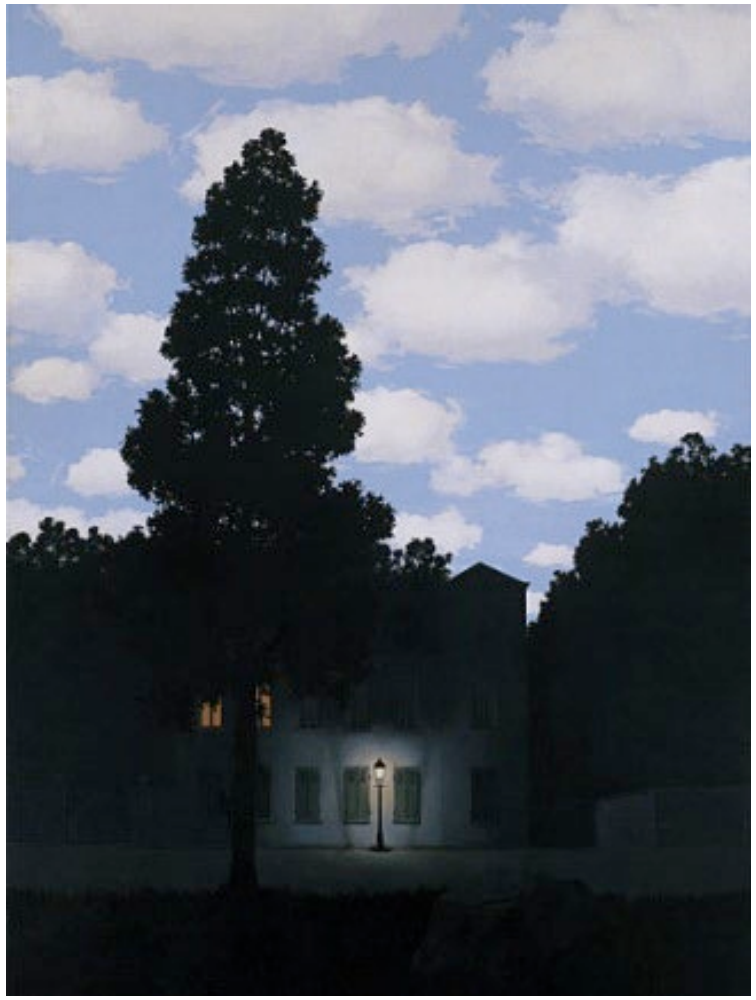
Histograms and Color Balancing

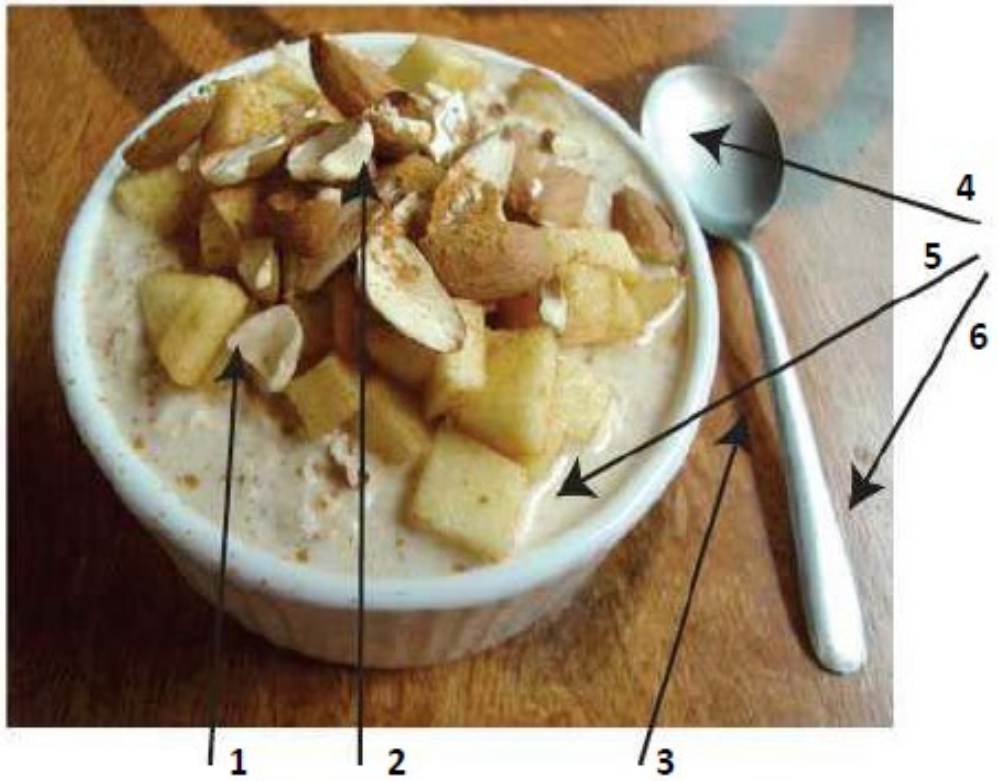


"Empire of Light", Magritte

Computational Photography

Derek Hoiem, University of Illinois

Review of last class

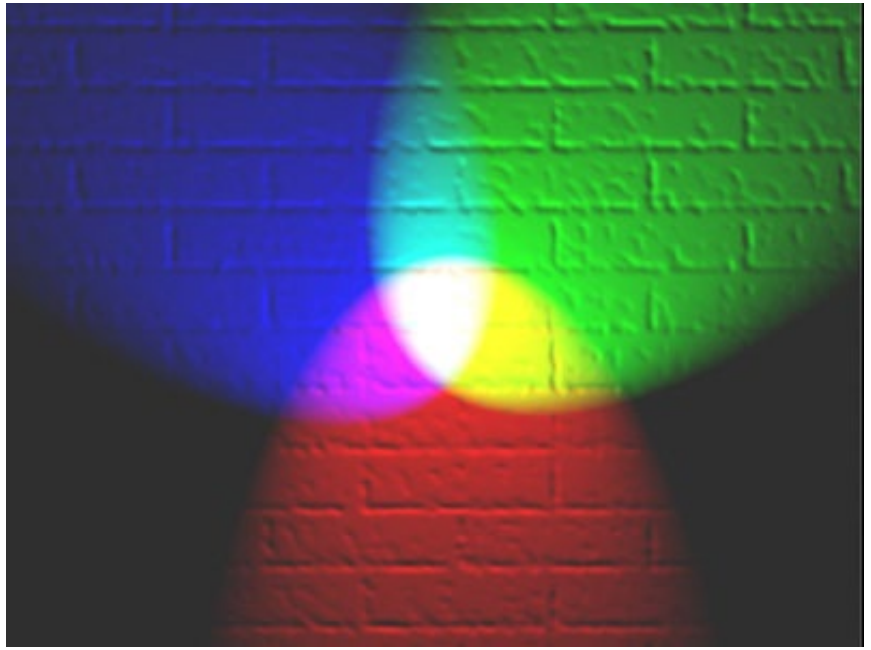


A. For each of the arrows in the above image, name the reasons the pixel near the end of the arrow has its brightness value and explain very briefly. The arrow pointing to milk is pointing to the thin bright line at the edge of the piece of apple; the arrow pointing to the spoon handle is pointing to the bright area on the handle.

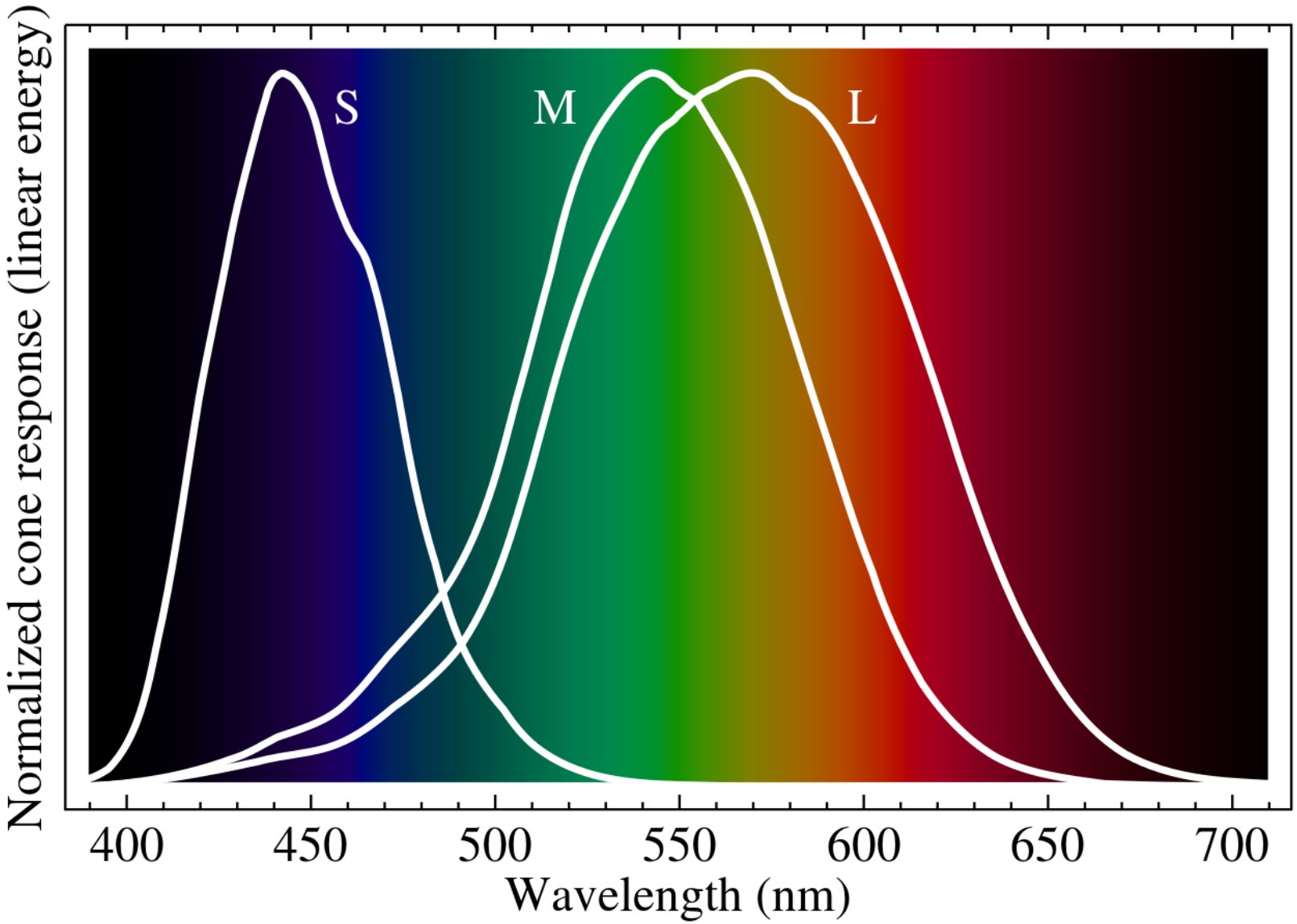
Possible factors: albedo, shadows, texture, specularities, curvature, lighting direction

Today’s class

- How can we represent color?
- How do we adjust the intensity of an image to improve contrast, aesthetics?

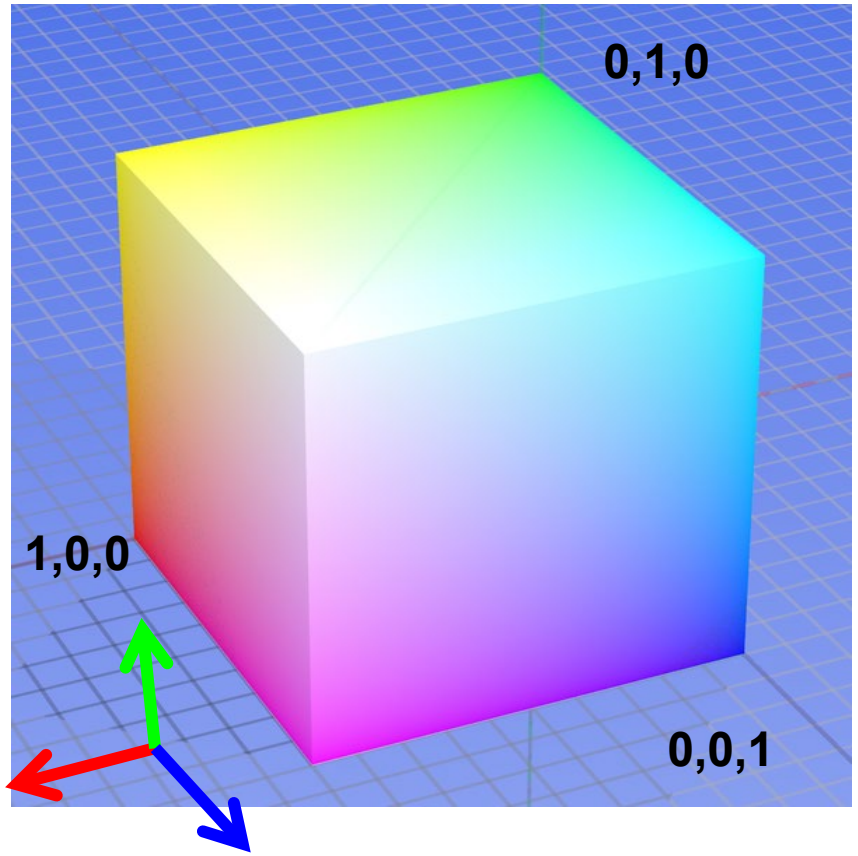


Human eye cone responsivity



Color spaces: RGB

Default color space



R
(G=0,B=0)

G
(R=0,B=0)

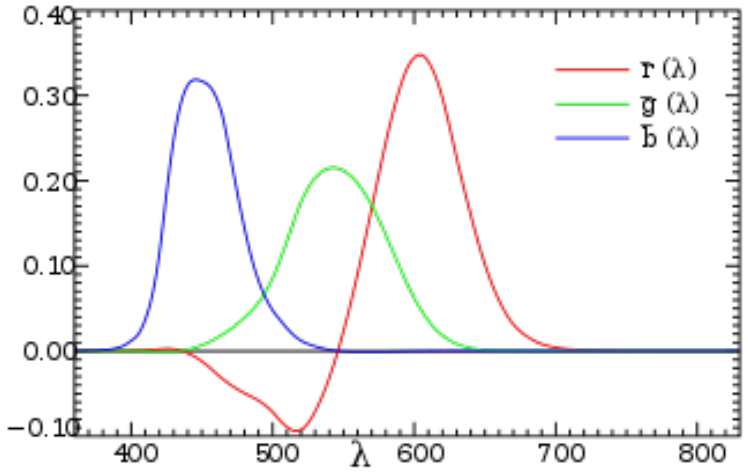
B
(R=0,G=0)

Some drawbacks

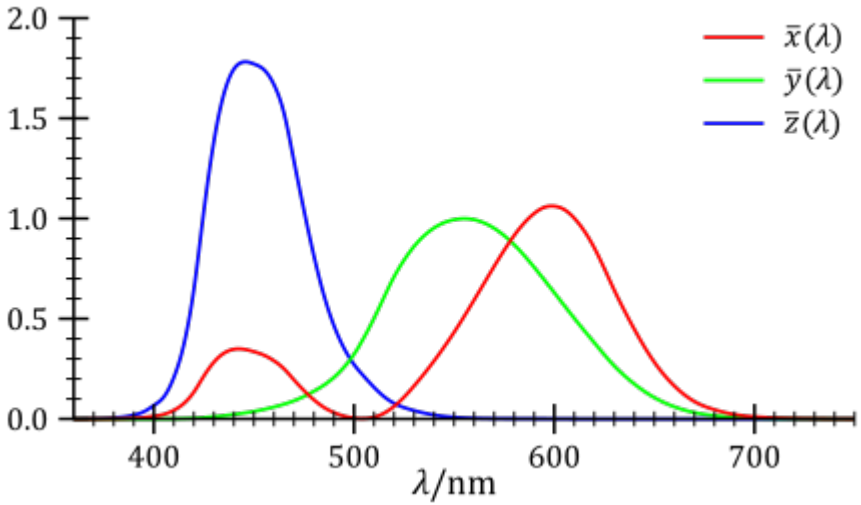
- Strongly correlated channels
- Non-perceptual

Image from: http://en.wikipedia.org/wiki/File:RGB_color_solid_cube.png

Trichromacy and CIE-XYZ



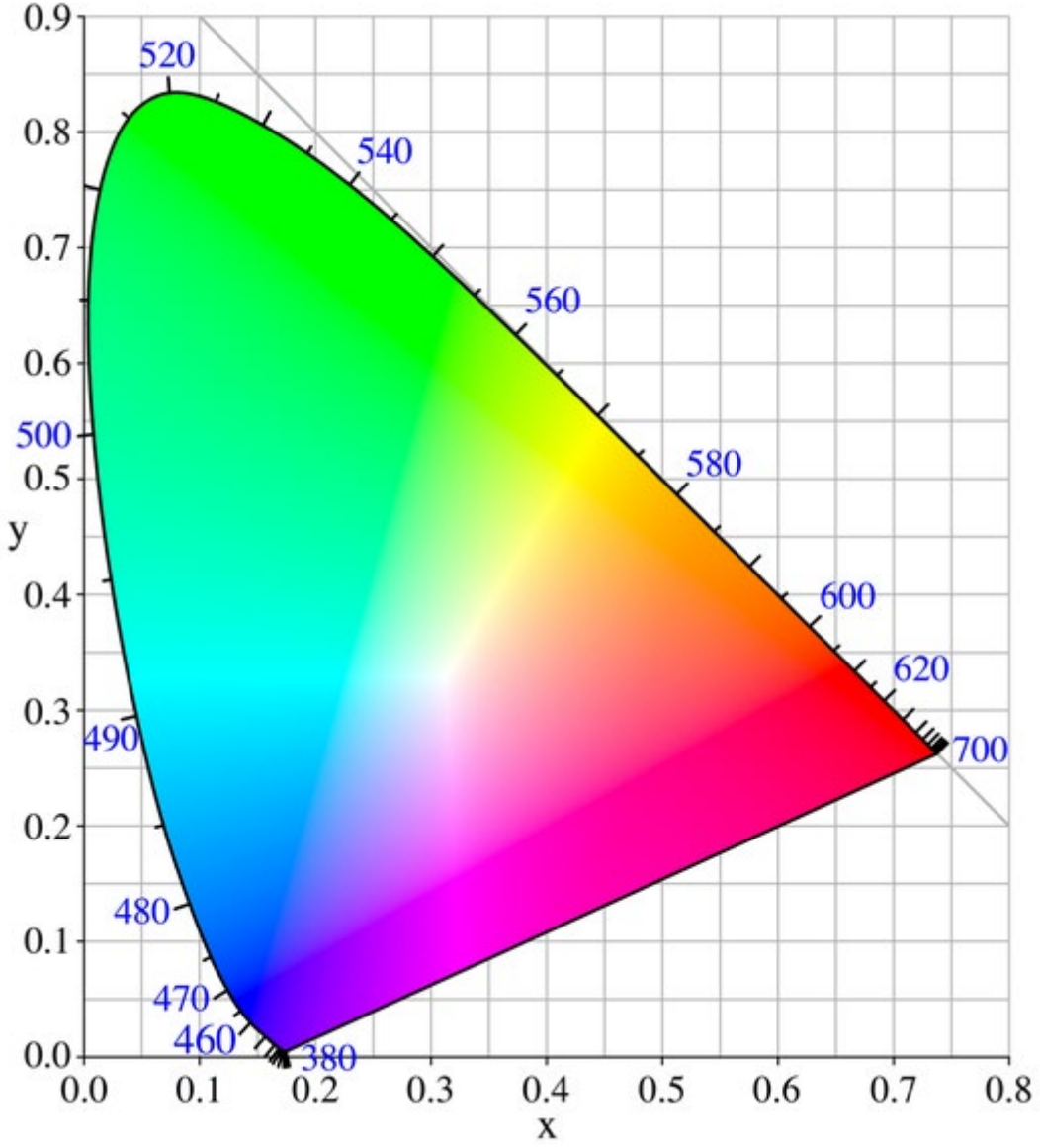
Perceptual equivalents with RGB



Perceptual equivalents with CIE-XYZ

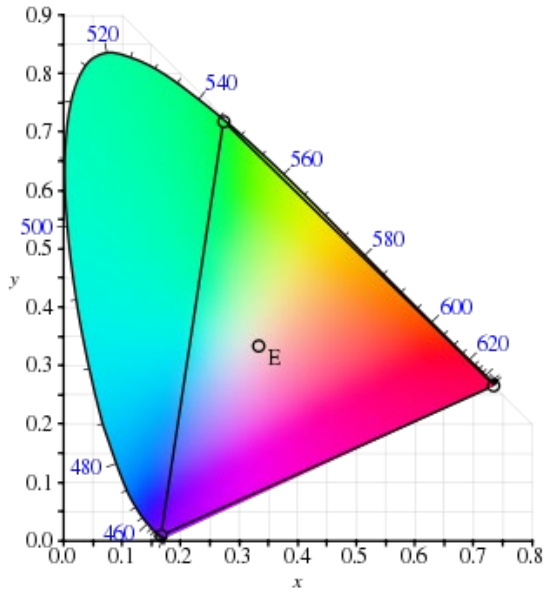
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{0.17697} \begin{bmatrix} 0.49 & 0.31 & 0.20 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.00 & 0.01 & 0.99 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Color Space: CIE-XYZ



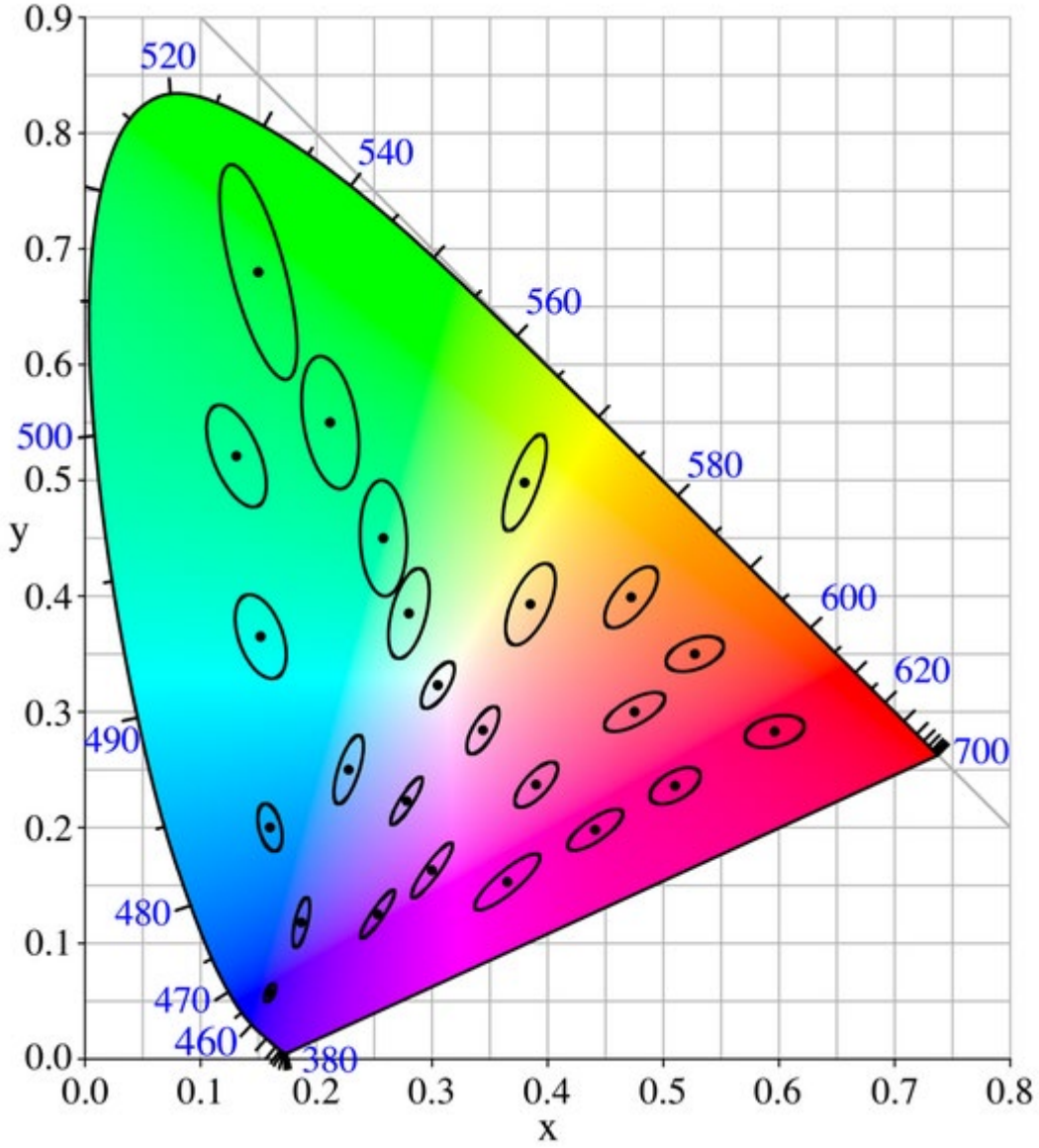
$$x = \frac{X}{X + Y + Z}$$

$$y = \frac{Y}{X + Y + Z}$$



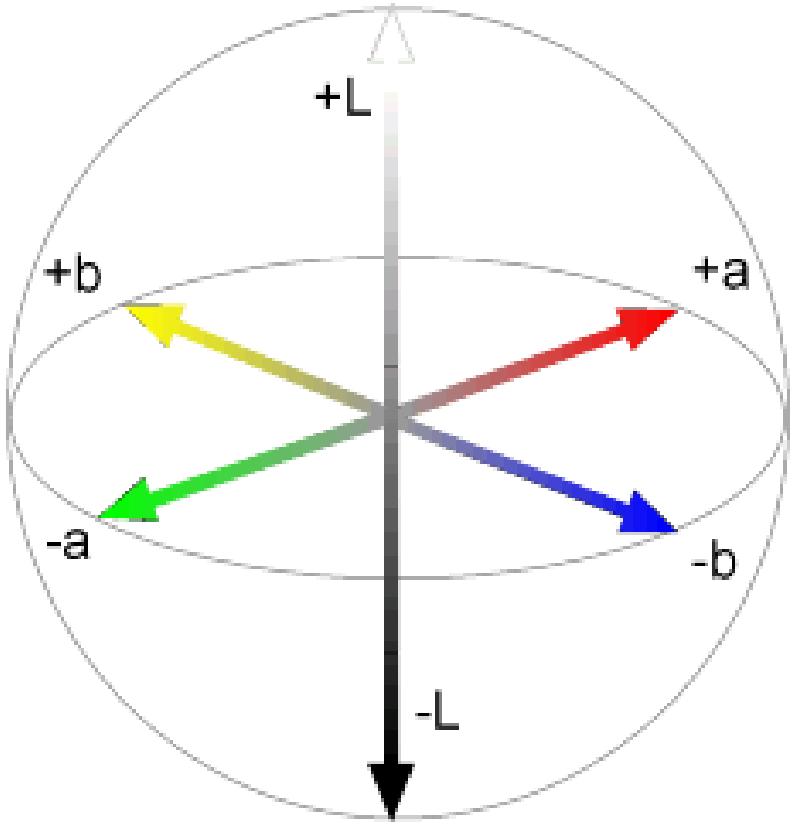
RGB portion is in triangle

Perceptual uniformity



Color spaces: CIE L*a*b*

"Perceptually uniform" color space



Luminance = brightness
Chrominance = color

L
(a=0,b=0)

a
(L=65,b=0)

b
(L=65,a=0)

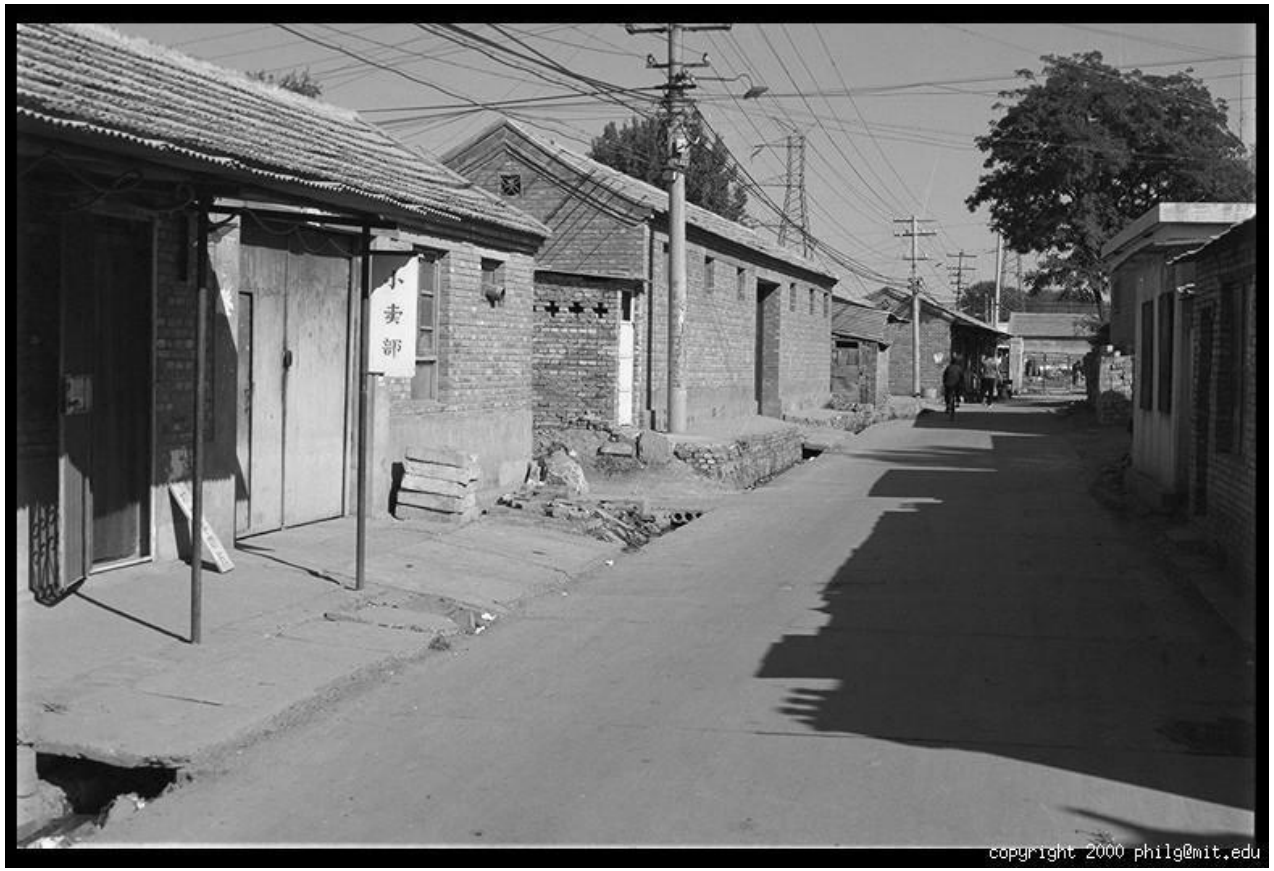
If you had to choose, would you rather go without luminance or chrominance?

If you had to choose, would you rather go without luminance or chrominance?

Most information in intensity

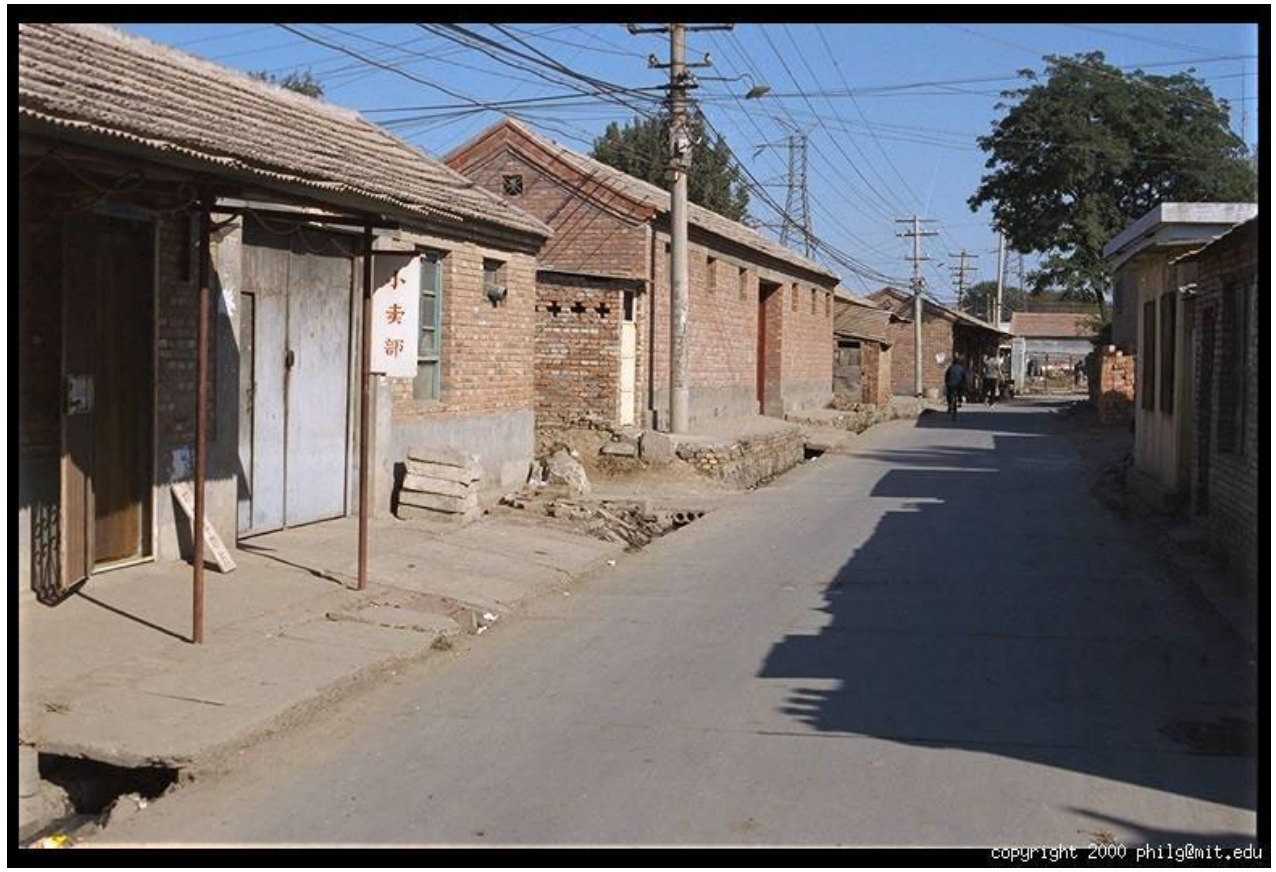
Only color shown – constant intensity

Most information in intensity



Only intensity shown – constant color

Most information in intensity

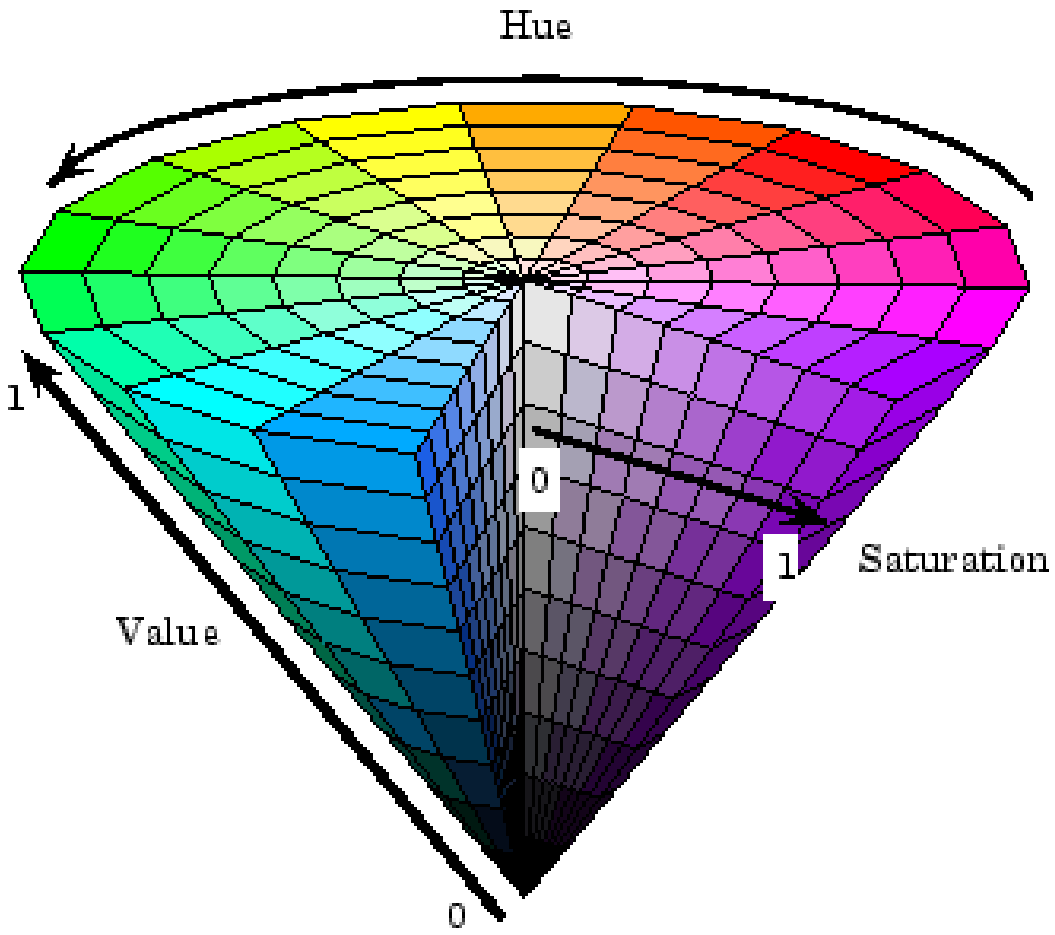


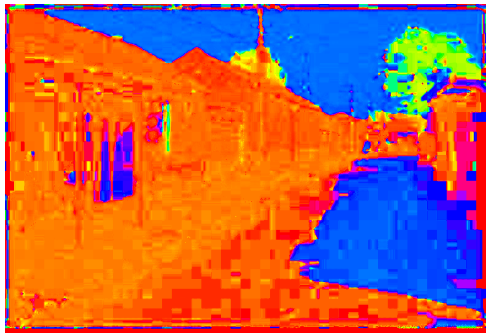
Original image

Image from Phillip Greenspun, used with permission

Color spaces: HSV

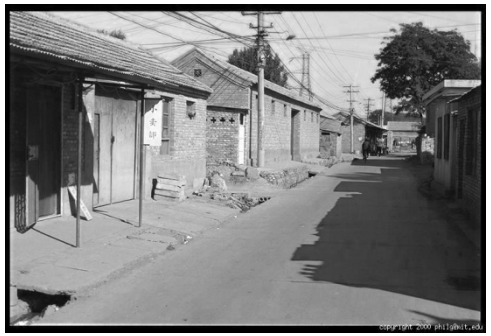
Intuitive color space





H
(S=1,V=1)

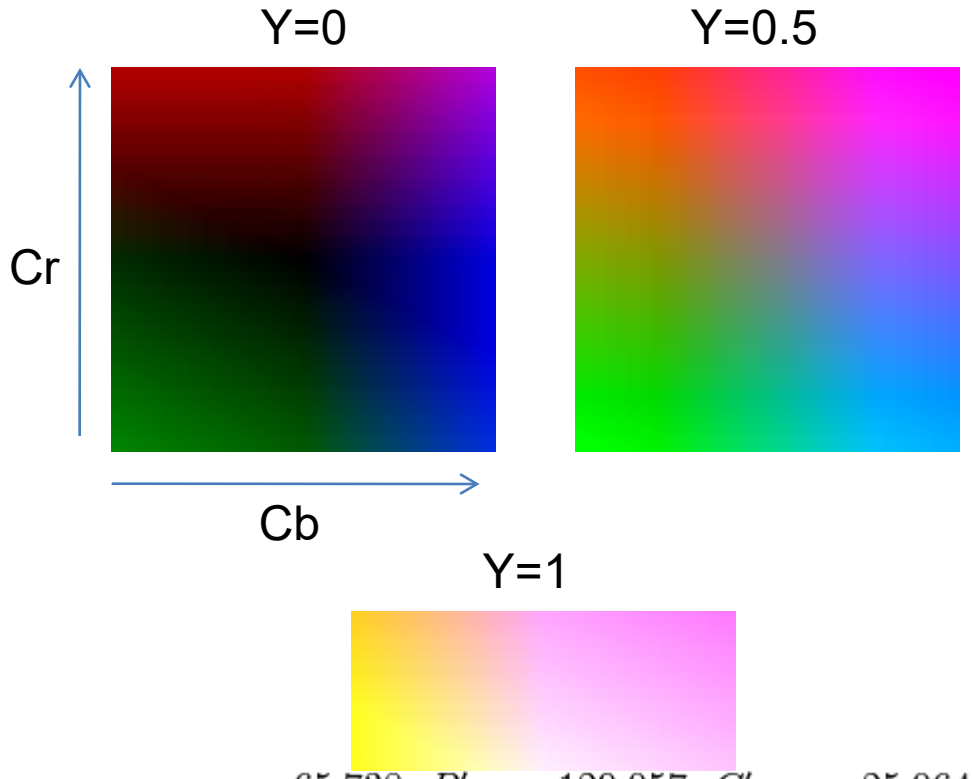
S
(H=1,V=1)



V
(H=1,S=0)

Color spaces: YCbCr

Fast to compute, good for compression, used by TV



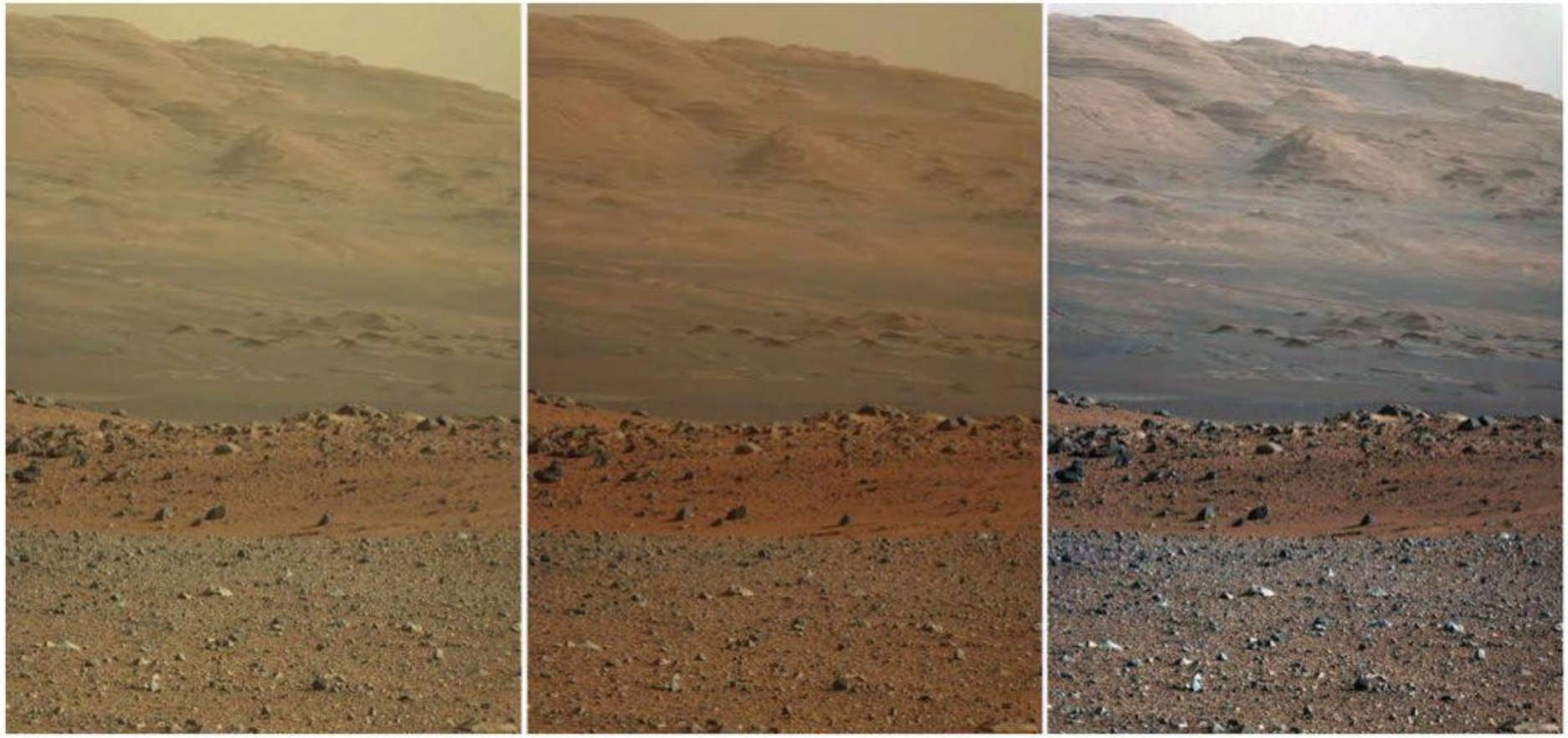
$$
\begin{aligned}
Y' &= 16 + \frac{65.738 \cdot R'_D}{256} + \frac{129.057 \cdot G'_D}{256} + \frac{25.064 \cdot B'_D}{256} \\
C_B &= 128 + \frac{-37.945 \cdot R'_D}{256} - \frac{74.494 \cdot G'_D}{256} + \frac{112.439 \cdot B'_D}{256} \\
C_R &= 128 + \frac{112.439 \cdot R'_D}{256} - \frac{94.154 \cdot G'_D}{256} - \frac{18.285 \cdot B'_D}{256}
\end{aligned}
$$

Y
(Cb=0.5,Cr=0.5)

Cb
(Y=0.5,Cr=0.5)

Cr
(Y=0.5,Cb=05)

Color balancing



Unprocessed Color (JPL Web site)
(raw data from Mars, uncalibrated)

“Natural” Color
(uses calibrated data)

“White Balanced” Color
(Assumes something in the scene is white)

Contrast enhancement

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

Important ideas

- Typical images are gray on average; this can be used to detect distortions
- Larger differences are more visible, so using the full intensity range improves visibility
- It’s often easier to work in a non-RGB color space

Color balancing via linear adjustment

- Simple idea: multiply R, G, and B values by separate constants

$$\begin{bmatrix} \tilde{r} \\ \tilde{g} \\ \tilde{b} \end{bmatrix} = \begin{bmatrix} \alpha_r & 0 & 0 \\ 0 & \alpha_g & 0 \\ 0 & 0 & \alpha_b \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix}$$

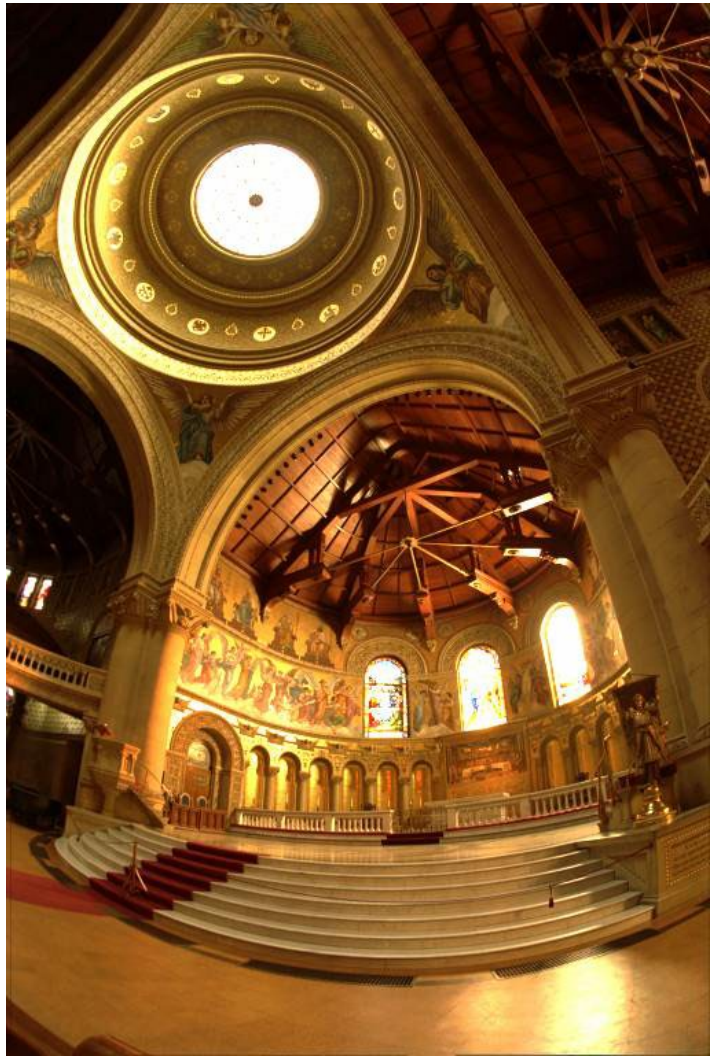
- How to choose the constants?
  - "Gray world" assumption: average value should be gray
  - White balancing: choose a reference as the white or gray color
  - Better to balance in camera's RGB (linear) than display RGB (non-linear)

Tone Mapping

- Typical problem: compress values from a high range to a smaller range
  - E.g., camera captures 12-bit linear intensity and needs to compress to 8 bits

Example: Linear display of HDR

Scaled for brightest pixels

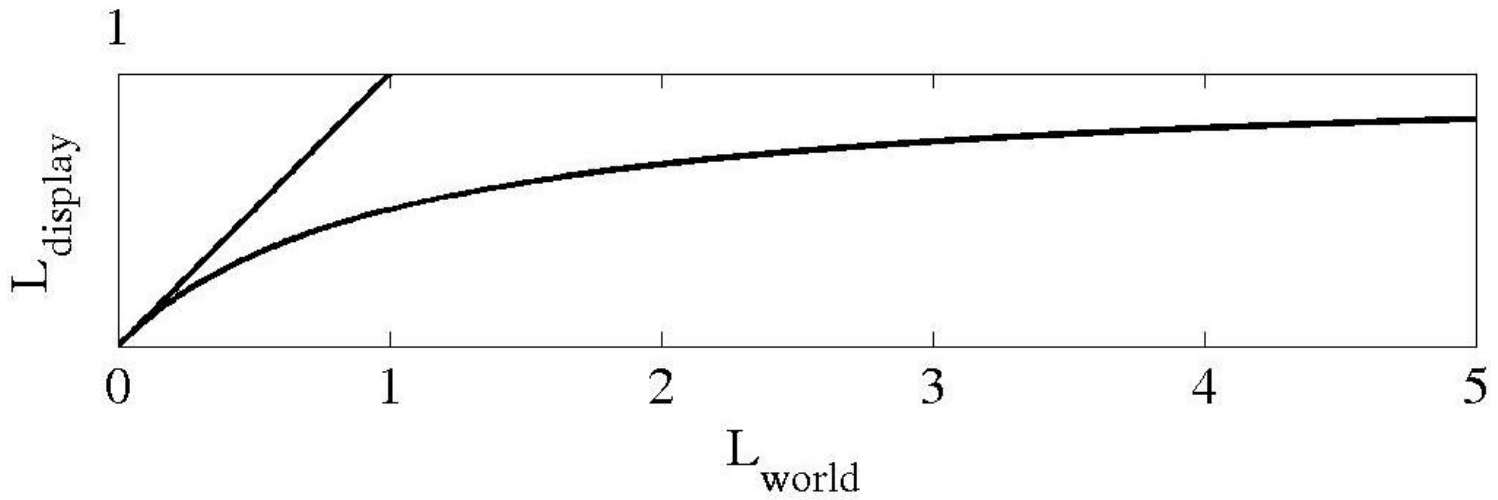


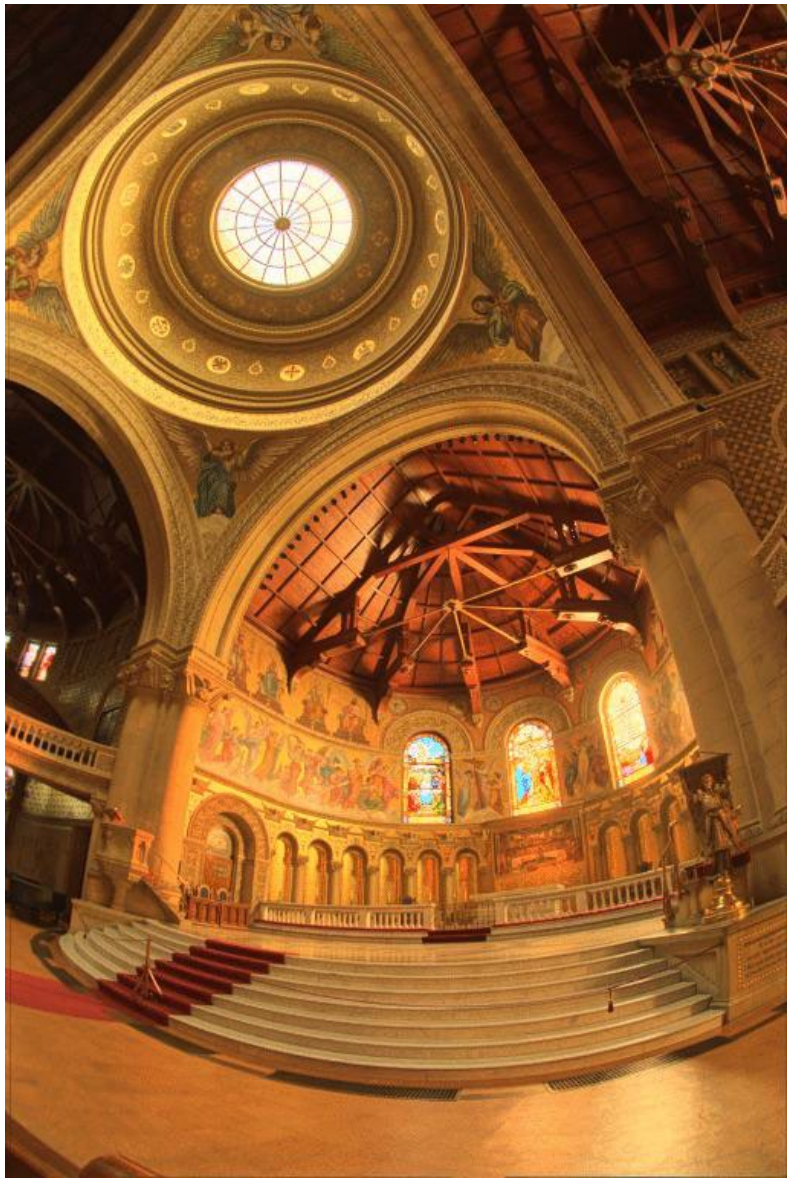
Scaled for darkest pixels

Global operator (Reinhart et al.)

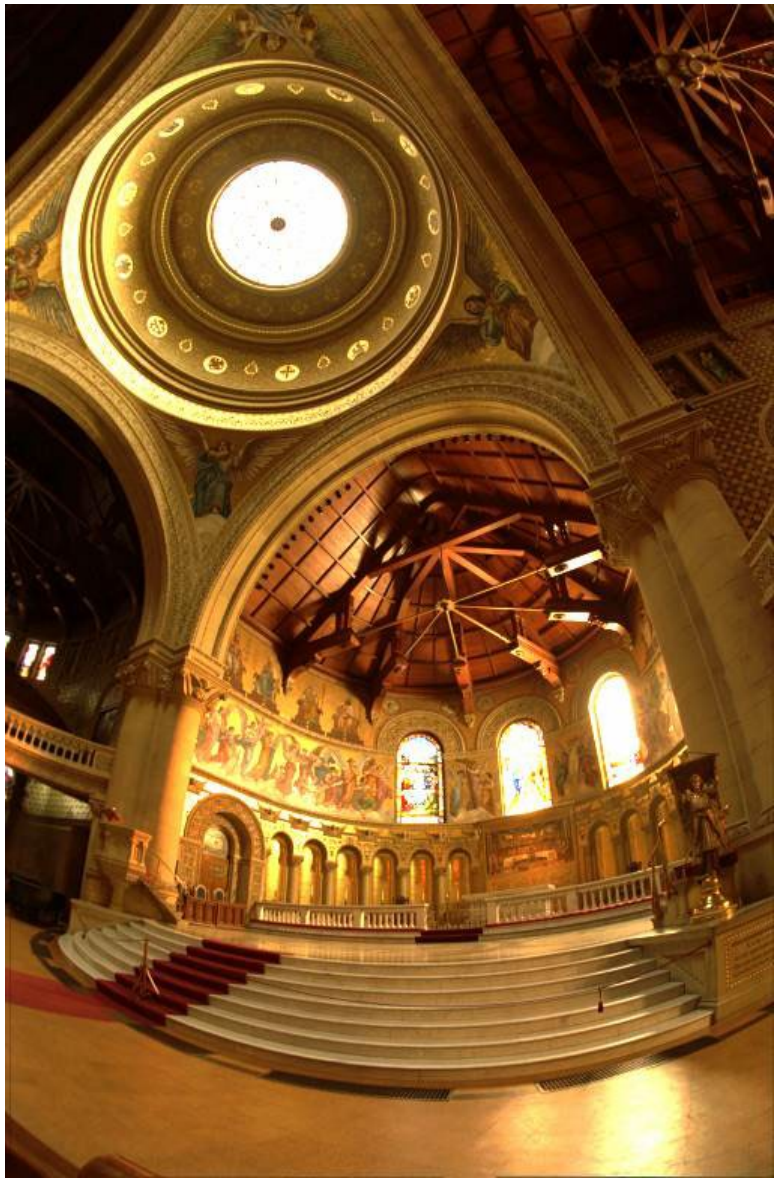
- Simple solution: map to a non-linear range of values

$$L_{display} = \frac{L_{world}}{1 + L_{world}}$$



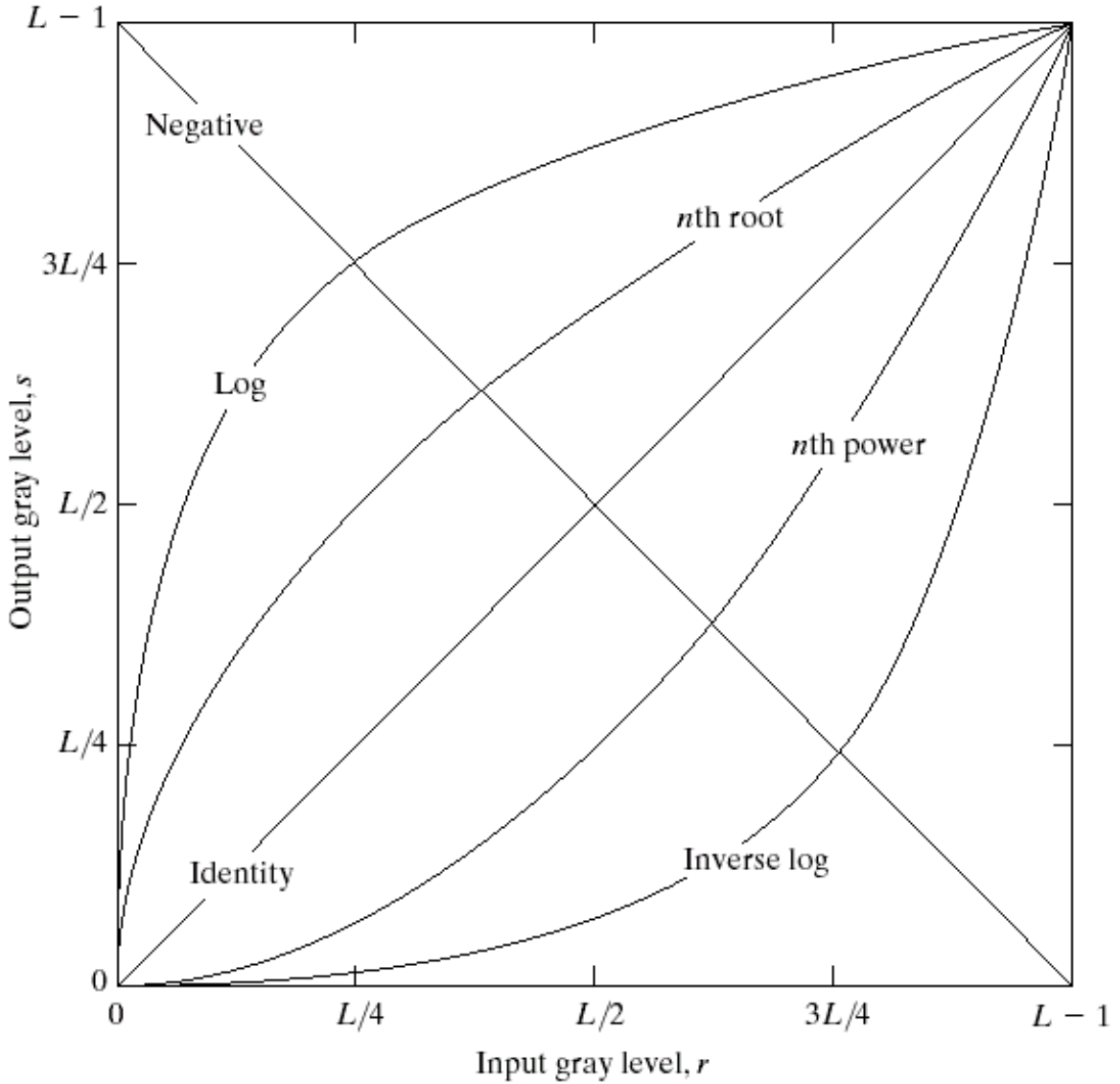


Reinhart Operator



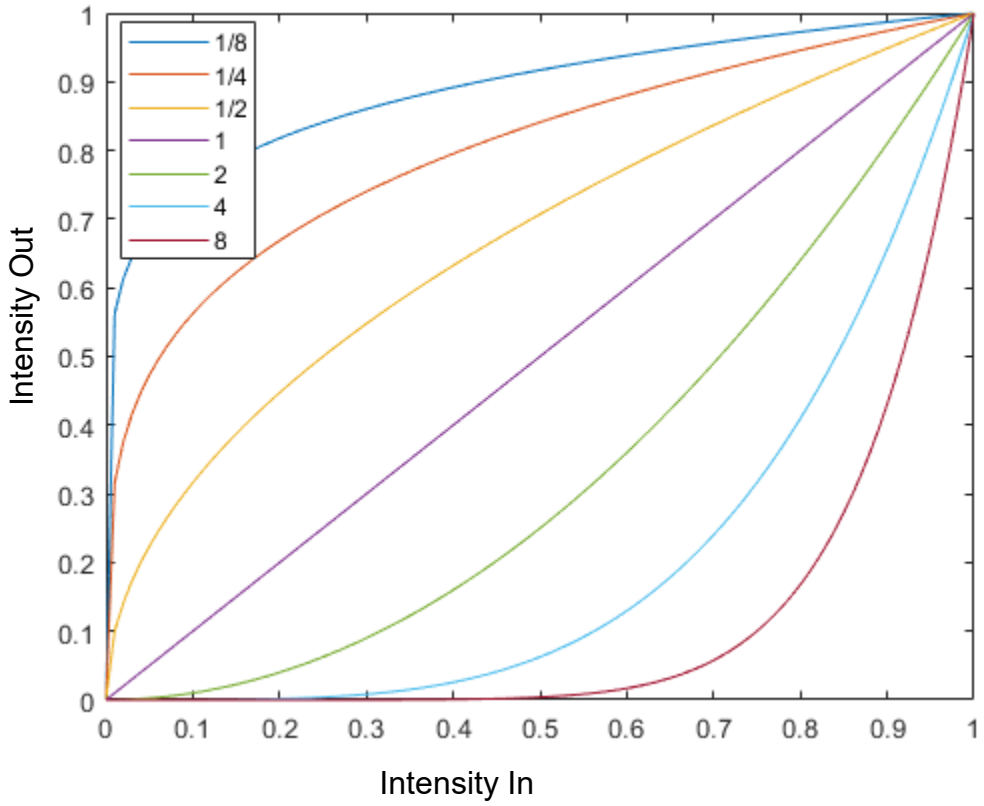
Darkest **0.1%** scaled to display

Point Processing: apply a function to each pixel intensity to map it to a new value



Gamma adjustment

$$i_{out} = i_{in}^{\gamma}$$



γ = 0.5

γ = 1

γ = 2

Matlab example

Histogram equalization

- Basic idea: reassign values so that the number of pixels with each value is more evenly distributed
- Histogram: a count of how many pixels have each value

$$h_i = \sum_{j \in pixels} \mathbf{1}(p_j == i)$$

- Cumulative histogram: count of number of pixels less than or equal to each value

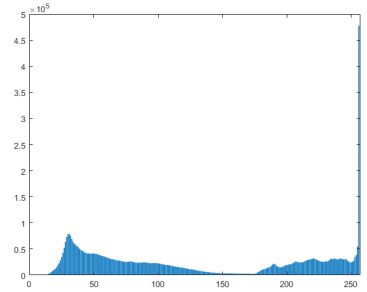
$$c_i = c_{i-1} + h_i$$

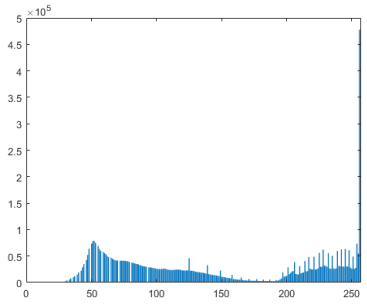
Histogram is count of elements that have a particular value or range of values

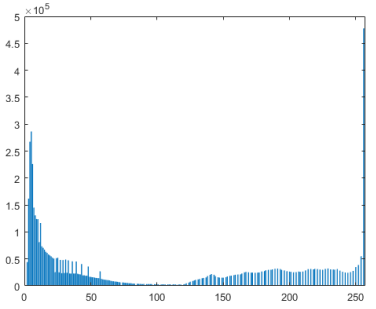
A = [1 1 2 3 3 3 5 6]
H = hist(A, 1:6)
H = [2 1 3 0 1 1]
C = cumsum(H)
C = [2 3 6 6 7 8]

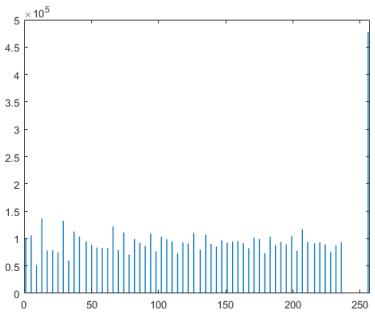
B = [5 6 6 6 8 8 9]
H = hist(B, 5:9)
H = ?
C = ?

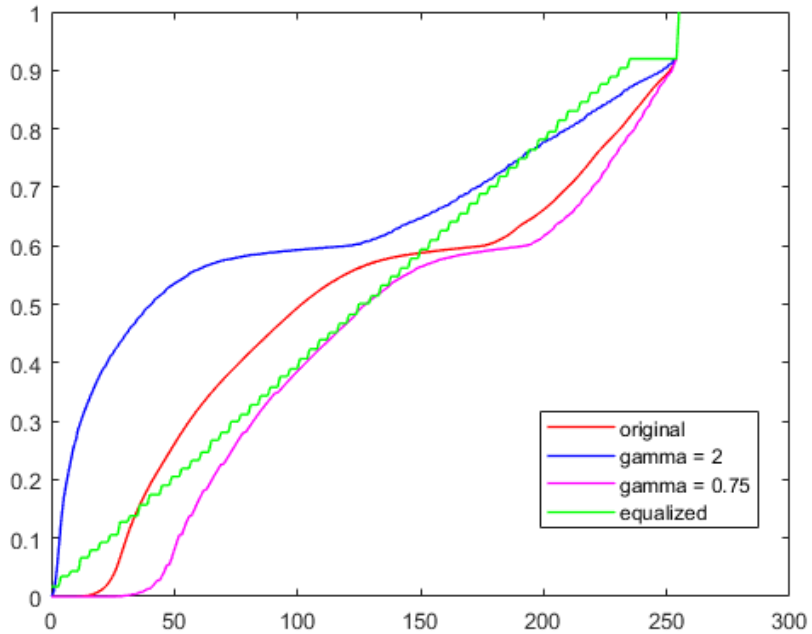
Image Histograms



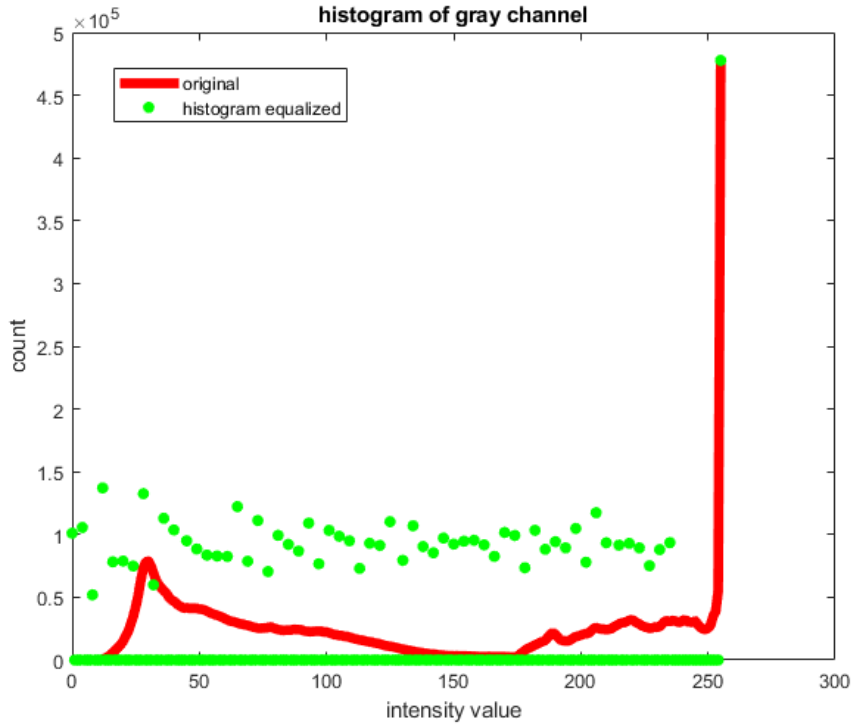






Cumulative Histograms

Histogram Equalization



Algorithm for global histogram equalization

Goal: Given image with pixel values $0 \leq pj \leq 255,\ j = 0..N$

specify function $\mathrm{f}(i)$ that remaps pixel values, so that the new values are more broadly distributed

1. Compute cumulative histogram: $c(i), i = 0..255$

$$h(i) = \sum_{j \in pixels} \mathbf{1}(p_j == i),\ c(i) = c(i-1) + h(i)$$

2. $\mathrm{f}(i) = \alpha \cdot \frac{c(i)}{N} \cdot 255 + (1-\alpha) \cdot i$

   – Blends between original image and image with equalized histogram

Locally weighted histograms

- Compute cumulative histograms in non-overlapping MxM grid
- For each pixel, interpolate between the histograms from the four nearest grid cells

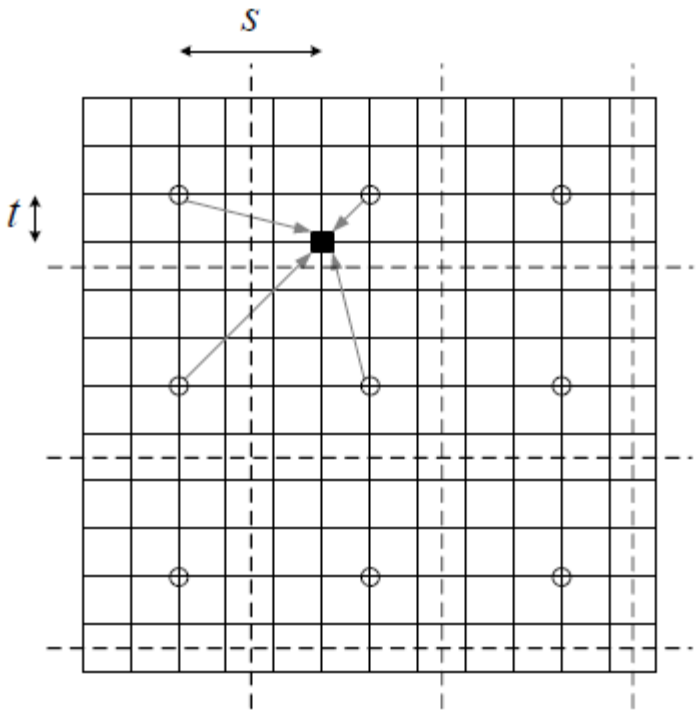
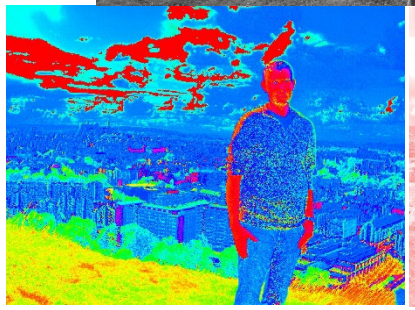
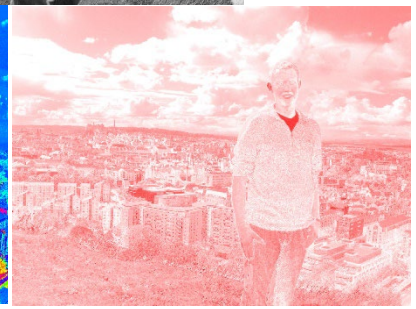


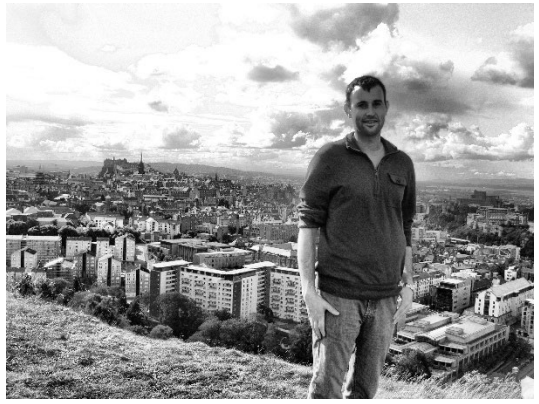
Figure from Szeliski book (Fig. 3.9)
Pixel (black) is mapped based on interpolated value from its cell and nearest horizontal, vertical, diagonal neighbors

Application of adaptive histogram equalization to color image

rgb2hsv







Locally Adaptive Histogram Equalization of “v” channel

hsv2rgb

Before

After

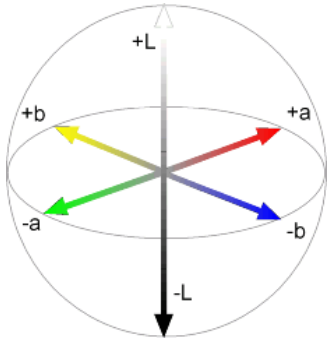
Other issues

- Dealing with color images
  - Often better to split into luminance and chrominance to avoid unwanted color shift
- Manipulating particular regions
  - Can use mask to select particular areas for manipulation
- Useful Python functions/modules
  - `skimage.color: color conversion, e.g. rgb2hsv`
  - `numpy: histogram, cumsum`

Matlab Example 2

Things to remember

- Familiarize yourself with the basic color spaces: RGB, HSV, Lab



- Simple auto contrast/color adjustments: gray world assumption, histogram equalization

- When improving contrast in a color image, often best to operate on luminance channel

Next class: texture synthesis and transfer