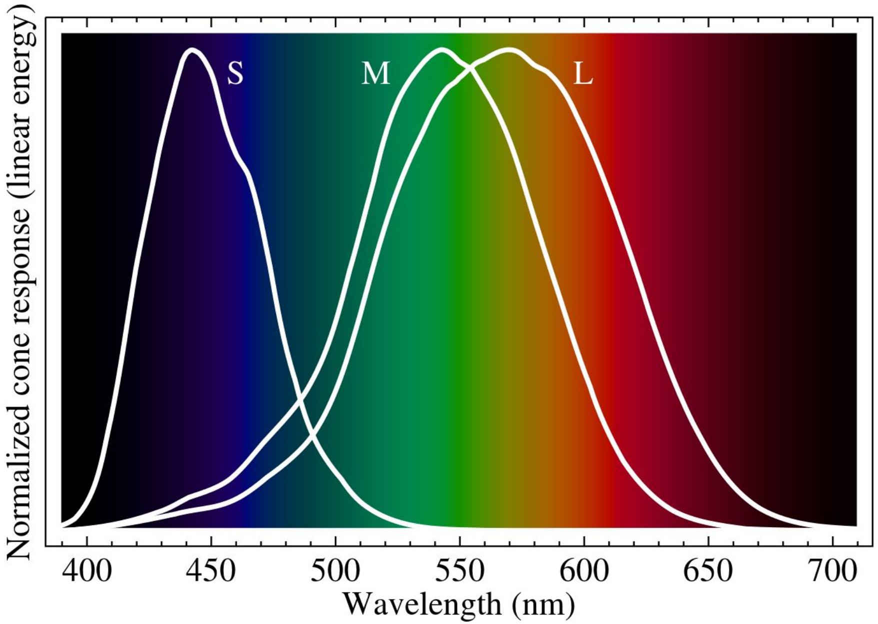
Color Theory, Part 1: Color Difference Metrics

Stephen Huan
TJ Vision & Graphics Club, January 27, 2021

Introduction

Basics: Physics & Biology

- Light electromagnetic radiation
- Wavelength “color”
- Human eye contains cone cells
- (L)ong, (M)edium, (S)hort type
    - Roughly (**R**)ed, (**G**)reen, **(B)**lue
- Hence, “RGB” color space

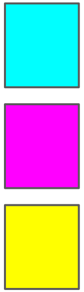


But What About RYB?

- Red, Yellow, Blue taught in elementary school
- If human eye roughly RGB, why?
- Think of colors as vectors, “color system” = basis
  - Effectiveness of system the size of span, “color gamut”
- Most effective therefore RGB!
  - To form color, linear combination of $R\,(1, 0, 0) + G\,(0, 1, 0) + B\,(0, 0, 1) = (R, G, B)$
  - Rough correspondence with cones
- But this is if we are emitting the light…
  - e.g from a computer monitor
  - “Additive” color system
- What about printing?

Subtractive Color System

- Paper has color, doesn't generate its own light
- Instead, *reflects* white light containing all colors
- Remove colors by absorbing
- Hence, use *opposite* color for basis
    - Opposite in a discrete space: 0 -> 1, 1 -> 0. $\sim x = 1 - x$
- ***R*** = (255, 0, 0) -> -***R*** = (0, 255, 255) = "cyan"
- -***G*** = (255, 0, 255) = "magenta"
- -***B*** = (255, 255, 0) = "yellow"
- Thus, CMY system for "subtractive" color system



Basis of the Subtractive Color System

$\boldsymbol{W} = 2*255*(1, 1, 1)$

$\boldsymbol{W} - [c_1\ (0, 1, 1) + c_2\ (1, 0, 1) + c_3\ (1, 1, 0)] = (R, G, B)$

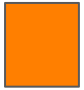
$\boldsymbol{W} - [c_1\ \boldsymbol{v}_1 + c_2\ \boldsymbol{v}_2 + c_3\ \boldsymbol{v}_3] = \boldsymbol{x}$

$c_1\ \boldsymbol{v}_1 + c_2\ \boldsymbol{v}_2 + c_3\ \boldsymbol{v}_3 = \boldsymbol{W} - \boldsymbol{x}$

$[\boldsymbol{v}_1\ \ \boldsymbol{v}_2\ \ \boldsymbol{v}_3]\ [c_1\ \ c_2\ \ c_3]^T = \boldsymbol{W} - \boldsymbol{x}$

$\boldsymbol{c} = M^{-1}(\boldsymbol{W} - \boldsymbol{x})$

- e.g. $\boldsymbol{W} - (319\boldsymbol{C} + 191\boldsymbol{M} + 64\boldsymbol{Y}) = (255, 127, 0)$



Implications

- Most color spaces 3D, R^3
- Hence, kd-tree viable in low dimensionality
- kd-tree naturally implies metric of “closeness”
- Can use *k*-means on color space
  - Assign point to “closest” center
  - Assign centers to “centroid”
- Are these well defined?
- How to measure “distance?”

Color Difference

Two Basic Approaches

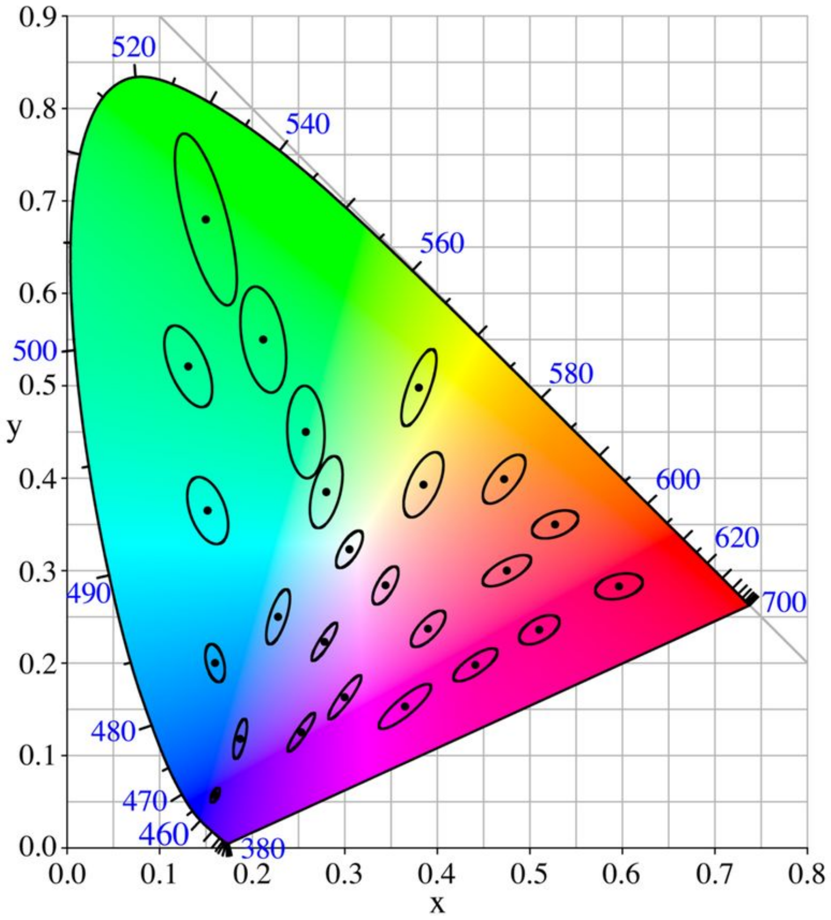
1. Create a *Uniform Color Space* (UCS)
    - Projection where Euclidean distance works well
    - Distance, centroid, etc. naturally well-defined
    - If Euclidean doesn’t work, centroid doesn’t work!
        - More on uniform color spaces in the future....
2. Change distance metrics

1. Uniform Color Spaces

- International Commission on Illumination, (CIE) - Commission internationale de l'éclairage
- Transform RGB (more commonly CIE’s XYZ, space based off wavelengths) to some other space, use Euclidean in that space
- How to measure effectiveness?
- Need to ask humans to evaluate!

MacAdam ellipses

- If uniform, points (colors) equally far away from center color define circle
- Ask people to match color to target color until equal
- Found matches fall into ellipses
- Thus, XYZ not perceptually uniform!



CIELAB Color Space

- “Uniform” color space designed to fix XYZ
- L^* = lightness, 0 = black, 100 = white
- a^* = green-red, negative = green, positive = red
- b^* = blue-yellow, negative = blue, positive = yellow

- RGB -> LAB?

1. RGB -> XYZ
2. XYZ -> LAB

RGB -> XYZ

Standard-RGB → XYZ

```
//sR, sG and sB (Standard RGB) input range = 0 ÷ 255
//X, Y and Z output refer to a D65/2° standard illuminant.

var_R = ( sR / 255 )
var_G = ( sG / 255 )
var_B = ( sB / 255 )

if ( var_R > 0.04045 ) var_R = ( ( var_R + 0.055 ) / 1.055 ) ^ 2.4
else                   var_R = var_R / 12.92
if ( var_G > 0.04045 ) var_G = ( ( var_G + 0.055 ) / 1.055 ) ^ 2.4
else                   var_G = var_G / 12.92
if ( var_B > 0.04045 ) var_B = ( ( var_B + 0.055 ) / 1.055 ) ^ 2.4
else                   var_B = var_B / 12.92

var_R = var_R * 100
var_G = var_G * 100
var_B = var_B * 100

X = var_R * 0.4124 + var_G * 0.3576 + var_B * 0.1805
Y = var_R * 0.2126 + var_G * 0.7152 + var_B * 0.0722
Z = var_R * 0.0193 + var_G * 0.1192 + var_B * 0.9505
```

XYZ -> LAB

$$L^\star = 116\, f\left(\frac{Y}{Y_{\mathrm{n}}}\right) - 16$$

$$a^\star = 500\left(f\left(\frac{X}{X_{\mathrm{n}}}\right) - f\left(\frac{Y}{Y_{\mathrm{n}}}\right)\right)$$

$$b^\star = 200\left(f\left(\frac{Y}{Y_{\mathrm{n}}}\right) - f\left(\frac{Z}{Z_{\mathrm{n}}}\right)\right)$$

where, being *t=Y/Yn*:

$$f(t) = \begin{cases} \sqrt[3]{t} & \text{if } t > \delta^3 \\ \frac{t}{3\delta^2} + \frac{4}{29} & \text{otherwise} \end{cases}$$

$$\delta = \frac{6}{29}$$

CIELAB Problems

- RGB -> LAB, distance? Euclidean! (CIE76)

$$\Delta E^*_{ab} = \sqrt{(L^*_2 - L^*_1)^2 + (a^*_2 - a^*_1)^2 + (b^*_2 - b^*_1)^2}$$

- Has noticeable failures for two cases:
  1. Low lightness
  2. Sucks at blue

2. Distance Metric

- Stick with CIELAB, “fix” it by changing distance metrics
- Might have better distance calculation, but loses space...

CIE94

$$\Delta E_{94}^{*} = \sqrt{\left(\frac{\Delta L^{*}}{k_L S_L}\right)^2 + \left(\frac{\Delta C_{ab}^{*}}{k_C S_C}\right)^2 + \left(\frac{\Delta H_{ab}^{*}}{k_H S_H}\right)^2}$$

where:

$$\Delta L^{*} = L_1^{*} - L_2^{*}$$

$$C_1^{*} = \sqrt{{a_1^{*}}^2 + {b_1^{*}}^2}$$

$$C_2^{*} = \sqrt{{a_2^{*}}^2 + {b_2^{*}}^2}$$

$$\Delta C_{ab}^{*} = C_1^{*} - C_2^{*}$$

$$\Delta H_{ab}^{*} = \sqrt{{\Delta E_{ab}^{*}}^2 - {\Delta L^{*}}^2 - {\Delta C_{ab}^{*}}^2} = \sqrt{{\Delta a^{*}}^2 + {\Delta b^{*}}^2 - {\Delta C_{ab}^{*}}^2}$$

$$\Delta a^{*} = a_1^{*} - a_2^{*}$$

$$\Delta b^{*} = b_1^{*} - b_2^{*}$$

$$S_L = 1$$

$$S_C = 1 + K_1 C_1^{*}$$

$$S_H = 1 + K_2 C_1^{*}$$

CIEDE2000

$$\Delta E^*_{00} = \sqrt{\left(\frac{\Delta L'}{k_L S_L}\right)^2 + \left(\frac{\Delta C'}{k_C S_C}\right)^2 + \left(\frac{\Delta H'}{k_H S_H}\right)^2 + R_T \frac{\Delta C'}{k_C S_C}\frac{\Delta H'}{k_H S_H}}$$

Note: The formulae below should use degrees rather than radians; the issue is significant for R_T.

The k_L, k_C, and k_H are usually unity.

$$\Delta L' = L^*_2 - L^*_1$$

$$\bar{L}' = \frac{L^*_1 + L^*_2}{2} \quad \bar{C} = \frac{C^*_1 + C^*_2}{2}$$

$$a'_1 = a^*_1 + \frac{a^*_1}{2}\left(1 - \sqrt{\frac{\bar{C}^7}{\bar{C}^7 + 25^7}}\right) \quad a'_2 = a^*_2 + \frac{a^*_2}{2}\left(1 - \sqrt{\frac{\bar{C}^7}{\bar{C}^7 + 25^7}}\right)$$

$$\bar{C}' = \frac{C'_1 + C'_2}{2} \text{ and } \Delta C' = C'_2 - C'_1 \quad \text{where } C'_1 = \sqrt{a'^2_1 + b^{*^2}_1} \quad C'_2 = \sqrt{a'^2_2 + b^{*^2}_2}$$

$$h'_1 = \operatorname{atan2}(b^*_1, a'_1) \mod 360^\circ, \quad h'_2 = \operatorname{atan2}(b^*_2, a'_2) \mod 360^\circ$$

Note: The inverse tangent ($\tan^{-1}$) can be computed using a common library routine `atan2(b, a')` which usually has a rang

means that the corresponding C' is zero); in that case, set the hue angle to zero. See Sharma 2005, eqn. 7.

$$\Delta h' = \begin{cases} h'_2 - h'_1 & |h'_1 - h'_2| \le 180^\circ \\ h'_2 - h'_1 + 360^\circ & |h'_1 - h'_2| > 180^\circ, h'_2 \le h'_1 \\ h'_2 - h'_1 - 360^\circ & |h'_1 - h'_2| > 180^\circ, h'_2 > h'_1 \end{cases}$$

Note: When either C'_1 or C'_2 is zero, then $\Delta h'$ is irrelevant and may be set to zero. See Sharma 2005, eqn. 10.

$$\Delta H' = 2\sqrt{C'_1 C'_2}\sin(\Delta h'/2), \quad \bar{H}' = \begin{cases} (h'_1 + h'_2)/2 & |h'_1 - h'_2| \le 180^\circ \\ (h'_1 + h'_2 + 360^\circ)/2 & |h'_1 - h'_2| > 180^\circ, h'_1 + h'_2 < 360^\circ \\ (h'_1 + h'_2 - 360^\circ)/2 & |h'_1 - h'_2| > 180^\circ, h'_1 + h'_2 \ge 360^\circ \end{cases}$$

Note: When either C'_1 or C'_2 is zero, then $\bar{H}'$ is $h'_1 + h'_2$ (no divide by 2; essentially, if one angle is indeterminate, then use the c

in the computation of average hue".

$$T = 1 - 0.17\cos(\bar{H}' - 30^\circ) + 0.24\cos(2\bar{H}') + 0.32\cos(3\bar{H}' + 6^\circ) - 0.20\cos(4\bar{H}' - 63^\circ)$$

$$S_L = 1 + \frac{0.015\left(\bar{L}' - 50\right)^2}{\sqrt{20 + \left(\bar{L}' - 50\right)^2}} \quad S_C = 1 + 0.045\bar{C}' \quad S_H = 1 + 0.015\bar{C}'T$$

$$R_T = -2\sqrt{\frac{\bar{C}'^7}{\bar{C}'^7 + 25^7}}\sin\left[60^\circ \cdot \exp\left(-\left[\frac{\bar{H}' - 275^\circ}{25^\circ}\right]^2\right)\right]$$

Summary

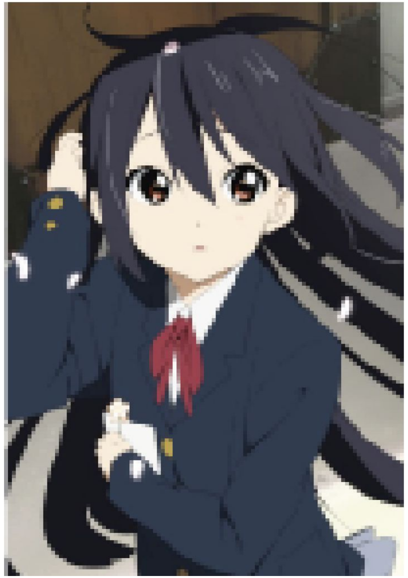
- Take your pick of color space, then try Euclidean
    - CIELAB, CAM02-UCS, CAM16-UCS
- Or try an actual metric:
    - CIE94, CIEDE2000
- Uncountable others
- Space has benefits over metric, e.g. centroid
    - But metric is often easier to compute
    - Can use gradient descent for “centroid” if metric differentiable
    - Again, more on this later

What’s the point?

ANSI Color Codes

- Image -> text
- catimg
- Some terminals support 3-byte “true color”
- Neither Vim’s AnsiEsc nor Vim’s Colorizer can render these
- Thus, have to use 255 color mode

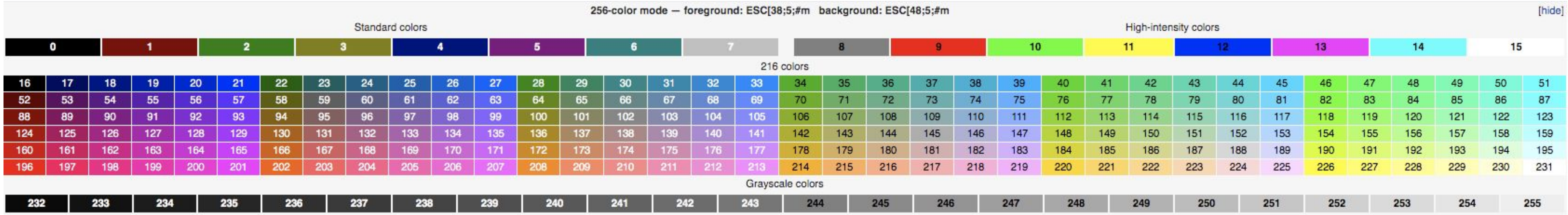
a. original image



b. catimg 24-bit “true color”

c. catimg 8-bit

The Algorithm



- Downscale image somehow (averaging window, bicubic, bilinear, etc.)
- Convert color to nearest ANSI color
  - Find closest color!
  - `catimg` uses YUV color space + Euclidean which explains the poor quality
- Keep this problem in mind, it'll come back later...

Food for thought...

Sound familiar?

- Ad-hoc formulas
- Problem is by definition not *a priori*
- Empirically determined
- Deep learning?

- Datasets of color1, color2 pairs and corresponding distance
- Neural network essentially color space, project color -> vector
- Backpropagate on distance
- Uniform color space???
- Implication of NNs modeling brain's neural network...

Sources

- My implementation
- Color Vision - Wikipedia
- Why are red, yellow, and blue the primary colors in painting but computer screens use red, green, and blue?
- MacAdam ellipse
- USEFUL: Color math and programming code examples - EasyRGB
    - scikit-image color functions
- CIELAB color space - Wikipedia

Sources

- Color difference - Wikipedia
- Color difference Delta E - A survey
- The development of the CIE 2000 colour-difference formula: CIEDE2000
- VERY USEFUL: http://www2.ece.rochester.edu/~gsharma/ciede2000/
  - Organized presentation of formulas, detailed testcases, a bit of mathematical analysis
- Delta-E Calculator (broken for CIE2000, works for CIE76 and CIE94)
- ANSI escape code - Wikipedia

Appendix

Why 2*255 in "Basis of the Subtractive Color System"?

- Need coefficients to be nonnegative to make sense
- Because of LP, extrema occurs at simplex

```
from itertools import product
import numpy as np

w = np.array([1, 1, 1])

m = np.array([[0, 1, 1],
              [1, 0, 1],
              [1, 1, 0]])
minv = np.linalg.inv(m)

for corner in product((0, 255), repeat=3):
    u = minv@(2*255*w - np.array(corner))
    print(corner, u, min(u), max(u))
```