# IN433
# Multimedia Processing

**Dr. Zein Al Abidin IBRAHIM**

2025 / 2026

Multimedia

# COLOR THEORY

# Outline

- Colors
- Color models and color spaces
- Conversion between color spaces

# Spectrum of White Light

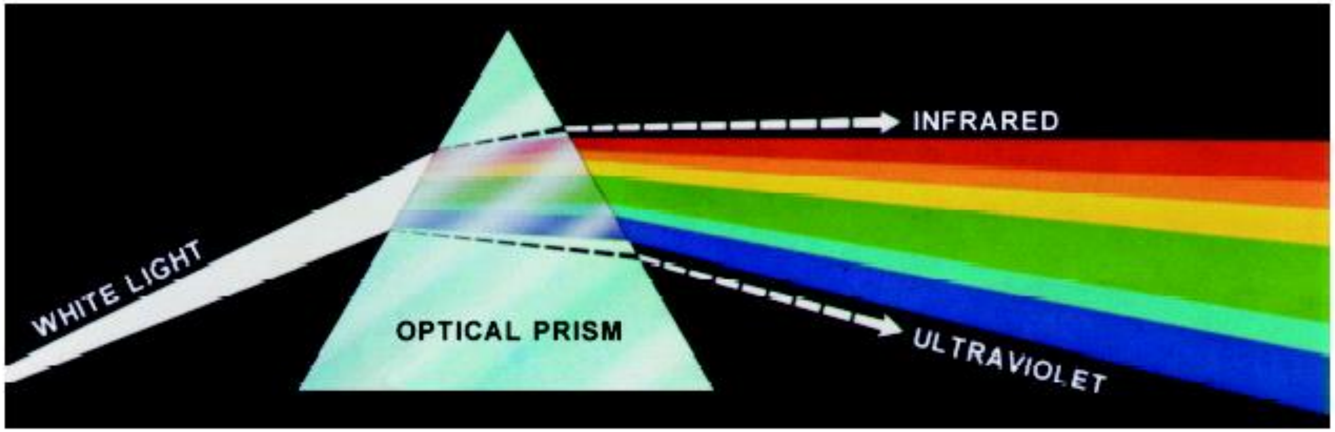
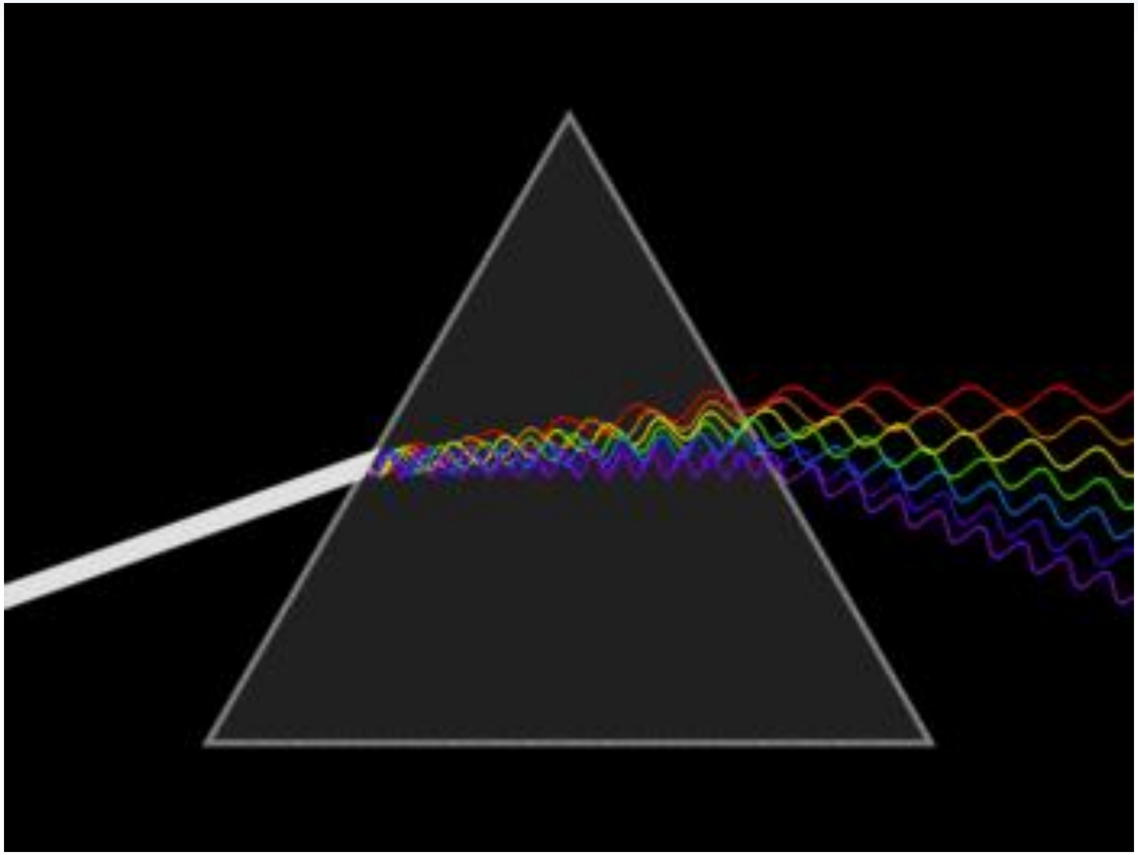


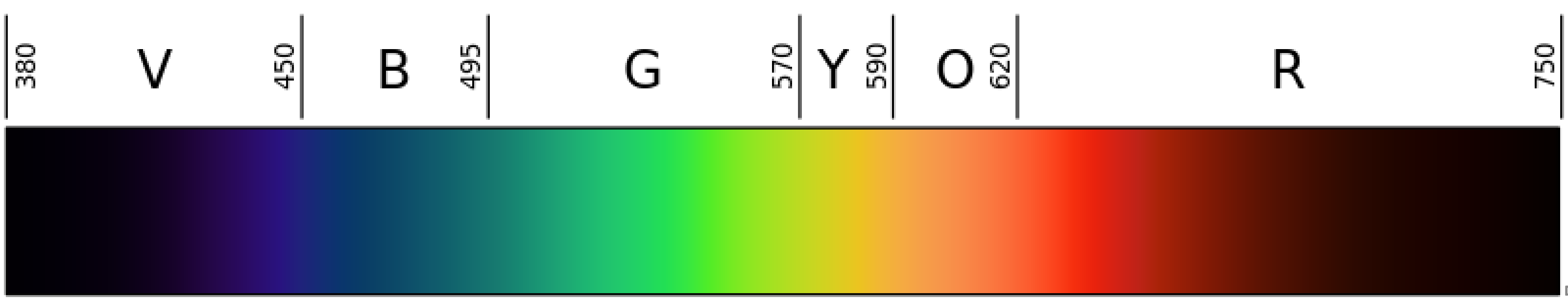

**FIGURE 6.1** Color spectrum seen by passing white light through a prism. (Courtesy of the General Electric Co., Lamp Business Division.)

1666 Sir Isaac Newton, 24 year old, discovered white light spectrum.

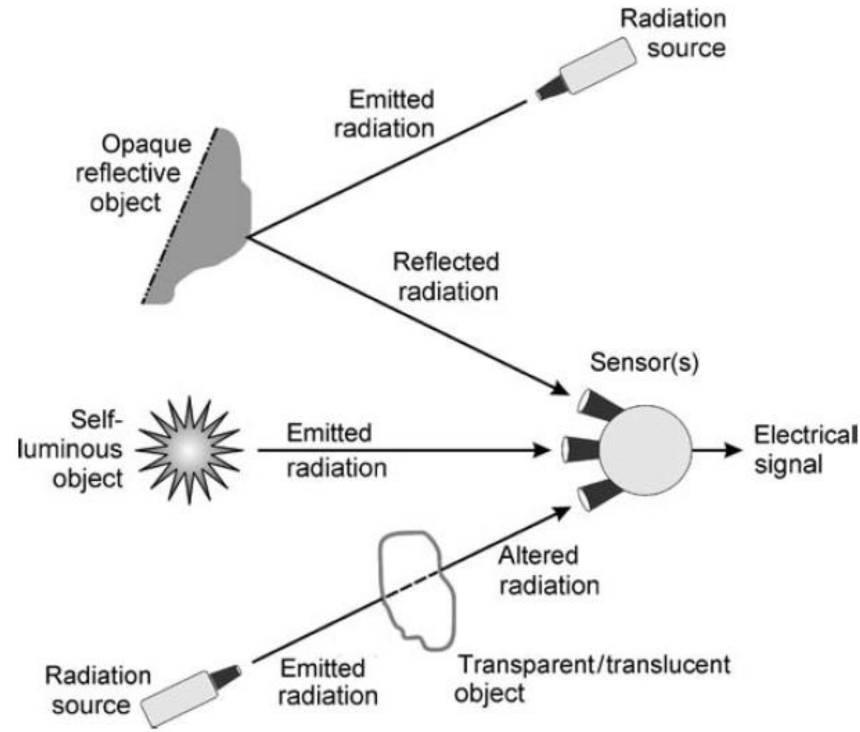
- Light described as electromagnetic waves or particles called photons
- Each has a wavelength
- Newton proved sunlight is composed of many different "colors" of light than one
- Colors are not in the light but the effect of light on the visual system

# Spectrum of White Light



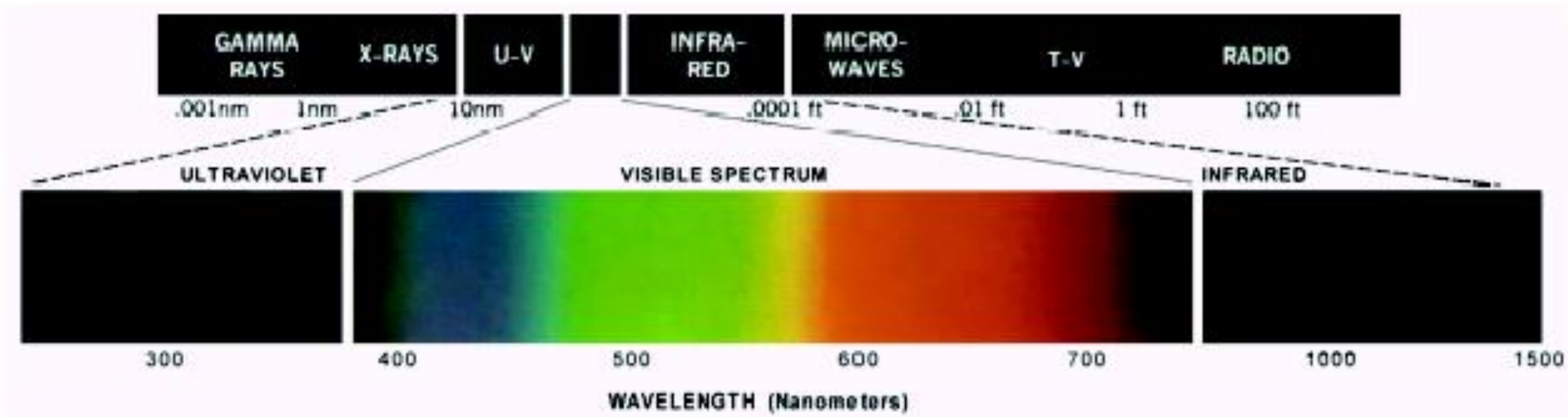


# Types of Images




- *Reflection Images*: Common type of images where lights are reflected from the surface of objects
- *Emission Images: Images that emit lights in the visible light range such as stars or beyond the visible range such as infrared images*
- *Absorption Images: result of radiation passing through objects such as X-Ray images*

# Electromagnetic Spectrum




Visible light wavelength: from around 400 to 700 nm

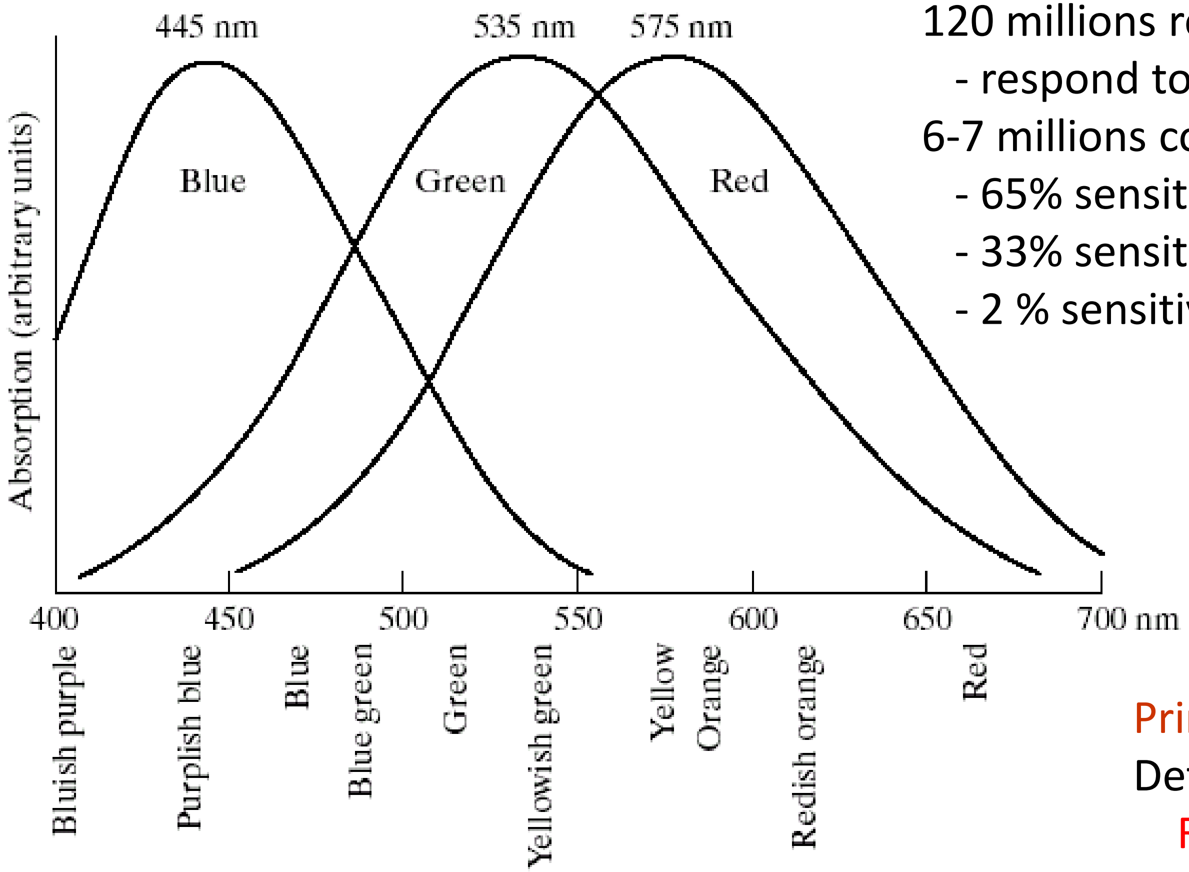
1. For an achromatic (monochrome) light source, there is only 1 attribute to describe the quality: intensity (or amount).
2. For a chromatic light source, there are 3 attributes to describe the quality:

Radiance = total amount of energy flow from a light source (Watts)

Luminance = amount of energy received by an observer (lumens)

Brightness = intensity

# Sensitivity: Human Eye



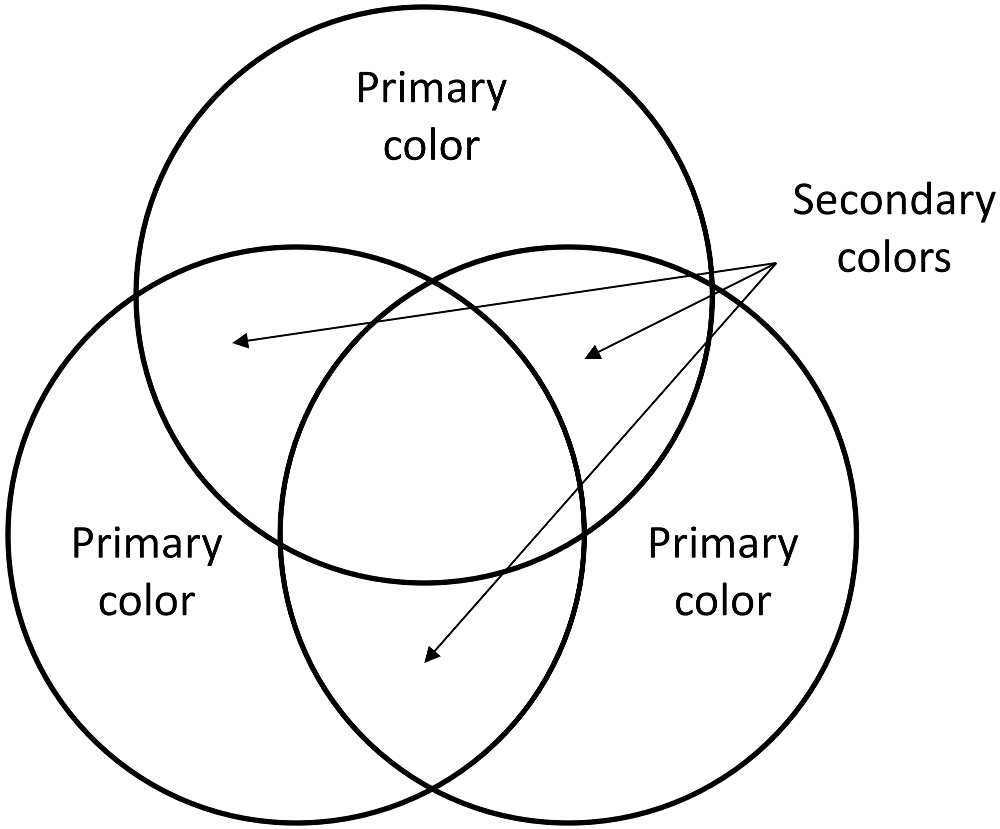

120 millions rods in a human eye
- respond to changes in illuminations

6-7 millions cones in a human eye
- 65% sensitive to Red light
- 33% sensitive to Green light
- 2 % sensitive to Blue light

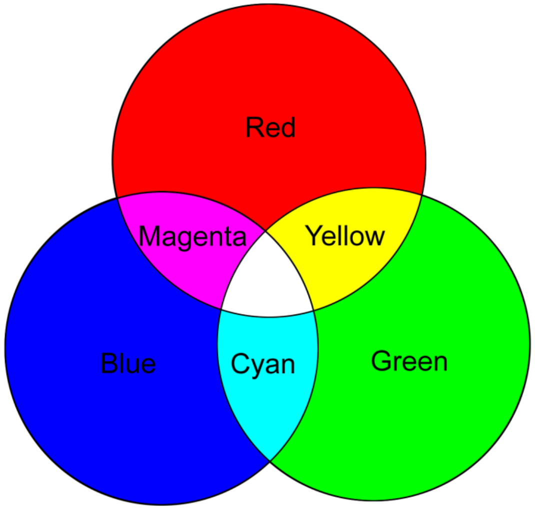
Primary colors:
Defined CIE in 1931
- Red = 700 nm
- Green = 546.1nm
- Blue = 435.8 nm

CIE = Commission Internationale de l'Eclairage
(The International Commission on Illumination)

# Primary & Secondary Colors

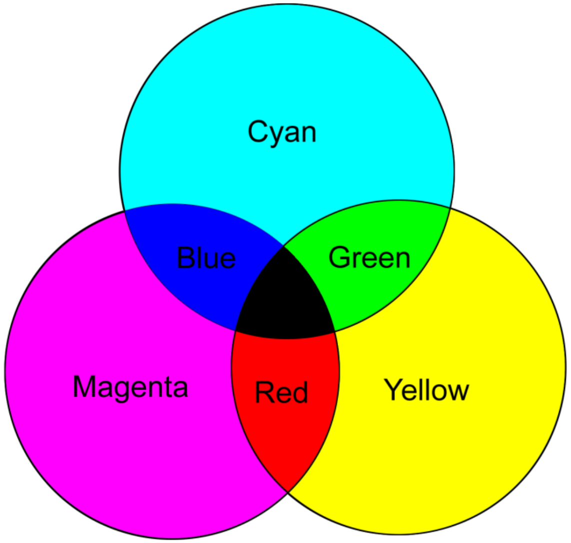


# Primary & Secondary Colors




Additive primary colors: RGB use in the case of light sources such as color monitors

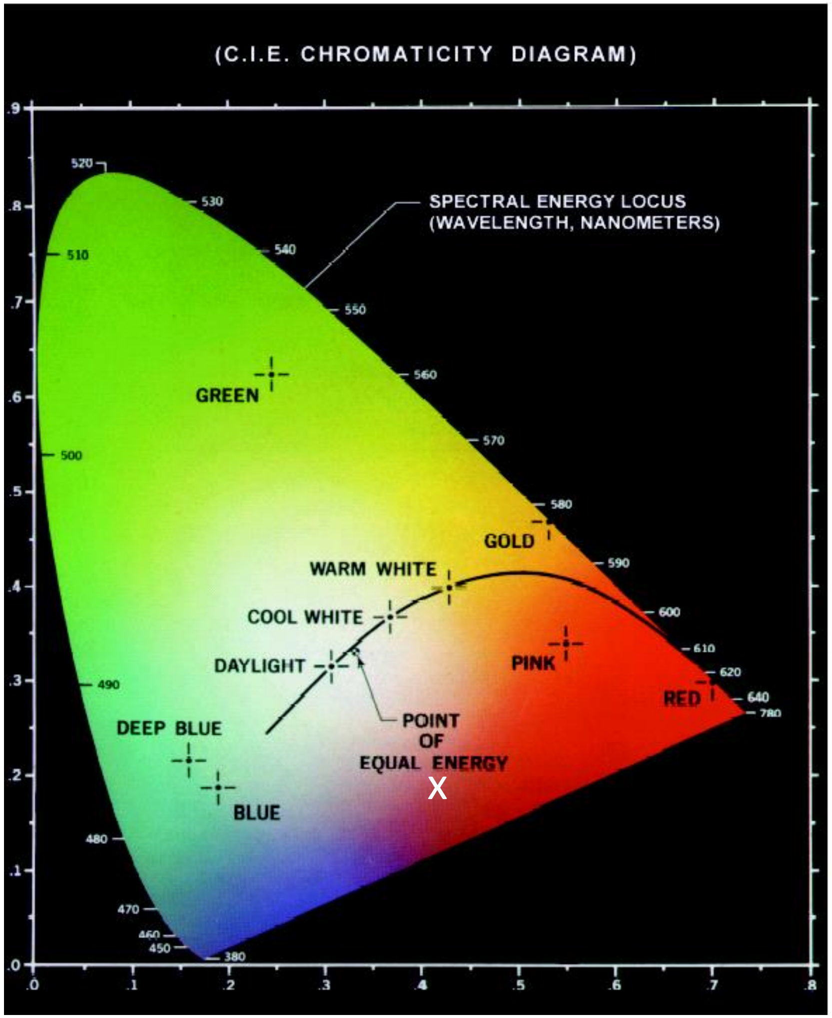
RGB add together to get white




Subtractive primary colors: CMY use in the case of pigments in printing devices

White subtracted by CMY to get Black

# CIE Chromaticity Diagram




A color is then specified by its trichromatic coefficients:

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

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

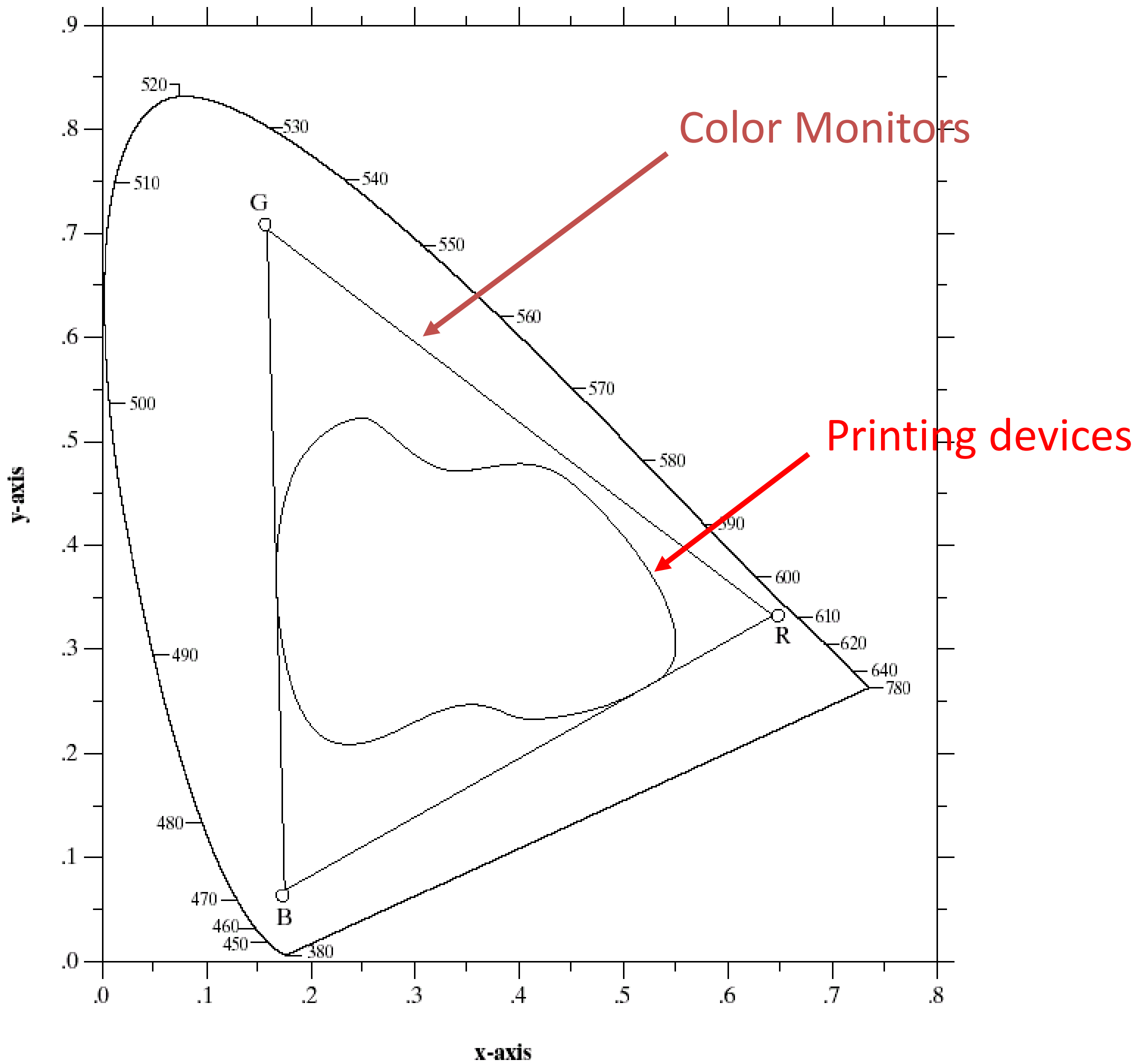
$$z = \frac{Z}{X+Y+Z}$$

$$x + y + z = 1$$

Points on the boundary are fully saturated colors

x (red), y (green) and z = 1-x-y → blue

# Color Gamut



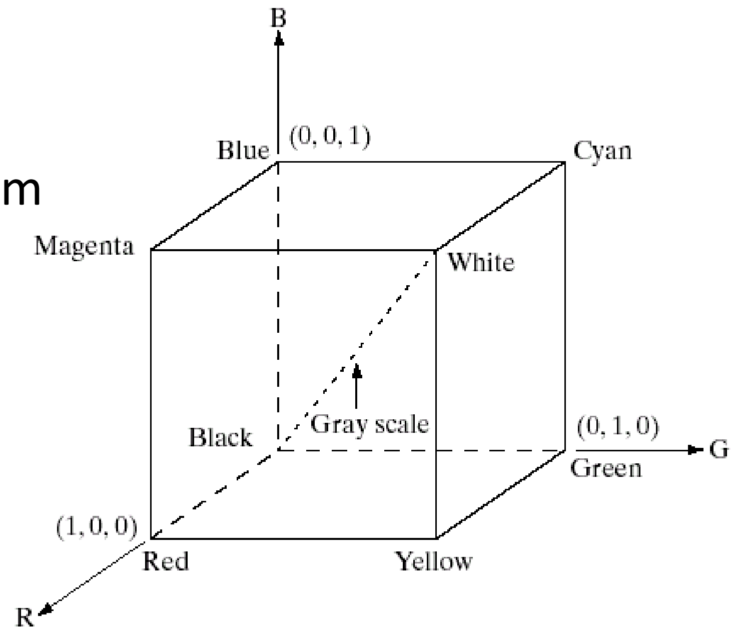
Color Monitors
Printing devices
G
R
B
520
530
540
550
560
570
580
590
600
610
620
640
780
510
500
490
480
470
460
450
380
.0
.1
.2
.3
.4
.5
.6
.7
.8
.9
y-axis
x-axis

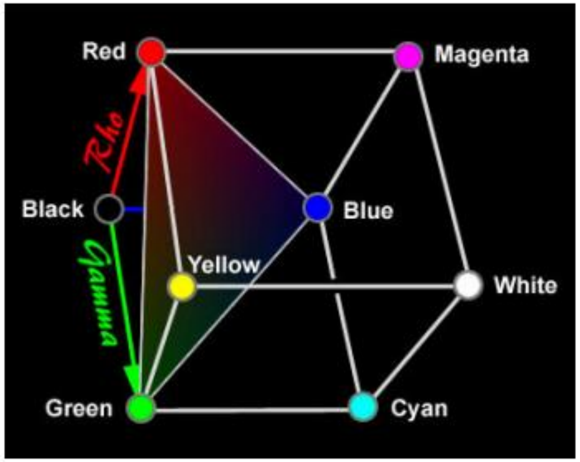
# Color Models

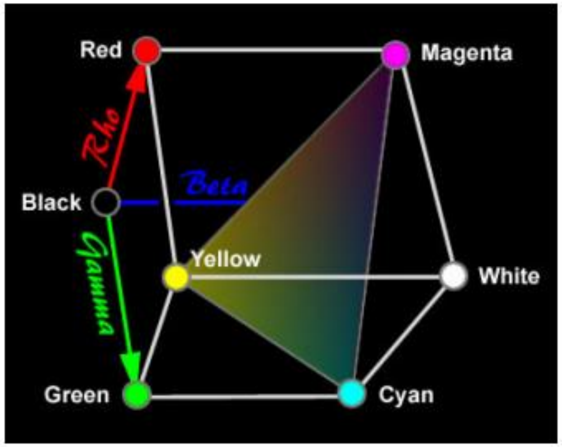
- ➢ Facilitate the specification of colors in some standard way
- ➢ A coordinate system where each color is represented by a point.
- ➢ Several color models exist today:
  - – RGB color model→ additive primaries
  - – HSI color model
    - HSV → Hue, Saturation, Value
    - HSL → Hue, Saturation, Lightness
    - HCI → Hue, Chroma, Intensity
    - HVC → Hue, Value, Chroma
    - HSD → Hue, Saturation, Darkness
  - – CMY and CMYK color models

# RGB Color Model

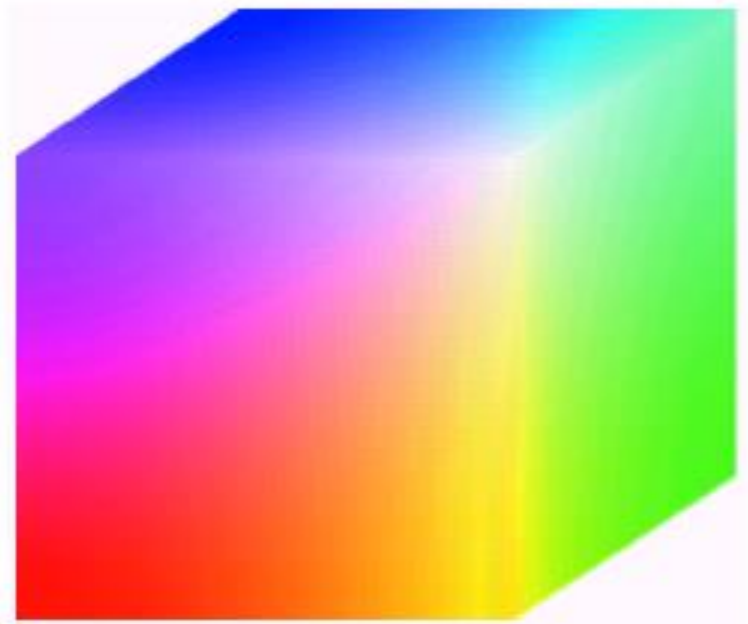
- Purpose of color models: to facilitate the specification of colors in some standard
- RGB color models:
  - based on cartesian coordinate system



# RGB Color Model



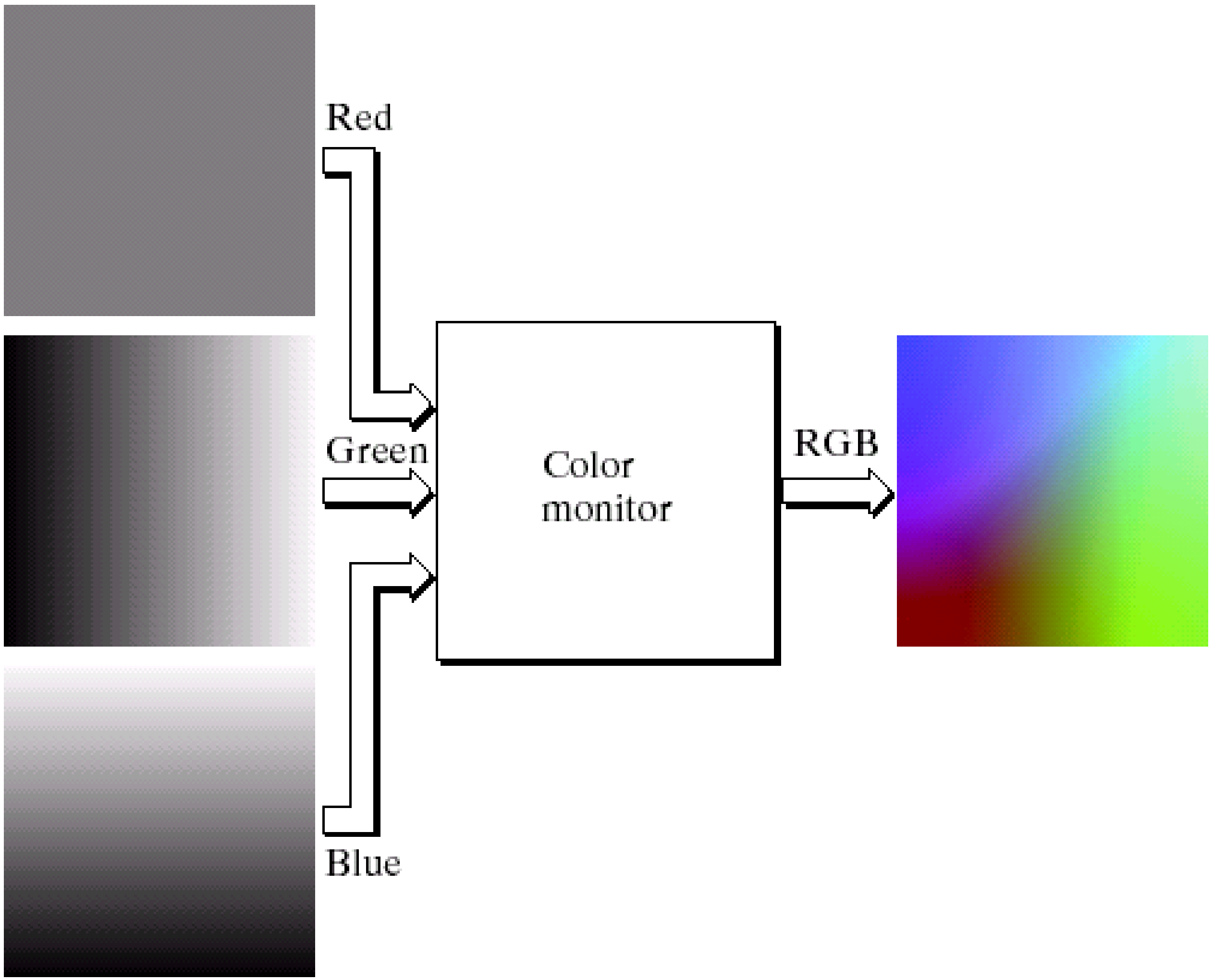
R = 8 bits
G = 8 bits
B = 8 bits

Color depth 24 bits
= 16777216 colors

$(R = 0)$ $(G = 0)$ $(B = 0)$

Hidden faces of the cube

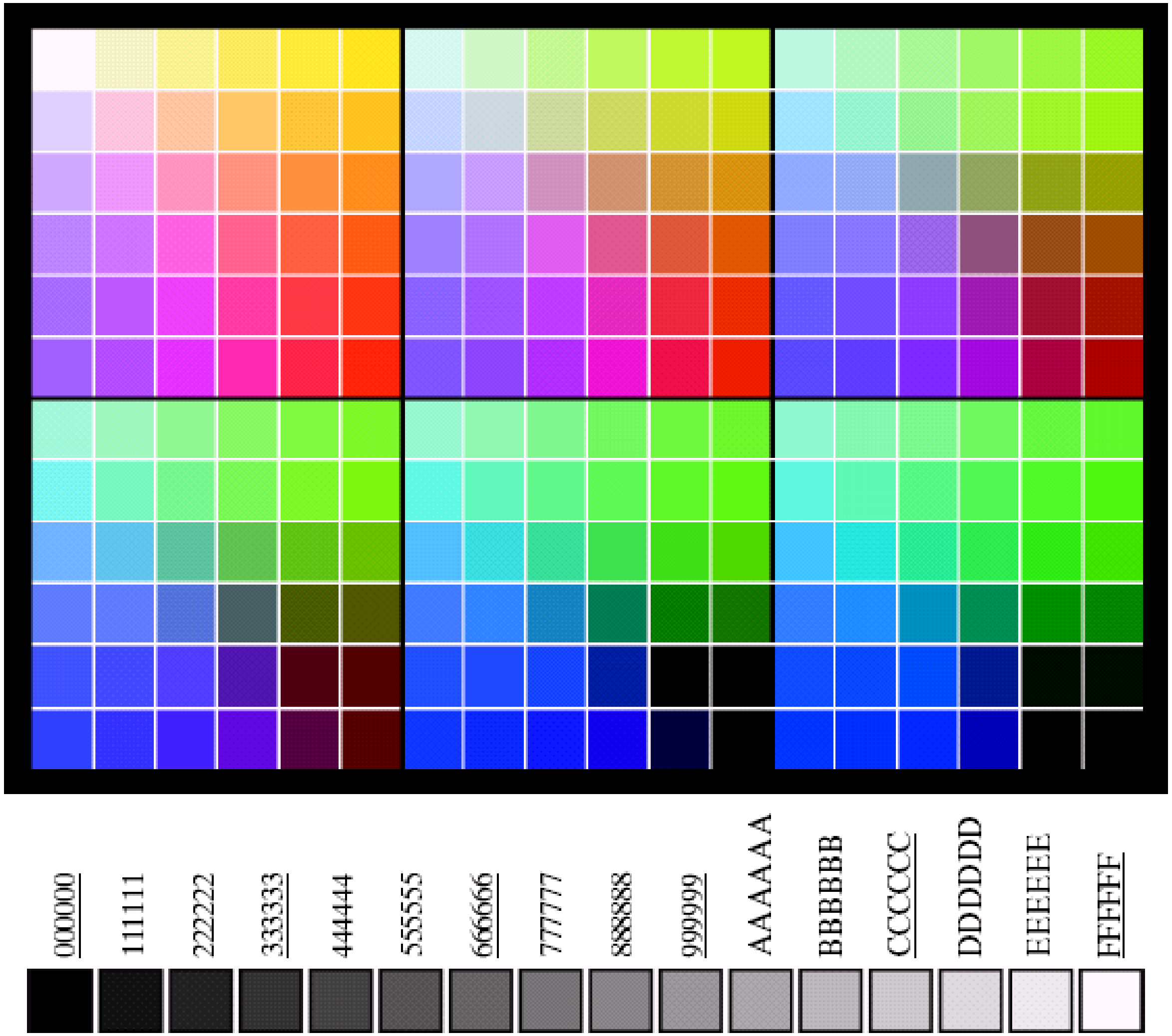
# RGB Color Model



# Safe RGB Colors

Safe RGB colors: a subset of RGB colors.

There are 216 colors common in most operating systems.



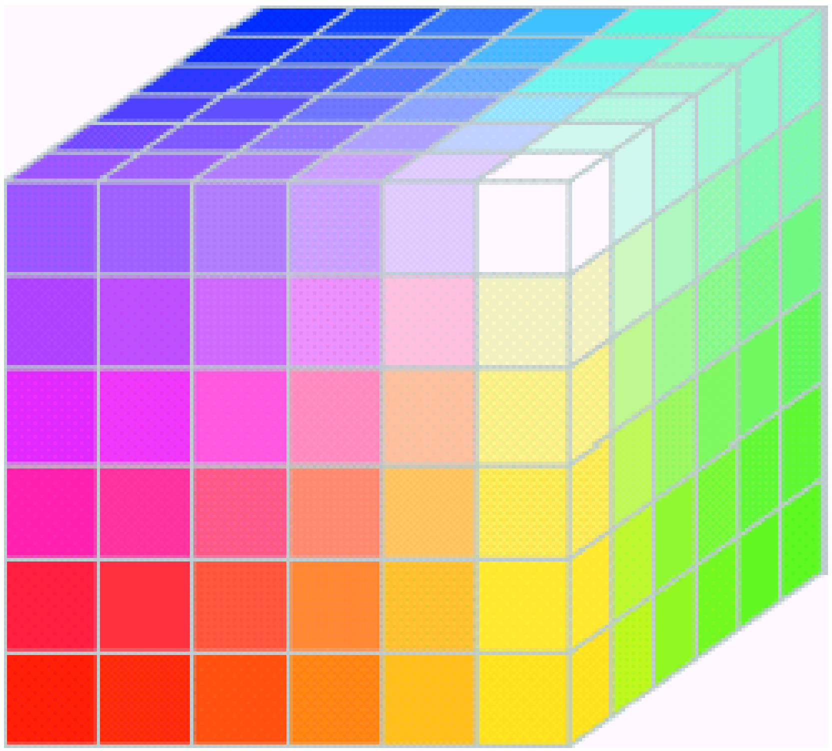

a
b

**FIGURE 6.10** (a) The 216 safe RGB colors. (b) All the grays in the 256-color RGB system (grays that are part of the safe color group are shown underlined).

# RGB Safe-Color Cube

| Number System | Color Equivalents | | | | | |
|---|---|---|---|---|---|---|
| Hex | 00 | 33 | 66 | 99 | CC | FF |
| Decimal | 0 | 51 | 102 | 153 | 204 | 255 |

**TABLE 6.1**
Valid values of each RGB component in a safe color.



The RGB Cube is divided into 6 intervals on each axis to achieve the total $6^3 = 216$ common colors.

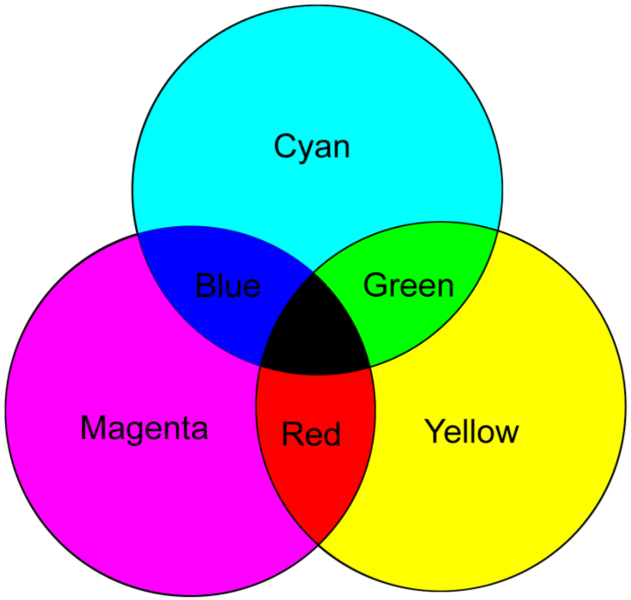
However, for 8 bit color representation, there are the total 256 colors. Therefore, the remaining 40 colors are left to OS.

# CMY & CMYK Color Models

C = Cyan
M = Magenta
Y = Yellow
K = Black

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

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




$$K = 1\text{-}\max(R', G', B')$$

$$R' = R/255 \qquad C = (1\text{-}R'\text{-}K) / (1\text{-}K)$$

$$G' = G/255 \qquad M = (1\text{-}G'\text{-}K) / (1\text{-}K)$$

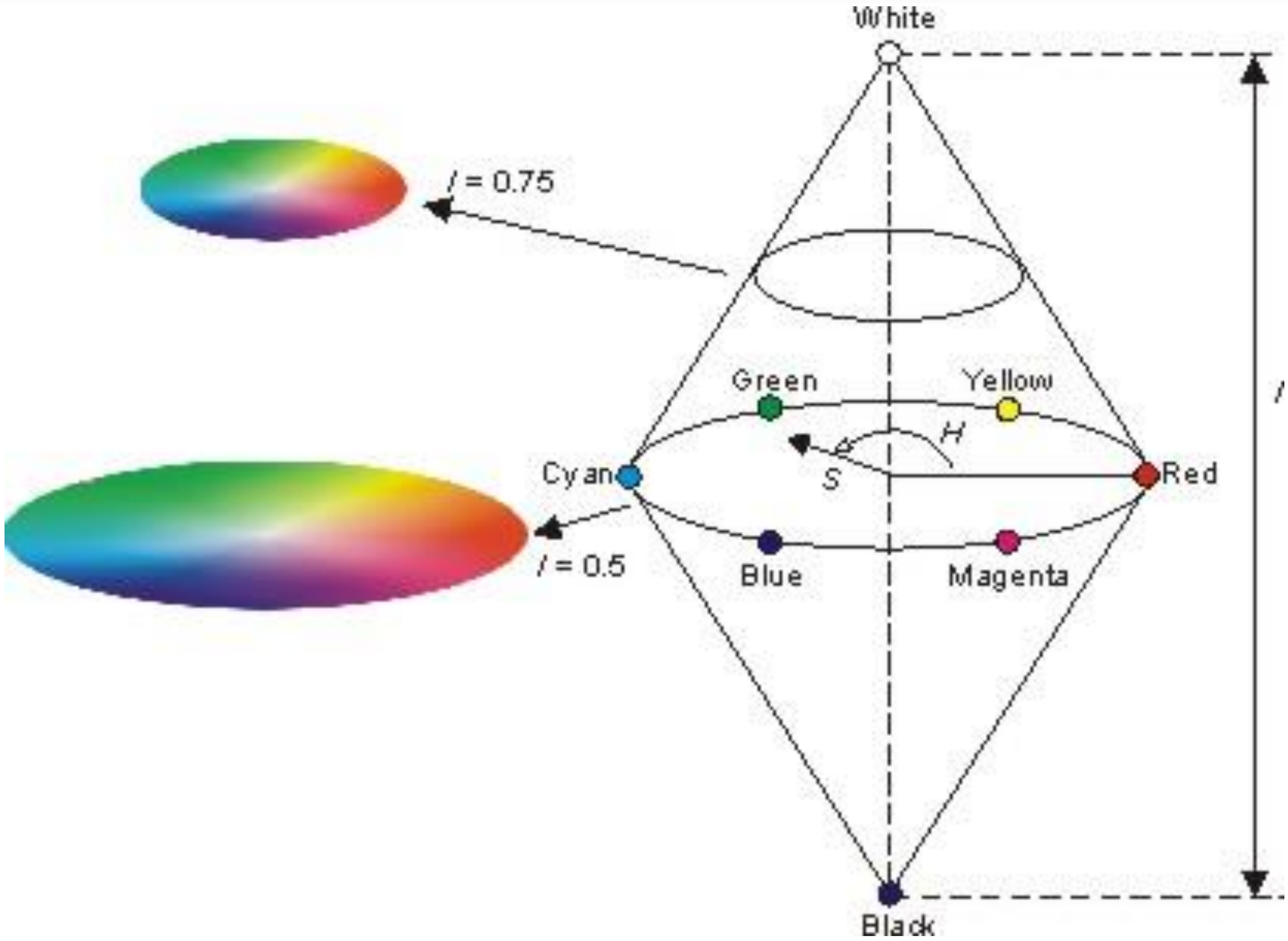
$$B' = \mathrm{B}/255 \qquad Y = (1\text{-}B'\text{-}K) / (1\text{-}K)$$

# HIS Color Model

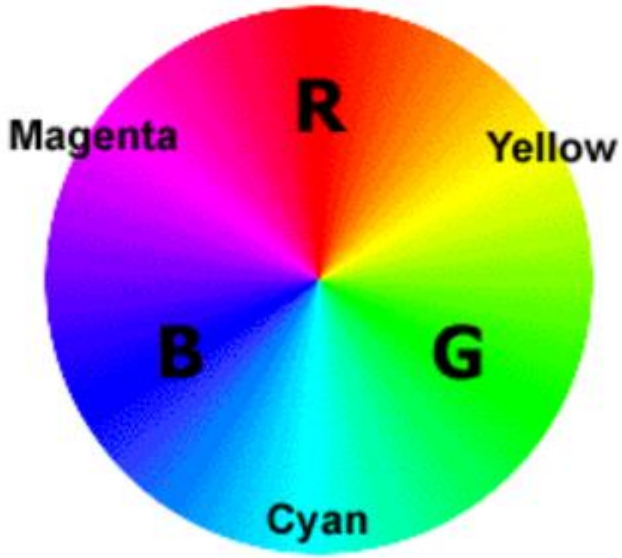
RGB, CMY models are not good for human interpreting

- **Hue:**
  - dominant color corresponding to the dominant wavelength of mixture light wave (red, orange, yellow,...)
- **Saturation:**
  - Relative purity or amount of white light mixed with a hue (inversely proportional to amount of white light added)
- **Brightness:**
  - Intensity
- Hue + Saturation = Chromaticity

# HSI



# HSI

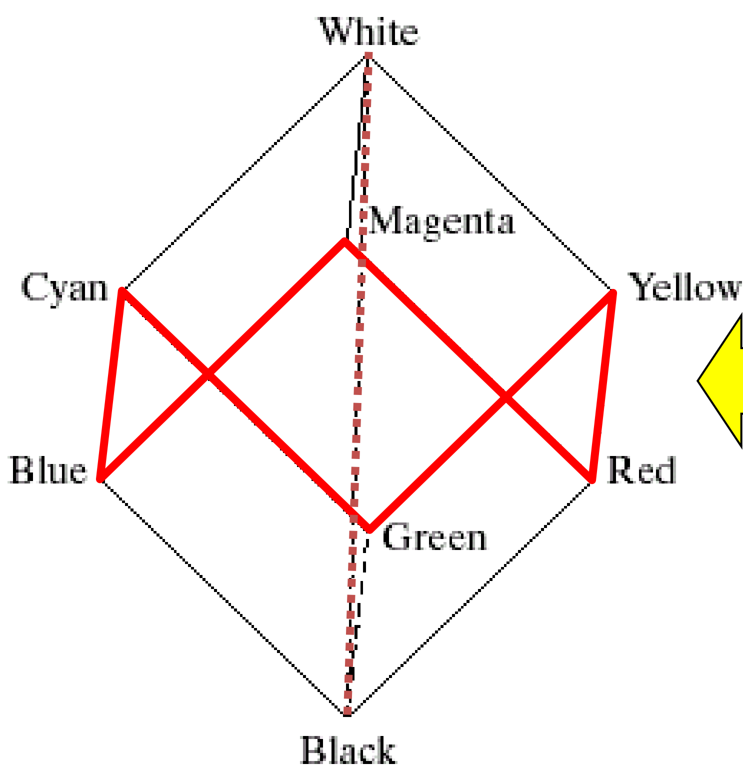




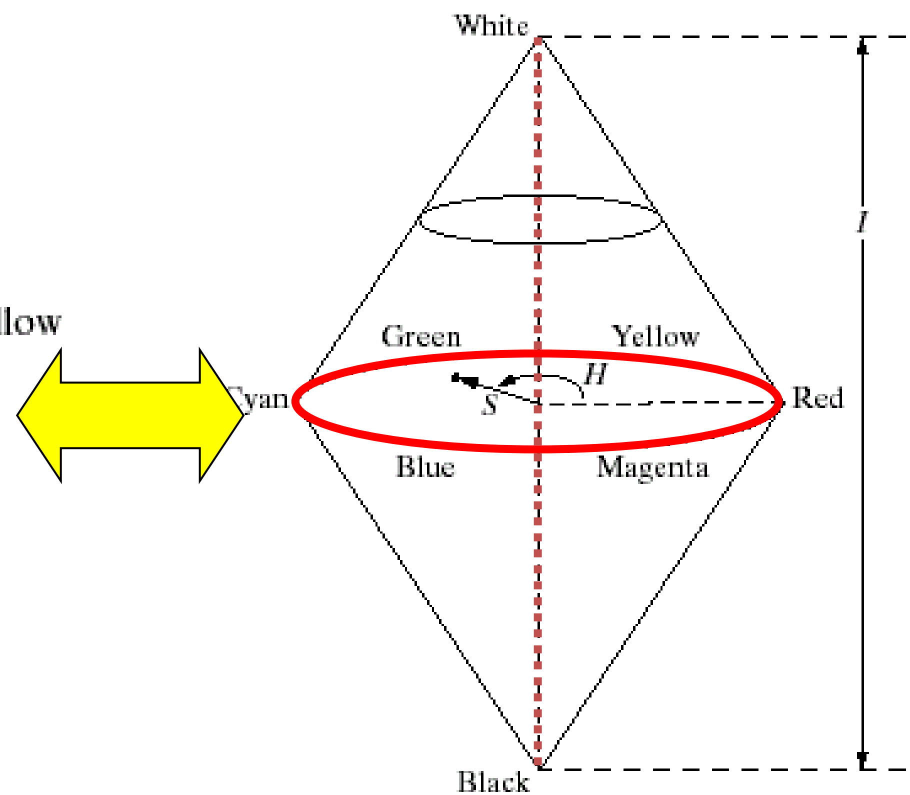
How much gray added
Goes from unsaturated to saturated


How much lightness or darkness added.
Goes from black to white

# Relation: RGB & HSI



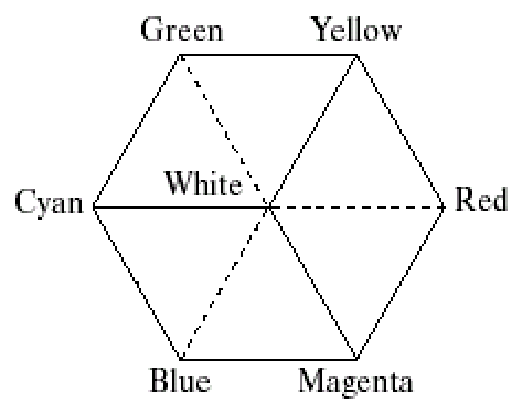

RGB

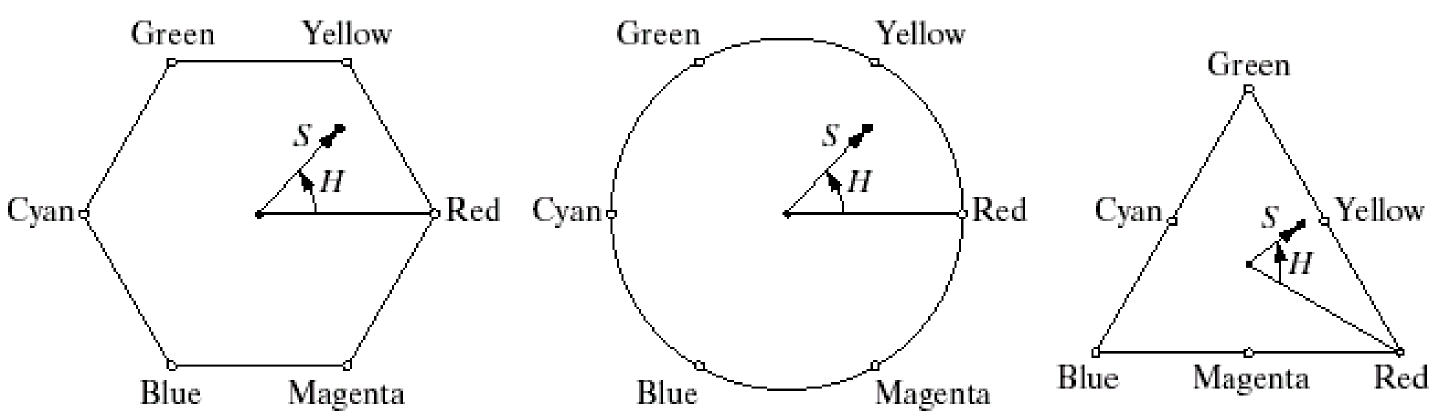



HSI

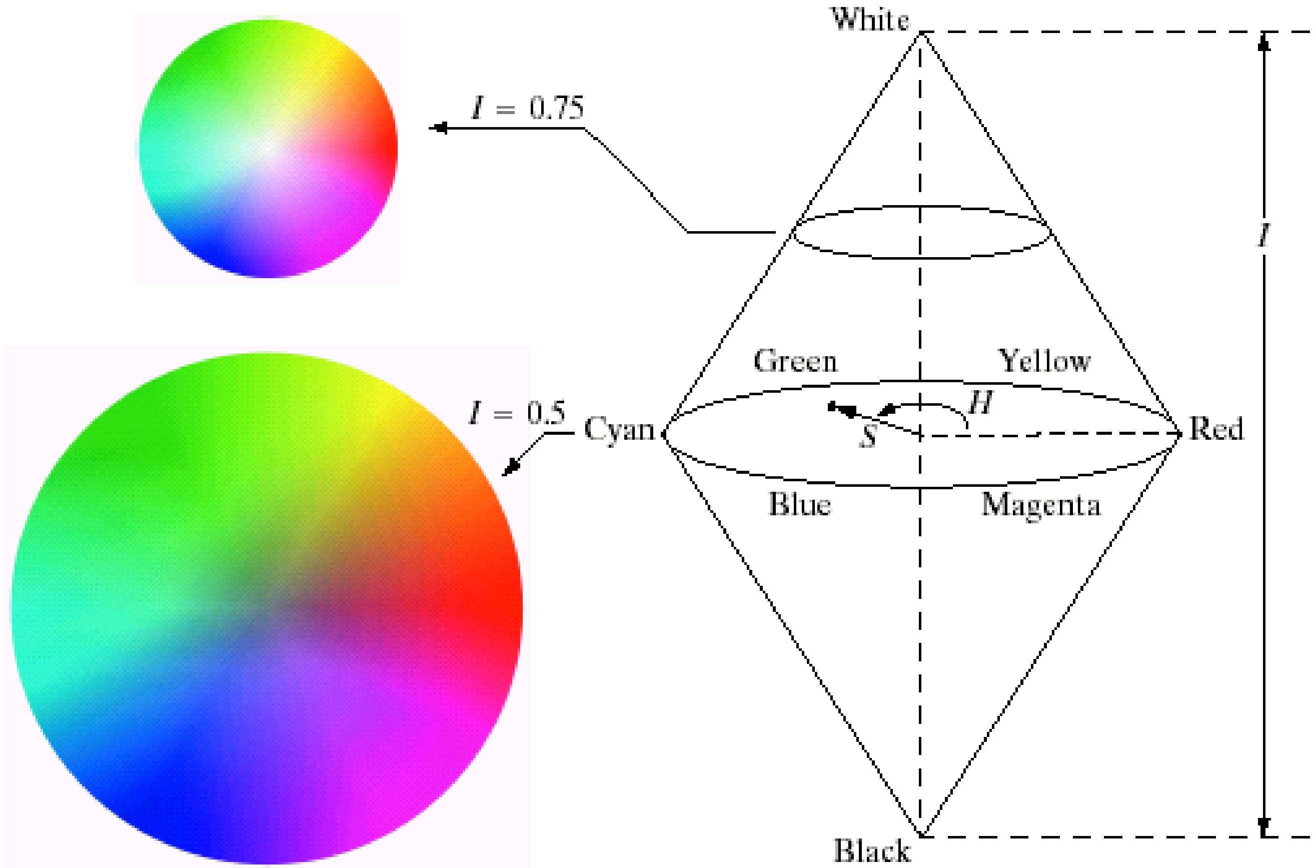
# Hue & Saturation on Color Planes

1. A dot in the plane is an arbitrary color
2. Hue is an angle from a red axis.
3. Saturation is a distance to the point.



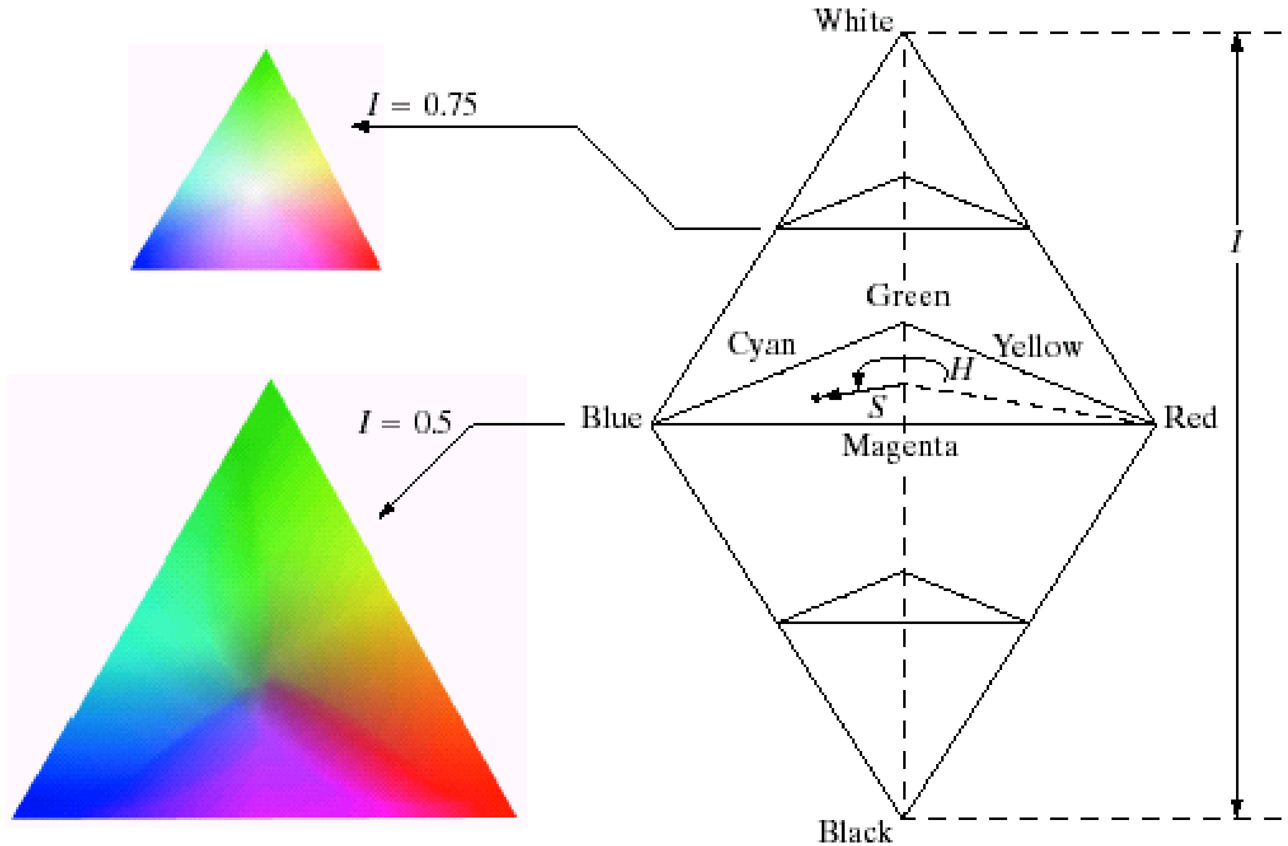



# HIS Color Model



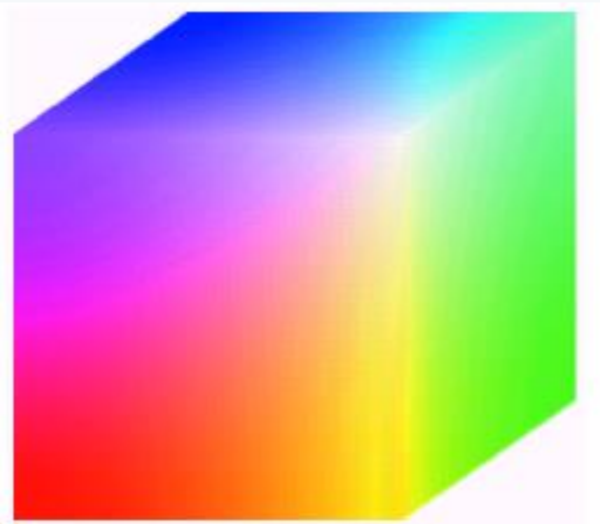

Intensity is given by a position on the vertical axis.

# HIS Color Model

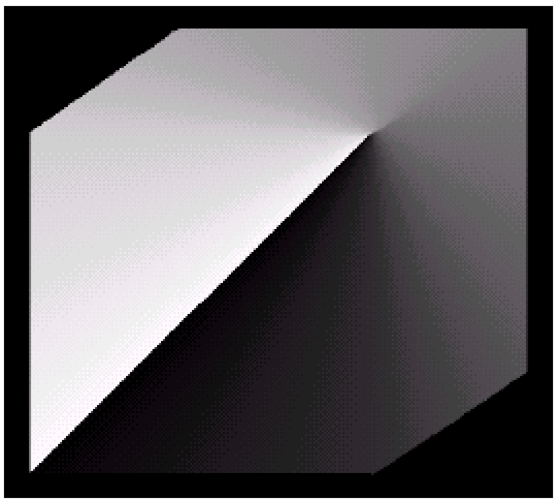



Intensity is given by a position on the vertical axis.

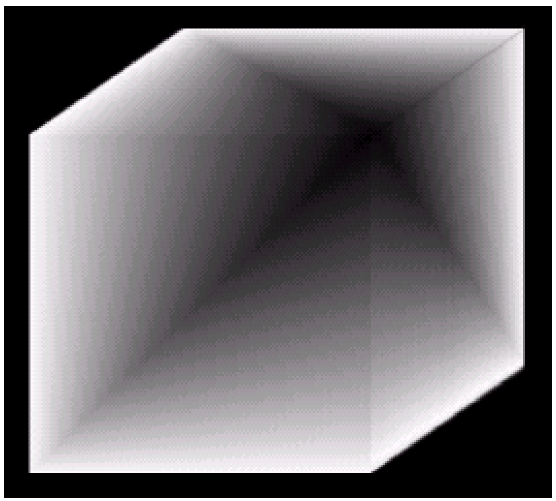
# Example: HSI Components of RGB Cube



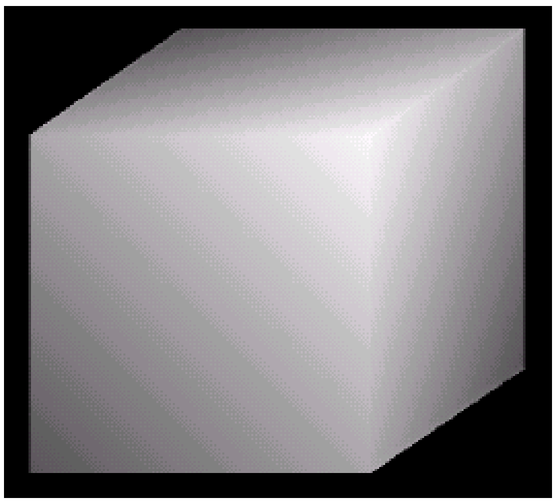
RGB Cube



Hue



Saturation



Intensity

# Converting between XYZ & RGB

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

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 0.41847 & -0.15866 & -0.082835 \\ -0.091169 & 0.25243 & 0.015708 \\ 0.00092090 & -0.0025498 & 0.17860 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

# Converting Colors from RGB to HSI

$$H = \begin{cases} \theta & \text{if } B \le G \\ 360 - \theta & \text{if } B > G \end{cases}$$

$$\theta = \cos^{-1}\left\{ \frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\left[(R-G)^2+(R-B)(G-B)\right]^{1/2}} \right\}$$

$$S = 1 - \frac{3}{R+G+B}$$

$$I = \frac{1}{3}(R+G+B)$$

# Converting Colors from HSI to RGB

RG sector: $0 \le H < 120$

$$R = I\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right]$$

$$B = I(1-S)$$

$$G = 1-(R+B)$$

GB sector: $120 \le H < 240$

$$H = H - 120$$

$$R = I(1-S)$$

$$G = I\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right]$$

$$B = 1-(R+G)$$

BR sector: $240 \le H \le 360$

$$H = H - 240$$

$$B = I\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right]$$

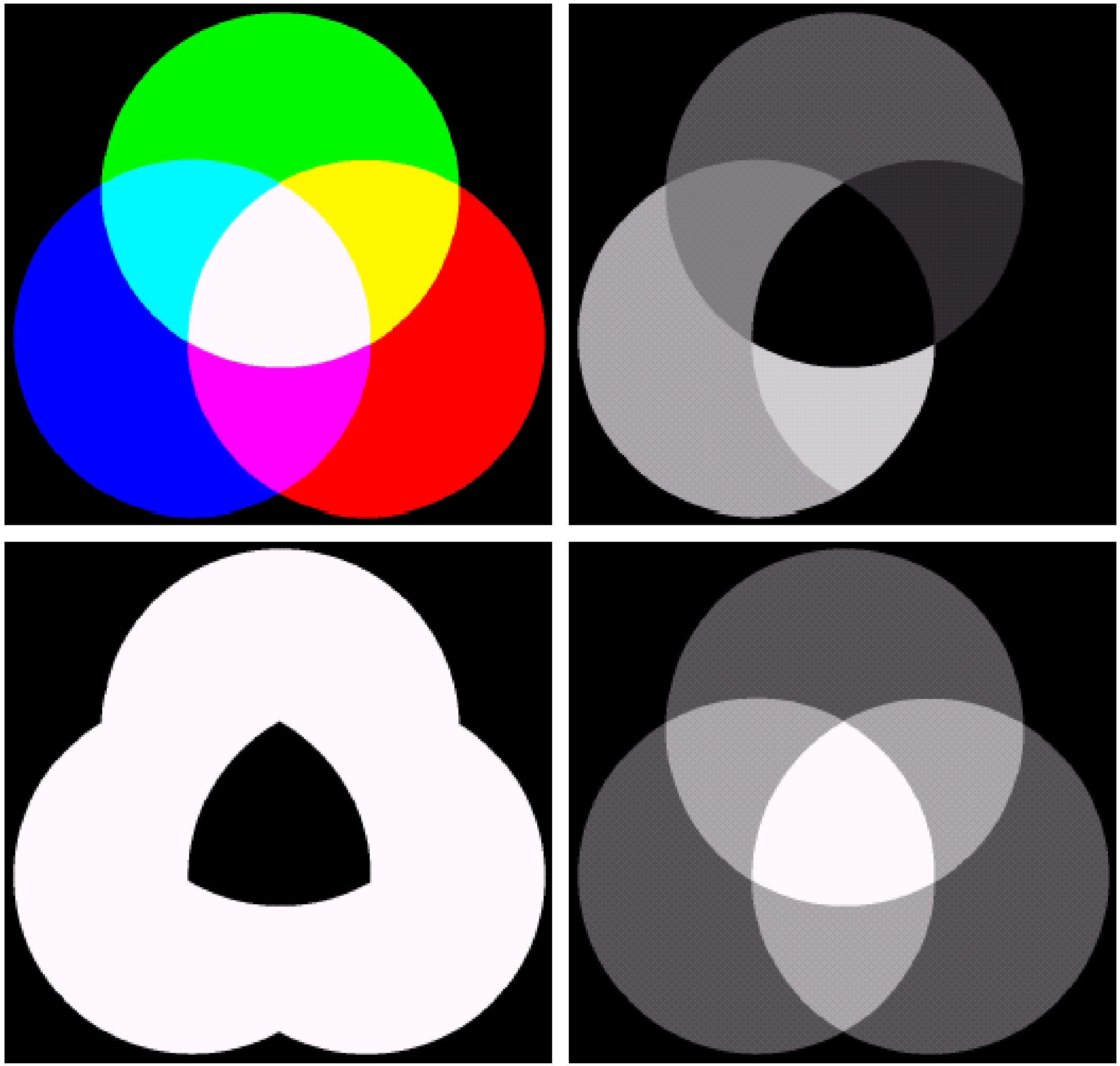
$$G = I(1-S)$$

$$R = 1-(G+B)$$

# Example: HSI Components of RGB Colors

RGB Image

Hue



Saturation

Intensity

# Color Models in Video

- **Video Color Transforms**
  - Largely derive from older analog methods of coding color for TV. Luminance is separated from color information.
  - For example, a matrix transform method called YIQ is used to transmit TV signals in North America and Japan.
  - This coding also makes its way into VHS video tape coding in these countries since video tape technologies also use YIQ.
  - In Europe, video tape uses the PAL or SECAM codings, which are based on TV that uses a matrix transform called YUV.
  - Finally, digital video mostly uses a matrix transform called YCbCr that is closely related to YUV

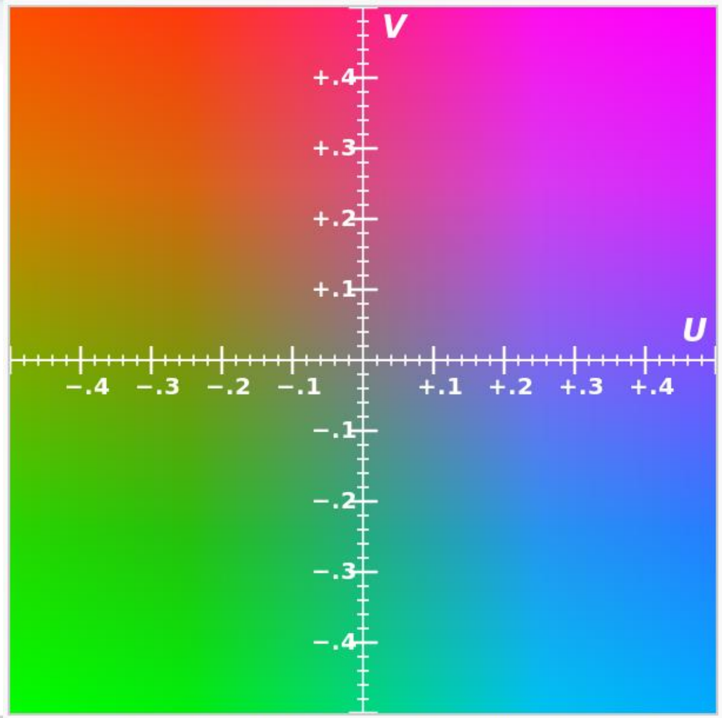
# YUV Color Model

- Y → Luminance, and U and V for chrominance (color)
- Y' → stands for the luma component (brightness)
- Y’ → (‘)  denote gamma compression
- Invented to be compatible with black and white devices
- If device is black and white → take the value of Y’

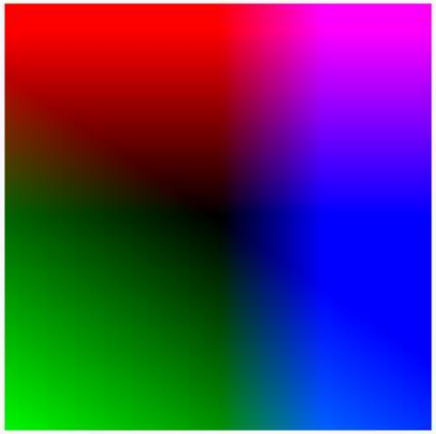
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{bmatrix} \begin{bmatrix} Y' \\ U \\ V \end{bmatrix}$$

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

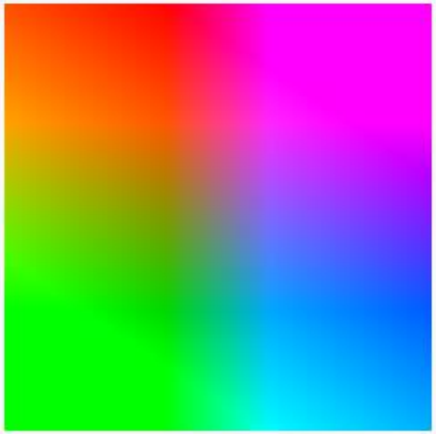
# YUV Color Model



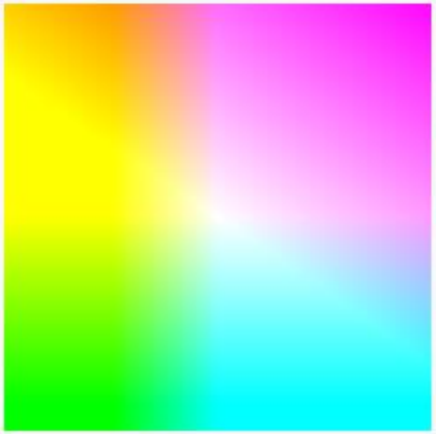

Example of U-V color plane, Y' value = 0.5



Y' value of 0



Y' value of 0.5



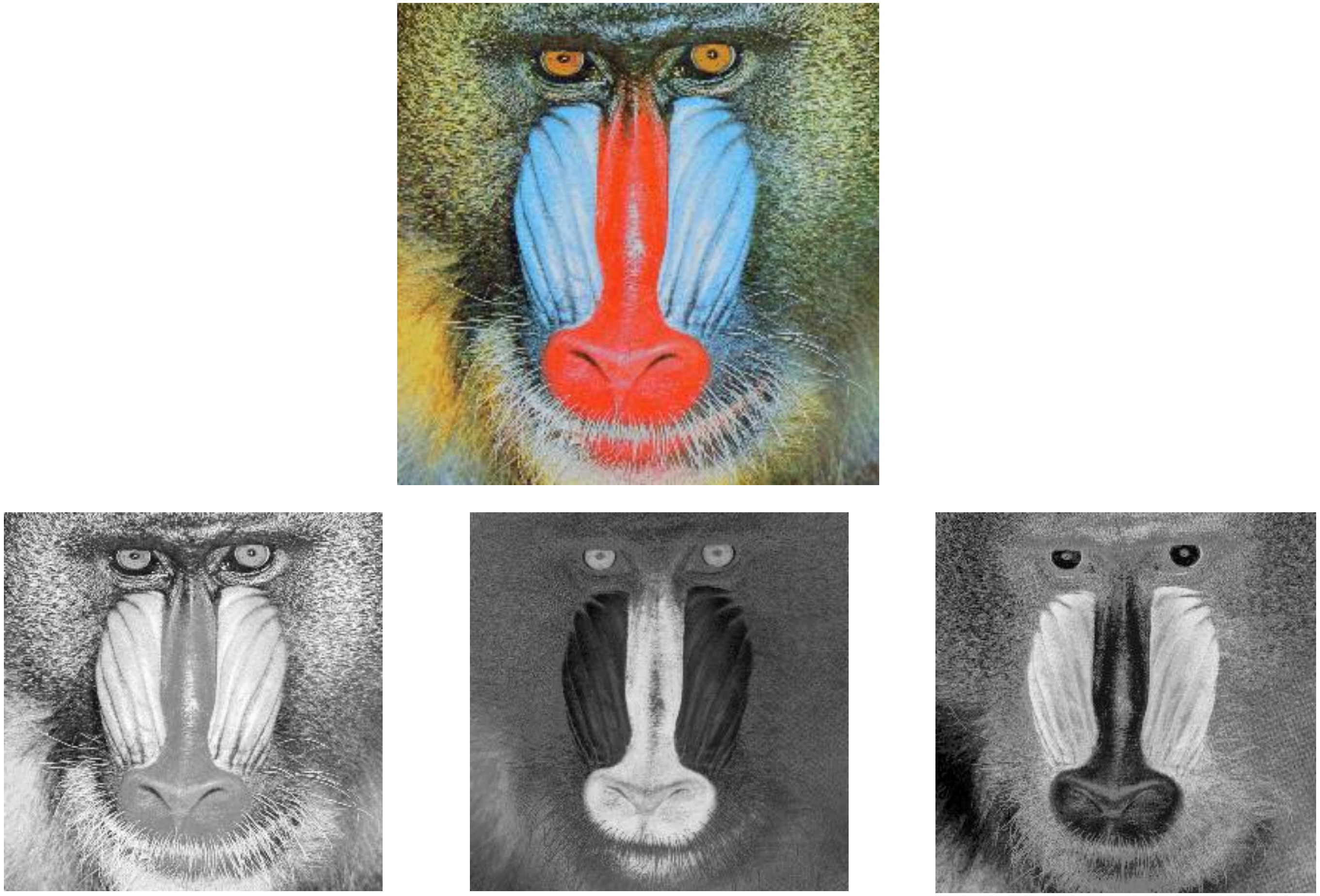
Y' value of 1

# YUV Color Model

Y’UV decomposition of color image (top image)

(a) is original color image;

(b) is Y ’; (c,d) are (U, V)



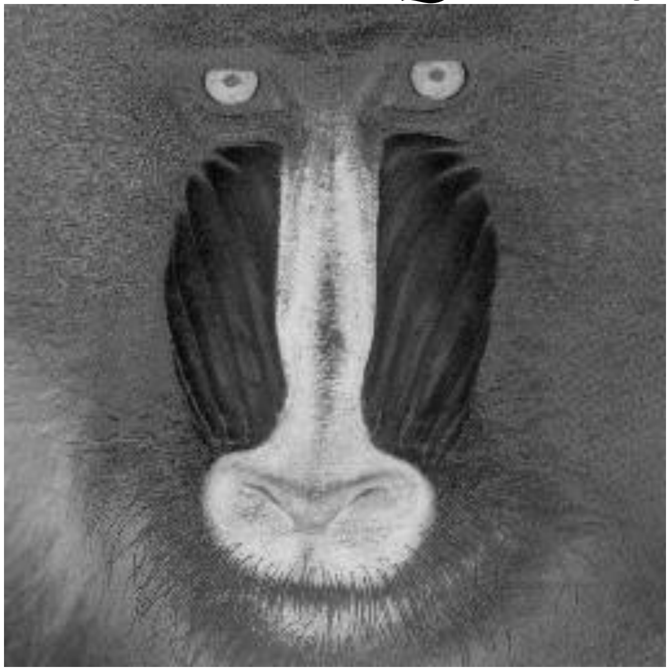
# YIQ Color Model

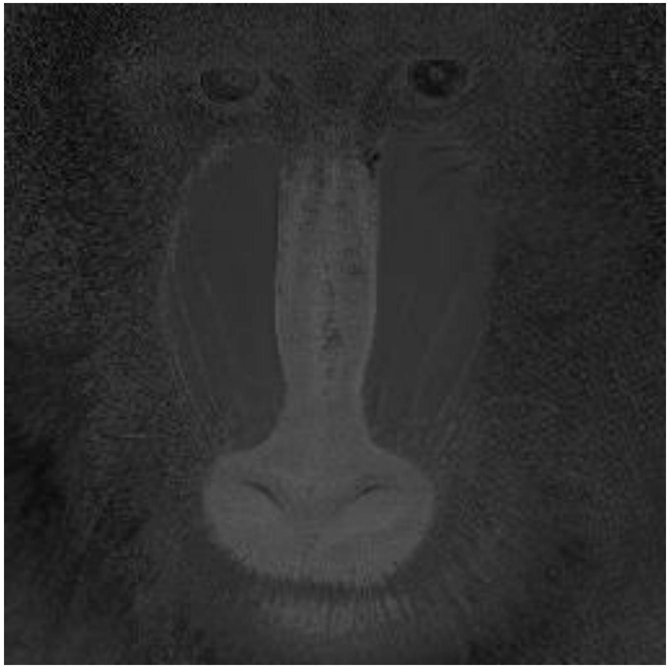
- YIQ is used in NTSC color TV broadcasting. Again, gray pixels generate zero ($I$, $Q$) chrominance signal.
- $I$ and $Q$ are a rotated version of $U$ and $V$ .
- $Y'$ in YIQ is the same as in YUV. $U$ & $V$ are rotated by 33°.
- Idea → eye is more sensitive to changes in the orange-blue (I) range than in the purple-green range (Q)
  - therefore less bandwidth is required for Q than for I

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.595716 & -0.274453 & -0.321263 \\ 0.211456 & -0.522591 & 0.311135 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0.9563 & 0.6210 \\ 1 & -0.2721 & -0.6474 \\ 1 & -1.1070 & 1.7046 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}$$

# I & Q Components

$I$ and $Q$ components of color image.





# YCbCr

- In practice, however, Recommendation 601 specifies 8-bit coding, with a maximum $Y'$ value of only 219, and a minimum of +16.
- Cb and Cr have a range of ±112 and offset of +128. If $R'$, $G'$, $B'$ are floats in [0.. + 1], then we obtain $Y'$, Cb, Cr in [0..255] via the transform:

$$R = Y + 1,402 \cdot (Cr - 128)$$
$$G = Y - 0,34414 \cdot (Cb - 128) - 0,71414 \cdot (Cr - 128)$$
$$B = Y + 1,772 \cdot (Cb - 128)$$

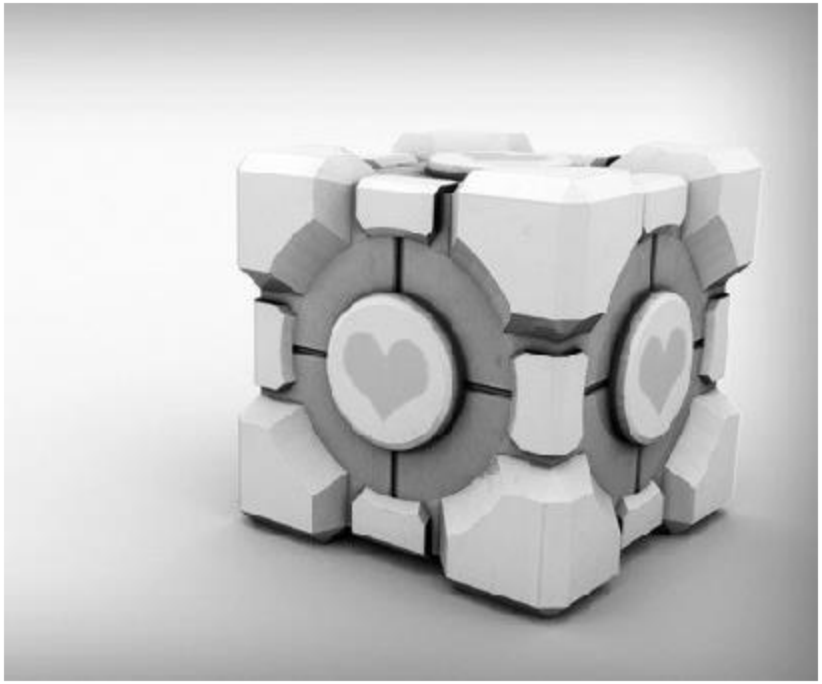
$$\begin{bmatrix} Y & Cb & Cr \end{bmatrix} = \begin{bmatrix} R & G & B \end{bmatrix} \begin{bmatrix} 0.299 & -0.168935 & 0.499813 \\ 0.587 & -0.331665 & -0.418531 \\ 0.114 & 0.50059 & -0.081282 \end{bmatrix}$$

- The YCbCr transform is used in JPEG image compression and MPEG video compression.

# CONVERSION BETWEEN IMAGE TYPES

# Gray Scale to Binary conversion

- Naïve conversion
  - [0, 127] replaced by 0
  - [128, 255] replaced by 1



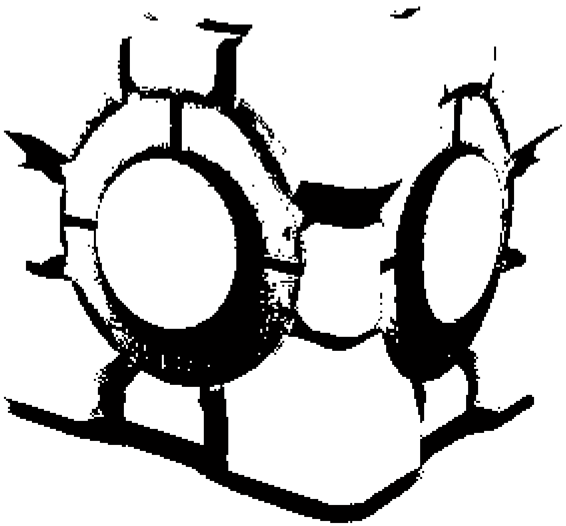
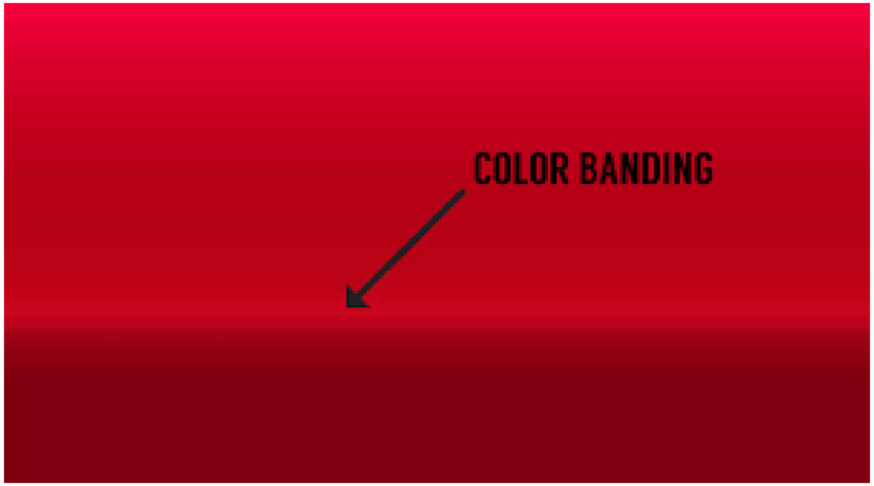


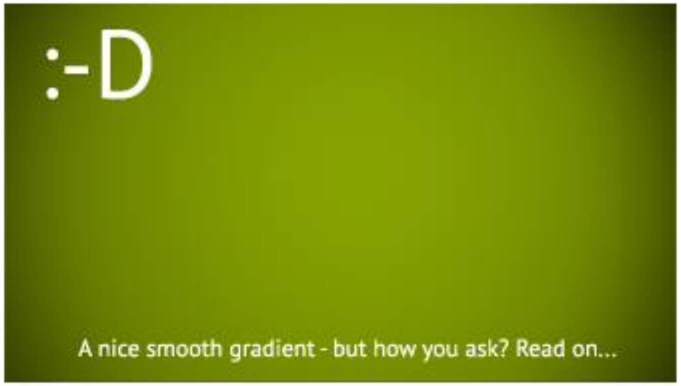
**Image is barely recognizable**

# Dithering Process

- Process used to randomize quantization error and prevents large-scale patterns such as color banding.





# Dithering Process

- Several algorithms for dithering
  - Thresholding or average dithering
  - Random dithering
  - Patterning using a fixed pattern
  - Ordered dithering
    - Halftone dithering
    - Bayer matrix
    - Void-and-cluster method
  - Error-diffusion dithering
    - Floyd-Steinberg
    - Jarvis, Judice, and Ninke dithering
    - Stucki dithering
    - .....

# Dithering Process

- Average dithering → use the average value of the image pixels as global threshold for quantization
  - ✓ Immense loss of details and contouring

# Dithering Process

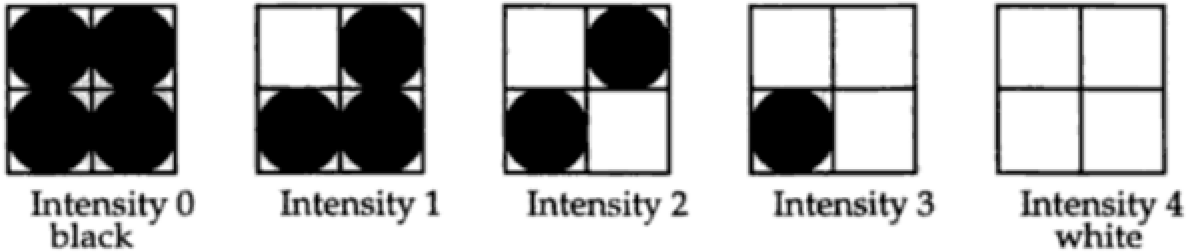
- Median-cut algorithm can help to find the threshold using histogram
  - ✓ Here choose the median value as a threshold

# Dithering Process

- Random dithering → first attempt to remedy thresholding technique
  - Each pixel is compared against a random threshold.
  - No patterned artifacts but noise tends to swamp the image details

# Dithering Process

- Ordered dithering with fixed pattern
  - Idea is to use area of dots proportional to intensity in image.
- Classical method is halftoning
- Example: If we use 2 x 2 pattern → 5 different intensities
- First convert grayscale values to [0, 4] then replace each value with one of below 2x2 (0 = left, 3 = right)



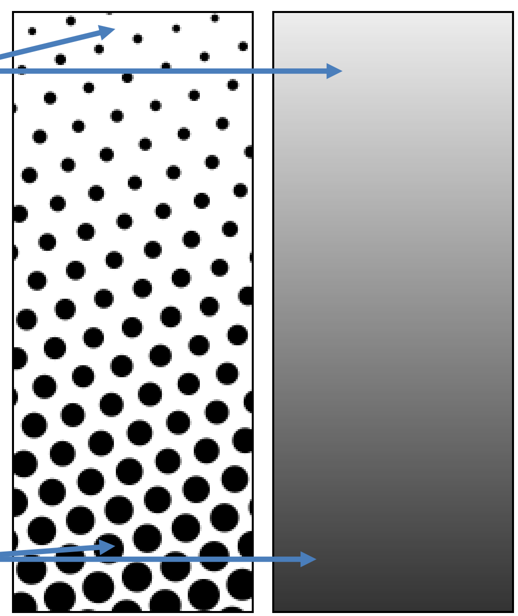

Dithering with 2x2 pattern

# Dithering Process

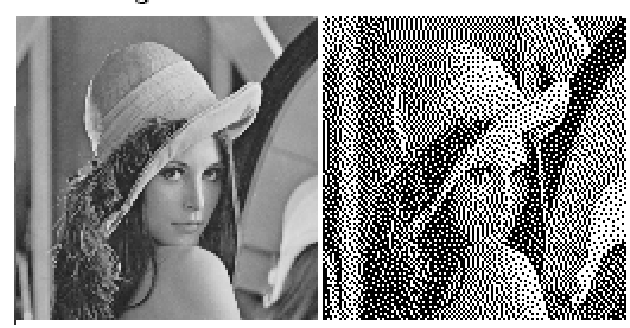
- The general idea is:
  - Replace each pixel value with a n x n pixel ( if dithering matrix is of size n x n)
- Problem → Size of image is bigger
  - If dithering matrix is 2x2 →4 times bigger
  - If dithering matrix is 4x4 → 16 times bigger

Lighter = smaller dots

Darker = bigger dots



Original Halftoned



# Dithering Process

- Ordered dithering → express the patterns in compact form as the order of dots added

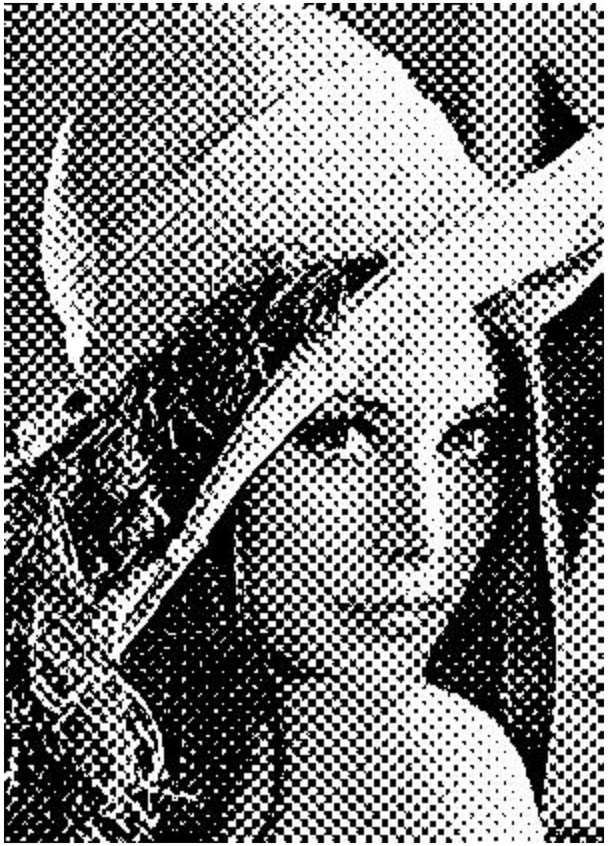
| | | | | | | |
|---|---|---|---|---|---|---|
| 8 | 3 | 4 | and | 1 | 7 | 4 |
| 6 | 1 | 2 | | 5 | 8 | 3 |
| 7 | 5 | 9 | | 6 | 2 | 9 |

- If pixel value (scaled to [0, 9] if 3x3) less than number in pattern at same position → replace it with 0, 1 otherwise
- Bayer has shown that matrices of size powers of two are optimal

| | |
|---|---|
| 1 | 3 |
| 4 | 2 |

| | | | |
|---|---|---|---|
| 1 | 9 | 3 | 11 |
| 13 | 5 | 15 | 7 |
| 4 | 12 | 2 | 10 |
| 16 | 8 | 14 | 6 |

# Dithering Process



# Dithering Process

- Several matrices in ordered dithering
  - Below 4x4, 8x8 Bayer matrices as examples

| | | | |
|---|---|---|---|
| 0 | 12 | 3 | 15 |
| 8 | 4 | 11 | 7 |
| 2 | 14 | 1 | 13 |
| 10 | 6 | 9 | 5 |

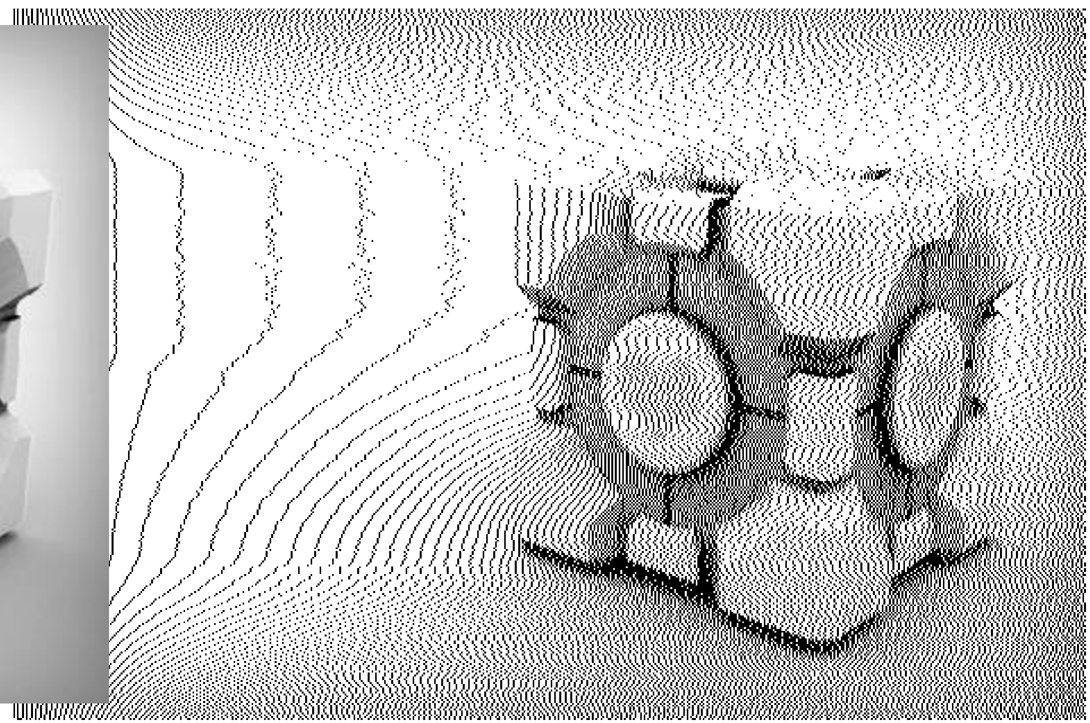
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 48 | 12 | 60 | 3 | 51 | 15 | 63 |
| 32 | 16 | 44 | 28 | 35 | 19 | 47 | 31 |
| 8 | 56 | 4 | 52 | 11 | 59 | 7 | 55 |
| 40 | 24 | 36 | 20 | 43 | 27 | 39 | 23 |
| 2 | 50 | 14 | 62 | 1 | 49 | 13 | 61 |
| 34 | 18 | 46 | 30 | 33 | 17 | 45 | 29 |
| 10 | 58 | 6 | 54 | 9 | 57 | 5 | 53 |
| 42 | 26 | 38 | 22 | 41 | 25 | 37 | 21 |

# Dithering Process

- Error-diffusion algorithms
  - Idea is to propagate errors of conversion to other pixels
  - Ex: 96 is closer to 0 so converted to black.
  - If several neighbor pixels of values less than 127 → huge empty black pixels
- When pixel converted to a value → error is added to next pixel value (reset at row end)
  - here we add 96 to the next pixel value

# Dithering Process

- Error-diffusion algorithms → simple version
  - ✓ lines of dots because of pushing errors only to the next pixel value



# Dithering Process

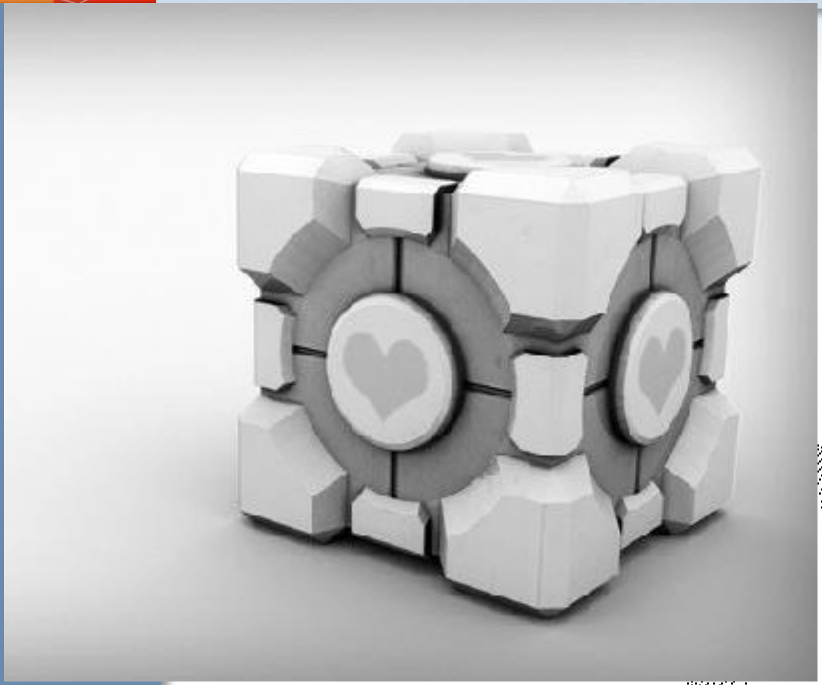
- Solution → distribute errors more intelligently
  - ✓ Horizontally and vertically
- Floyd-Steinberg dithering algorithm still the most popular algorithm that diffuses errors on neighboring pixels

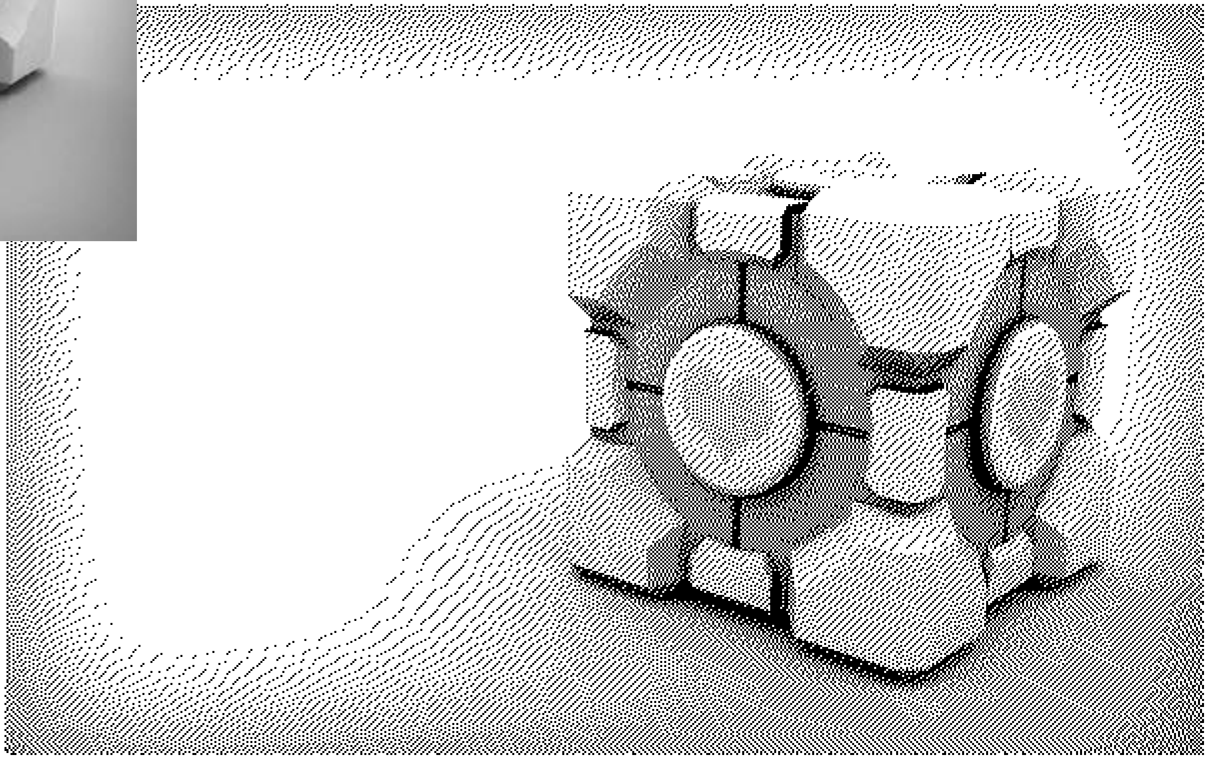
| | | |
|---|---|---|
| | Pixel value | 7/16 |
| 3/16 | 5/16 | 1/16 |

- 96 to be propagated is multiplied by pixels entries of above pattern and added to pixel neighbors

| | | |
|---|---|---|
| | 0 | 42 |
| 18 | 30 | 6 |

# Dithering Process

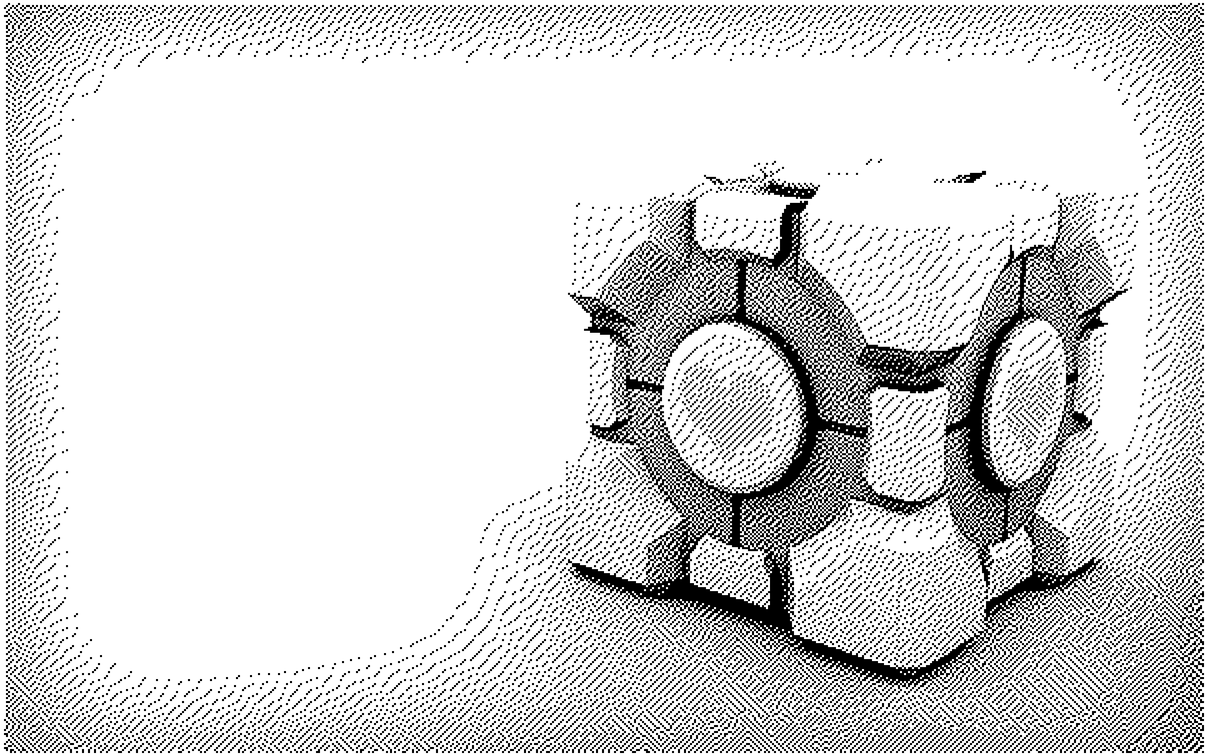




# Dithering Process

➢ Jarvice, Judice, and Ninke dithering → use more complex filter

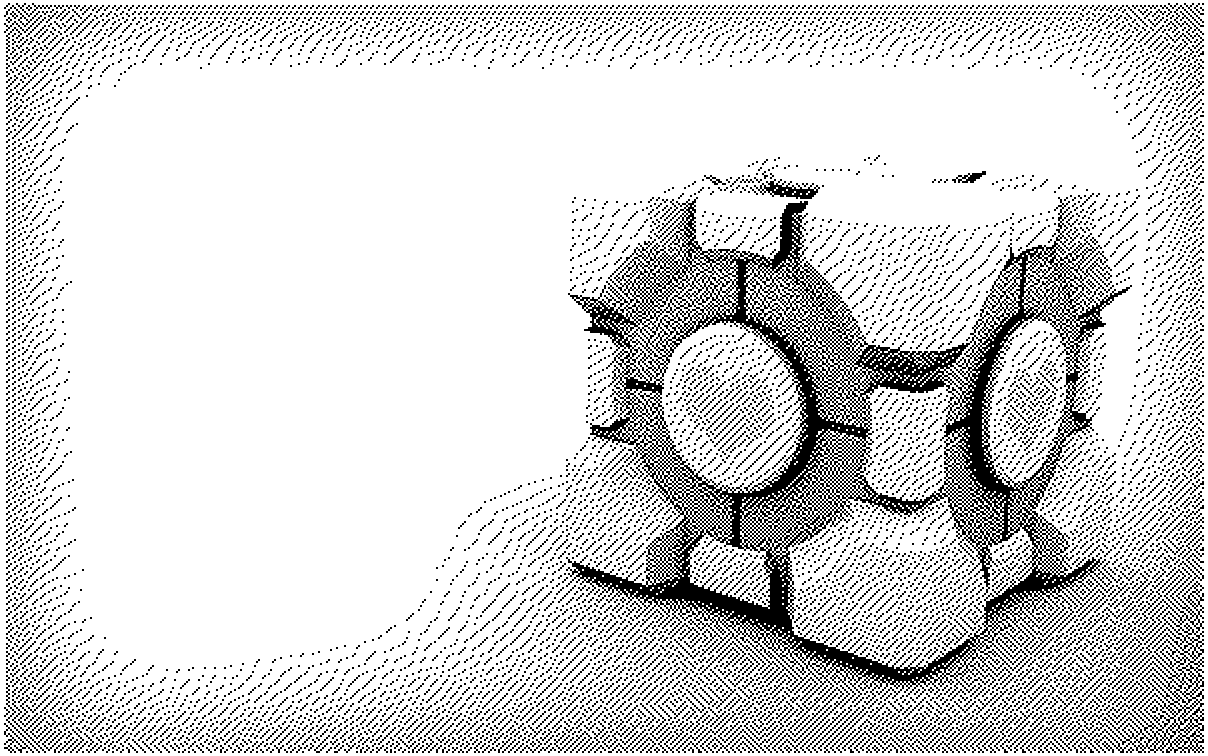
| | | | | |
|---|---|---|---|---|
| | | X | 7/48 | 5/48 |
| 3/48 | 5/48 | 7/48 | 5/48 | 3/48 |
| 1/48 | 3/48 | 5/48 | 3/48 | 1/48 |



# Dithering Process

- Stucki dithering

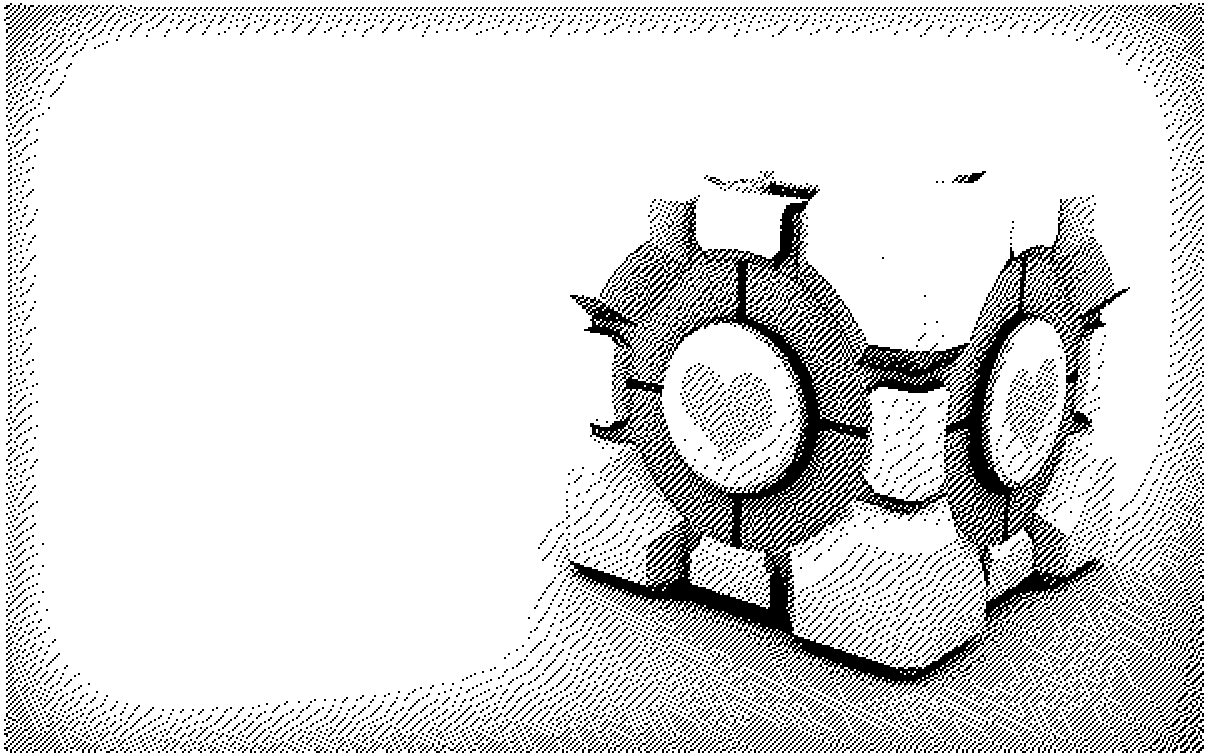
| | | | | |
|---|---|---|---|---|
| | | X | 8/42 | 4/42 |
| 2/42 | 4/42 | 8/42 | 4/42 | 2/42 |
| 1/42 | 2/42 | 4/42 | 2/42 | 1/42 |



# Dithering Process

- Atkinson dithering

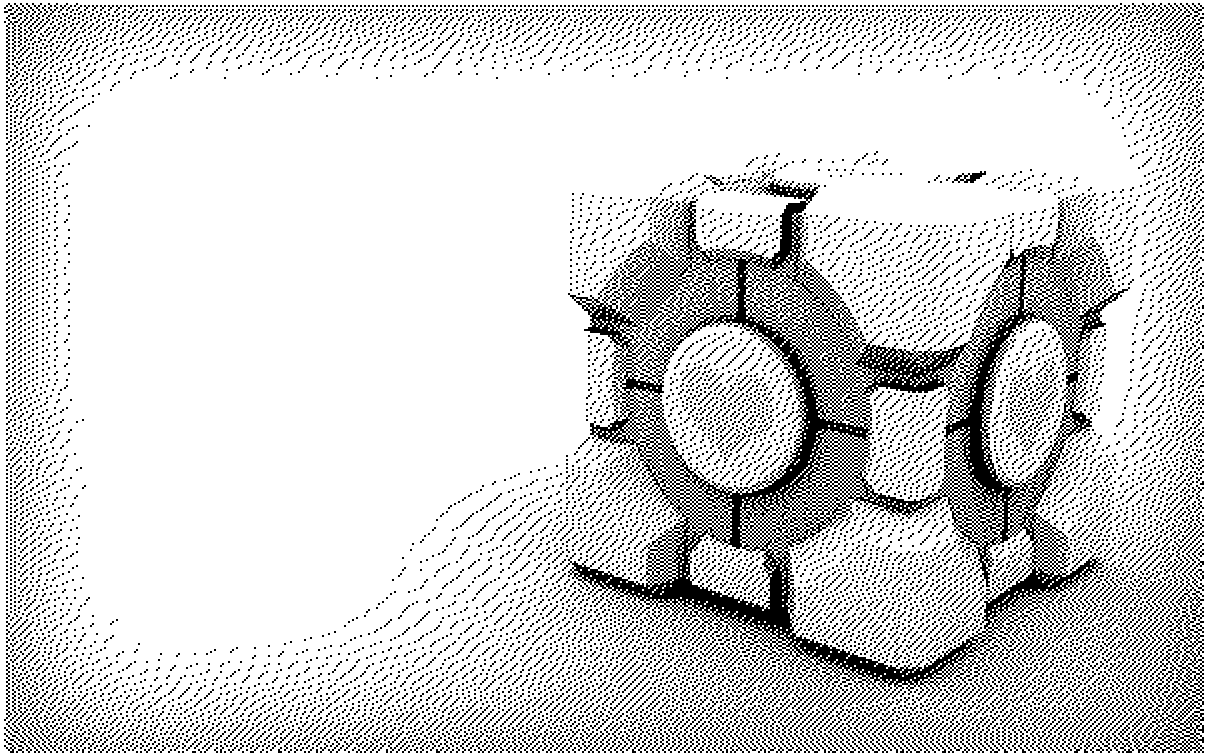
| | | | |
|---|---|---|---|
| | X | 1/8 | 1/8 |
| 1/8 | 1/8 | 1/8 | |
| | 1/8 | | |



# Dithering Process

- Burkes dithering

| | | | | |
|---|---|---|---|---|
| | | X | 8/32 | 4/32 |
| 2/32 | 4/32 | 8/32 | 4/32 | 2/32 |

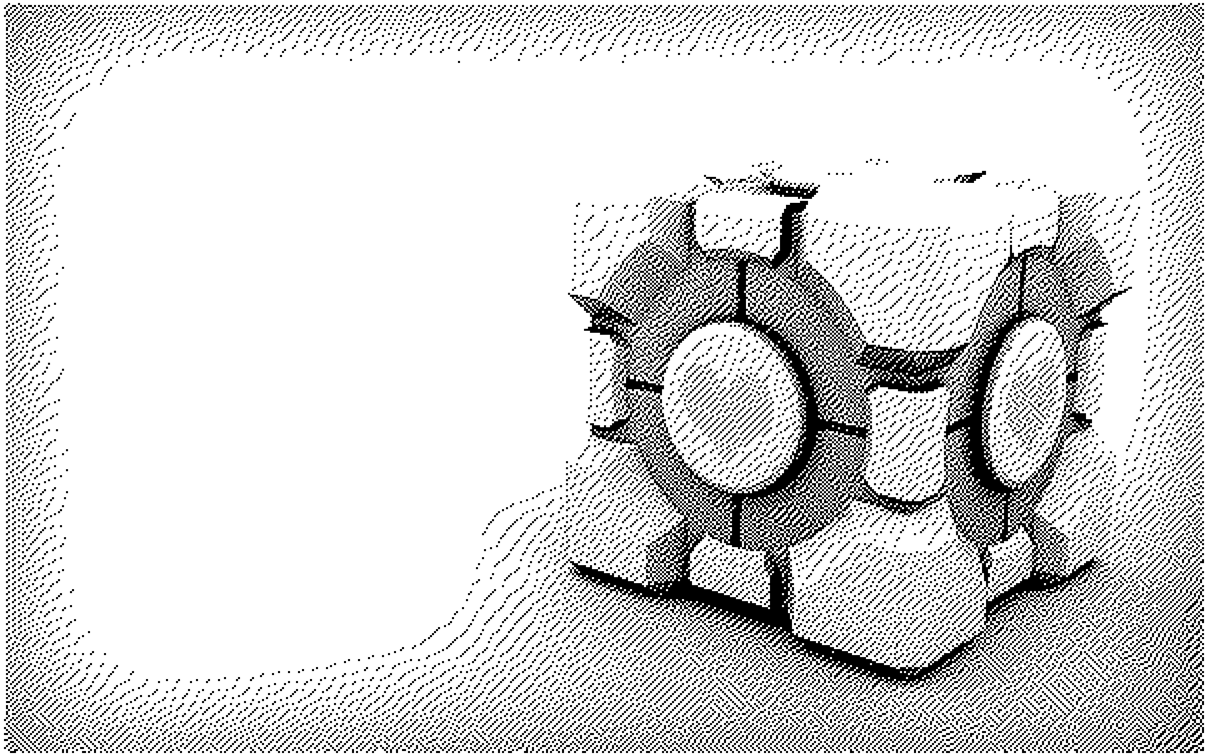


# Dithering Process

- Sierra dithering → Published three matrices

Sierra

| | | | | |
|---|---|---|---|---|
| | | X | 5/32 | 3/32 |
| 2/32 | 4/32 | 5/32 | 4/32 | 2/32 |
| | 2/32 | 3/32 | 2/32 | |

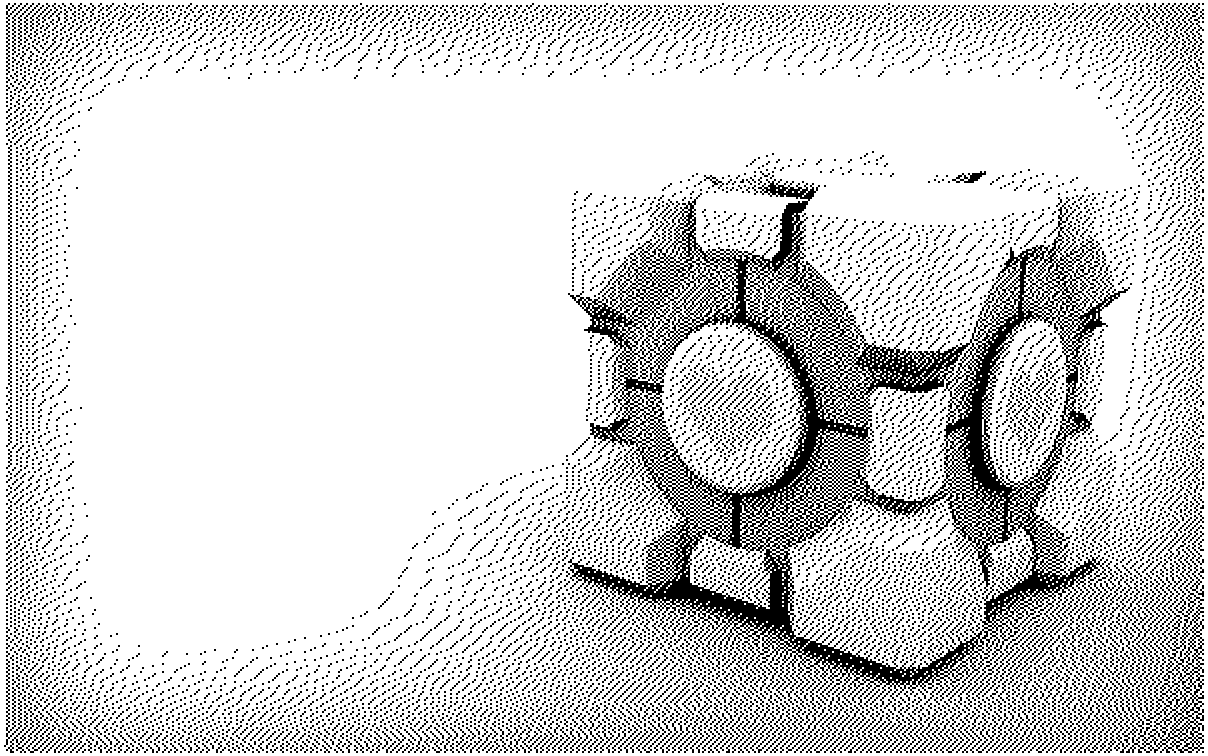


# Dithering Process

- Sierra dithering → Published three matrices

two-row Sierra

| | | | | |
|---|---|---|---|---|
| | | X | 4/16 | 3/16 |
| 1/16 | 2/16 | 3/16 | 2/16 | 1/16 |

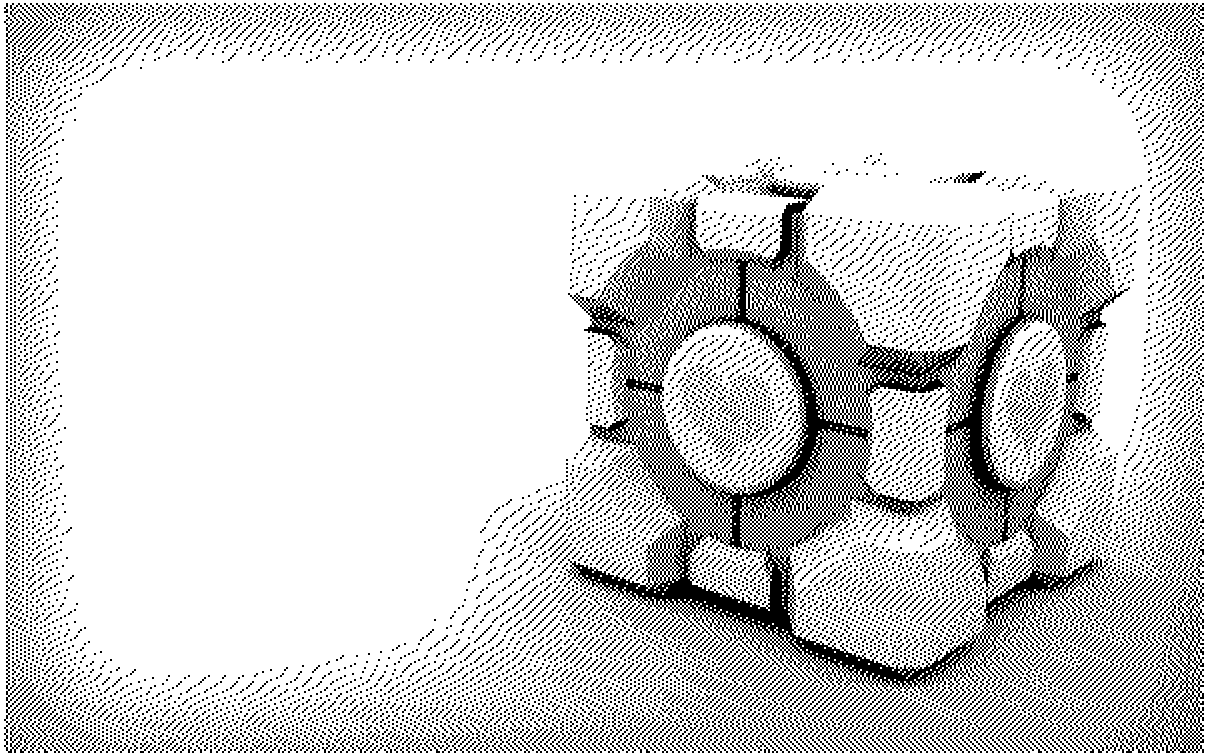


# Dithering Process

- Sierra dithering → Published three matrices

Sierra lite

| | | |
|---|---|---|
| | X | 2/4 |
| 1/4 | 1/4 | |



# Error-diffusion Dithering

# Colored Image to Index

- ➢ Using a colormap → map each color in the image to the nearest one in the colormap
- ➢ Create a colormap as the top 256 most used colors in the image and the others mapped to the nearest one.