Digital Imaging Basics

Computer Vision

Aziz M. Qaroush

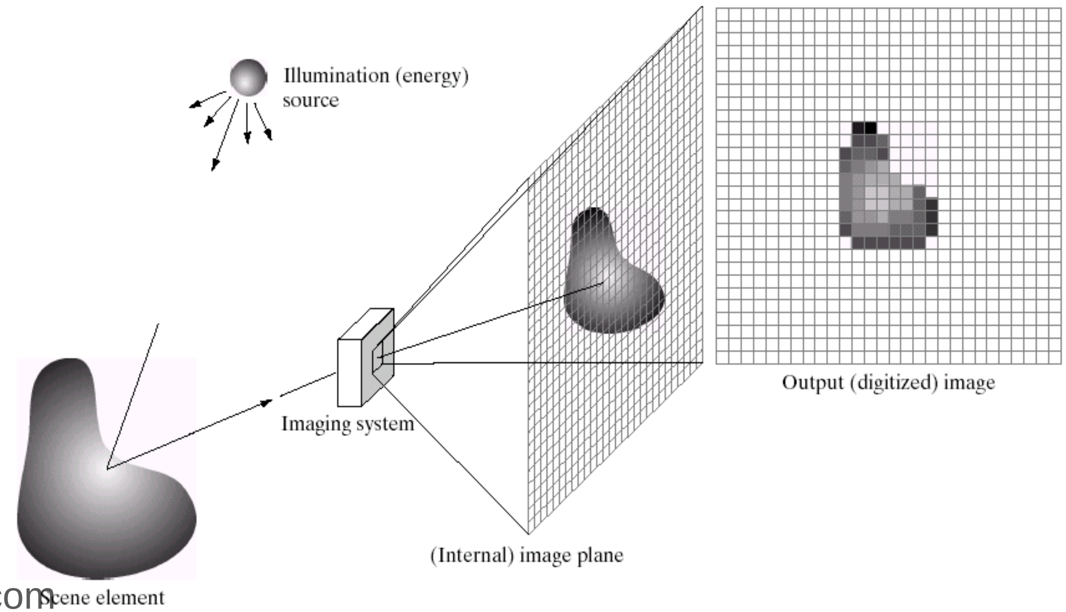
Birzeit University

Outline

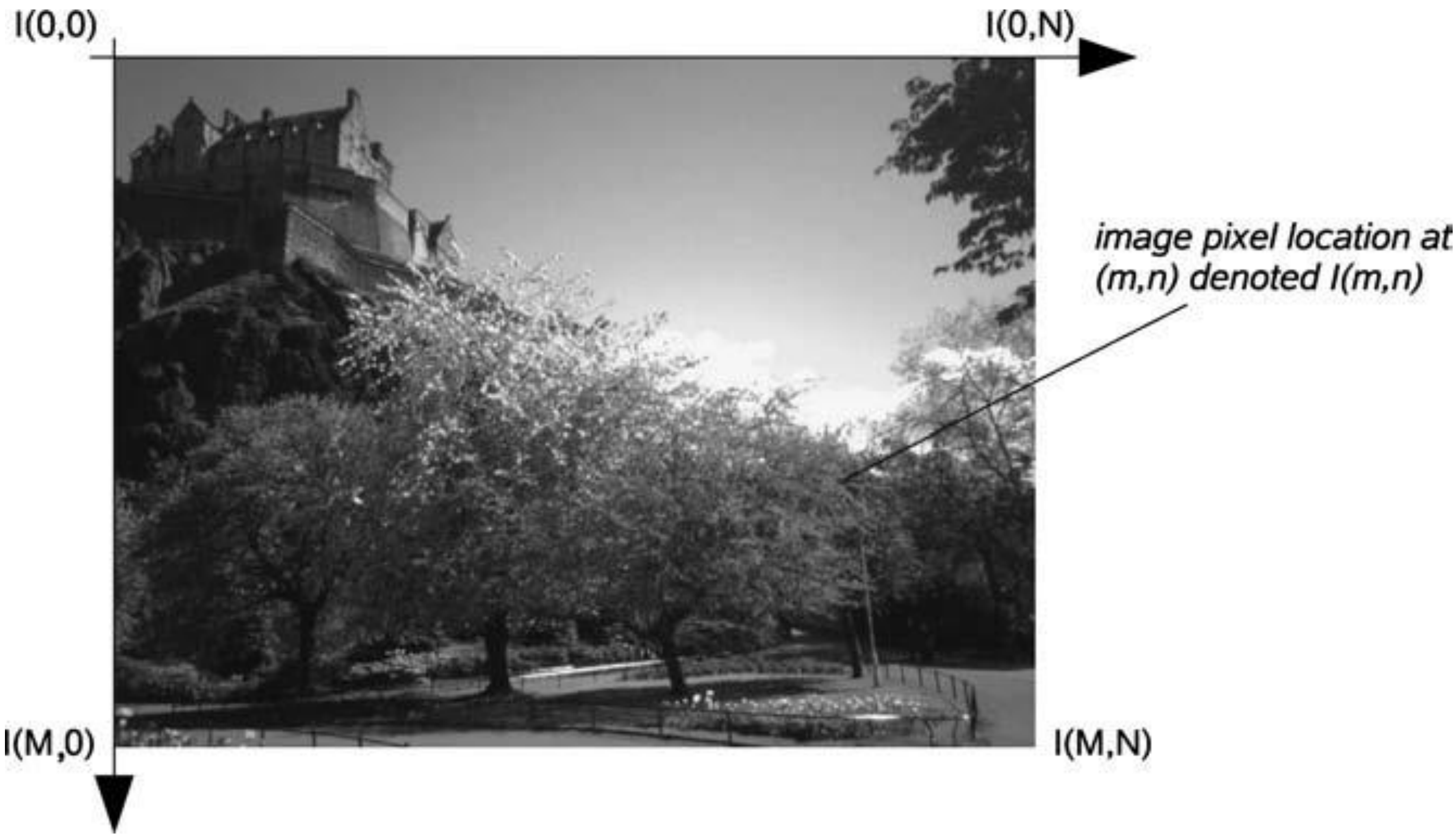
- What is a digital image?
- Image Sensing and Acquisition
- Image Sampling and Quantization
- Image Representation
- Color Models
- Thresholding
- Image Resizing

What is a Digital Image?

- A **digital image** is a representation of a two- dimensional image as a finite set of digital values, called picture elements or pixels.
  - An **image can be define as a two-dimensional function f(x,y) with x and y being the spatial coordinates and f is the amplitude**
  - A **digital image is the representation of an image using finite and discrete values for x,y, and f**
  - These values are called picture elements or pixels

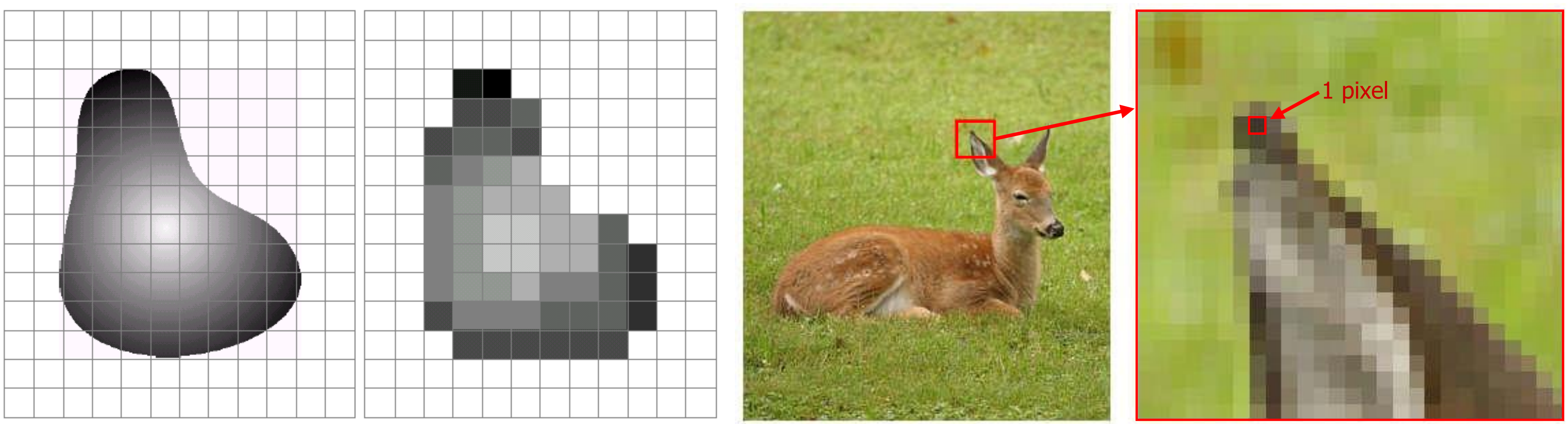


What is a Digital Image?



What is a Digital Image? (cont…)

- Pixel values typically represent gray levels, colors, heights, opacities etc
- **Remember** *digitization* implies that a digital image is an *approximation* of a real scene



What is a Digital Image? (cont…)

- Common image formats include:
  - 1 sample per point (Grayscale)
  - 3 samples per point (Red, Green, and Blue)
  - 4 samples per point (Red, Green, Blue, and Alpha (transparency channel as in the case of PNG format images)).

- For most of this course we will focus on grey-scale images

Image formats

Acronym	Name	Properties
GIF	Graphics interchange format	Limited to only 256 colours (8 bit); lossless compression
JPEG	Joint Photographic Experts Group	In most common use today; lossy compression; lossless variants exist
BMP	Bit map picture	Basic image format; limited (generally) lossless compression; lossy variants exist
PNG	Portable network graphics	New lossless compression format; designed to replace GIF
TIF/TIFF	Tagged image (file) format	Highly flexible, detailed and adaptable format; compressed/uncompressed variants exist

Outline

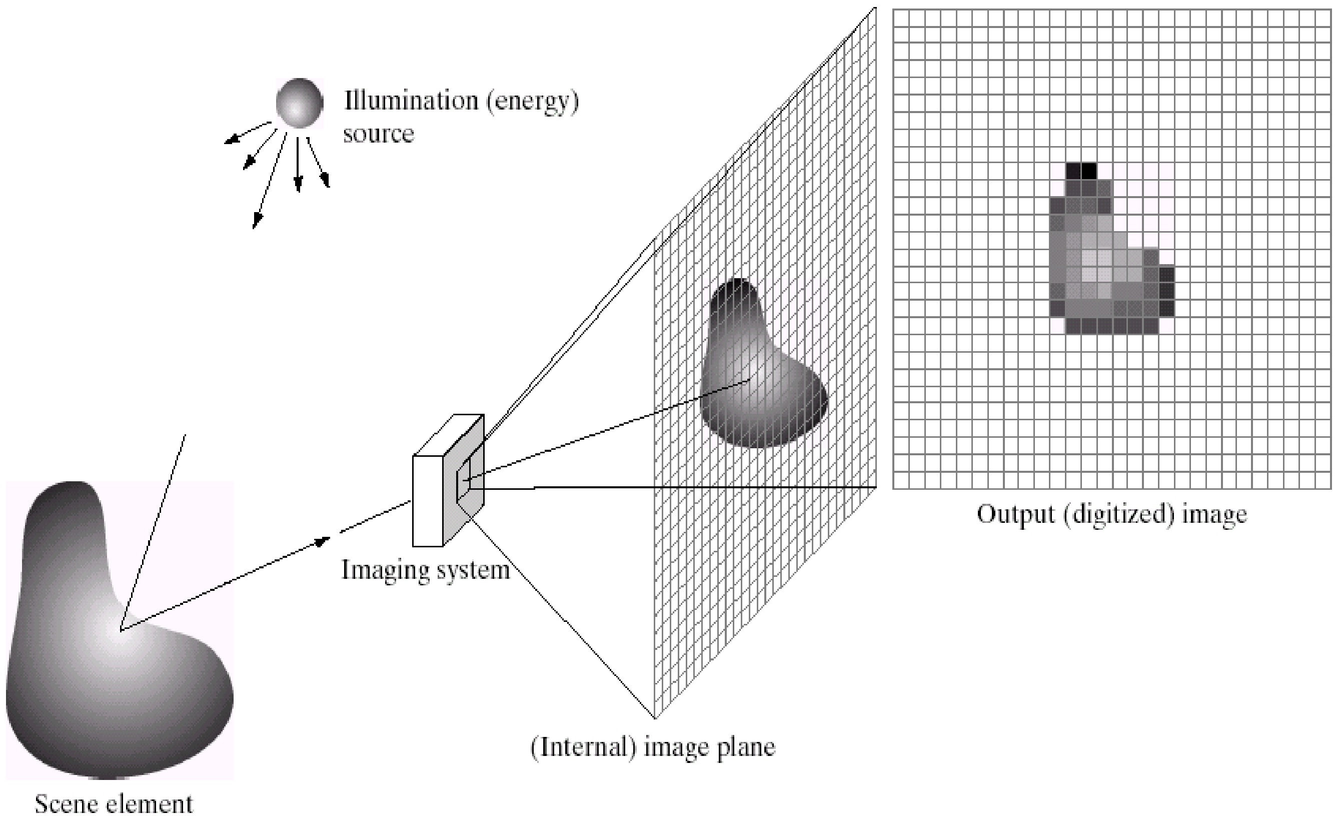
- What is a digital image?
- **Image Sensing and Acquisition**
- Image Sampling and Quantization
- Image Representation
- Color Models
- Thresholding
- Image Resizing

Image Sensing and Acquisition

- Sensing and acquisition depend on the problem domain
- Generally, we need
  - Source for illumination(gamma, x-ray, ultrasound …..)
  - Device(s) to collect the energy reflected-from/transmitted-through the objects in the scene (stars, patients, natural scenes)
- Radiance is the total energy that flows from a light source
- Luminance is the level of energy an observer perceives from a light source
- Fundamental limit:
  - To see an object the electromagnetic wavelength must be no bigger than the object

Image Sensing and Acquisition

- Acquisition with Sensor Arrays
  - Predominant arrangement found in many applications
  - Most expensive and usually no motion is required



Simple Image Formation Model

- An image can represented as a two-dimensional function **f(x,y)**
- The value of f at spatial coordinates (x, y) is a scalar quantity whose physical meaning is determined by the source of the image, and whose values are proportional to energy radiated by a physical source (e.g., electromagnetic waves).
- f (x, y) is characterized by two components:
  - The amount of source illumination incident on the scene being viewed **i(x,y)**, and
  - The amount of illumination reflected by the objects in the scene **r(x,y)**
- An image is proportional to the radiated energy

$$f(x,y) = i(x,y)r(x,y)$$

  - Illumination bound: $0 < i(x,y) < \infty$
  - Reflectivity bound: $0 < r(x,y) < 1$

A Simple Image Formation Model

- For monochrome images we deal with intensity or gray levels (L)

$$L = f(x,y)$$

$$L_{min} \le L \le L_{max}$$

- The only requirement on Lmin is to be positive and on Lmax to be finite
- Typical indoor values for Lmin and Lmax are 10 lm/m2 and 1000 lm/m2
- The interval [Lmin,Lmax] is called the gray scale and is usually shifted to [0,L-1] where 0 is considered black and L-1 is considered white

Outline

- What is a digital image?
- Image Sensing and Acquisition
- **Image Sampling and Quantization**
- Image Representation
- Color Models
- Thresholding
- Image Resizing

Image Sampling and Quantization

- Sensors used in the acquisition produce continuous voltage signal
- In order to produce a digital form of the image, it has to go through two processes:
  - **Sampling**: digitize the spatial coordinates, x and y
  - **Quantization**: digitize the amplitude f(x,y)

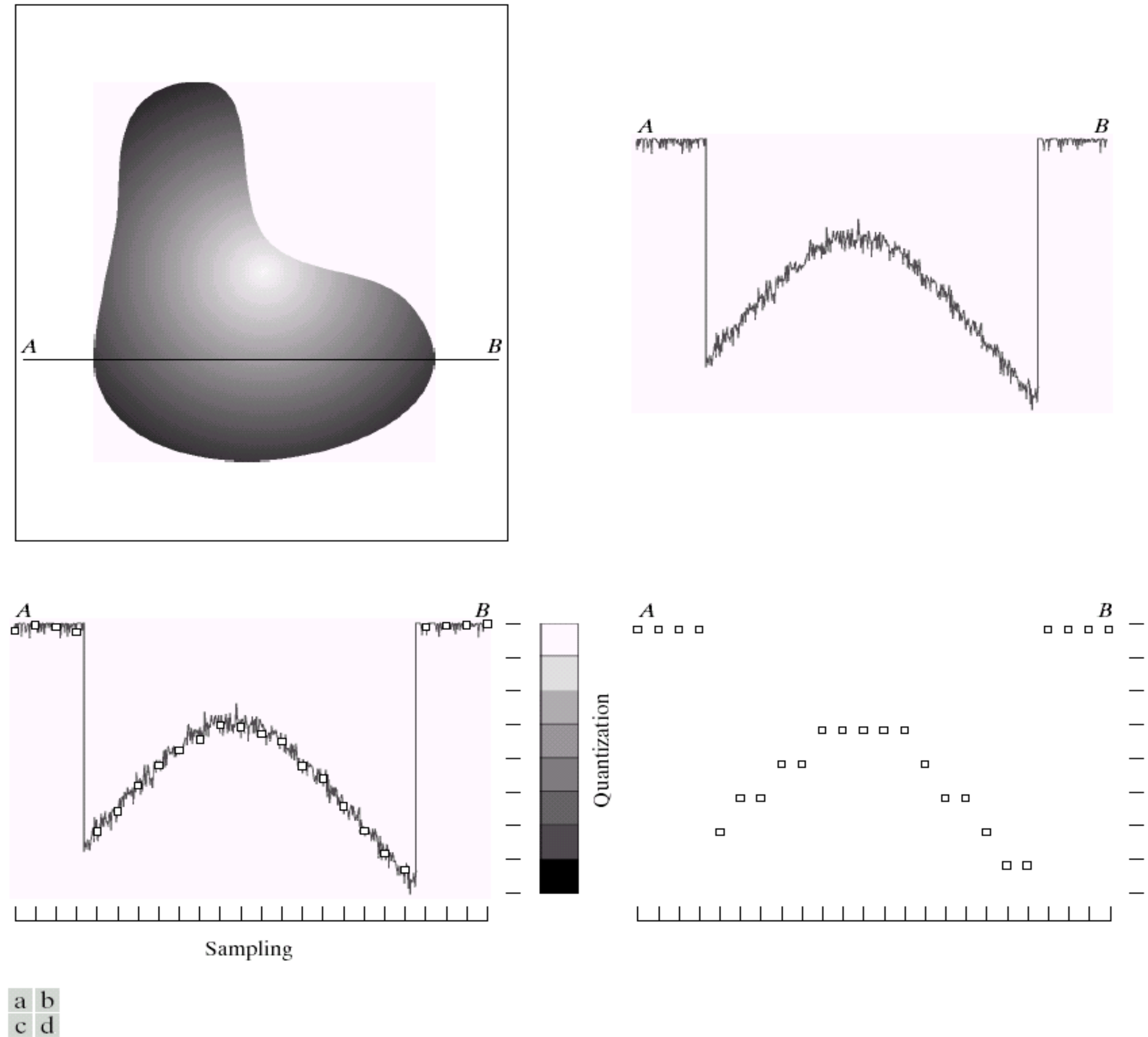
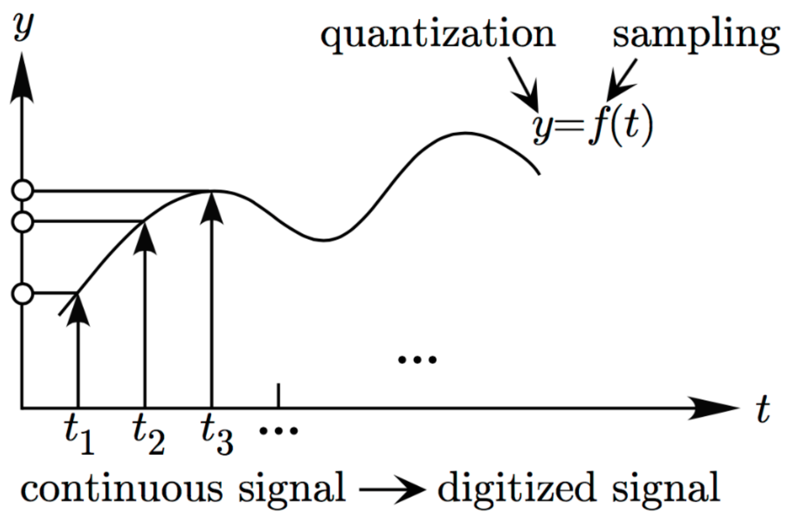


FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Image Sampling and Quantization…

one-dimensional



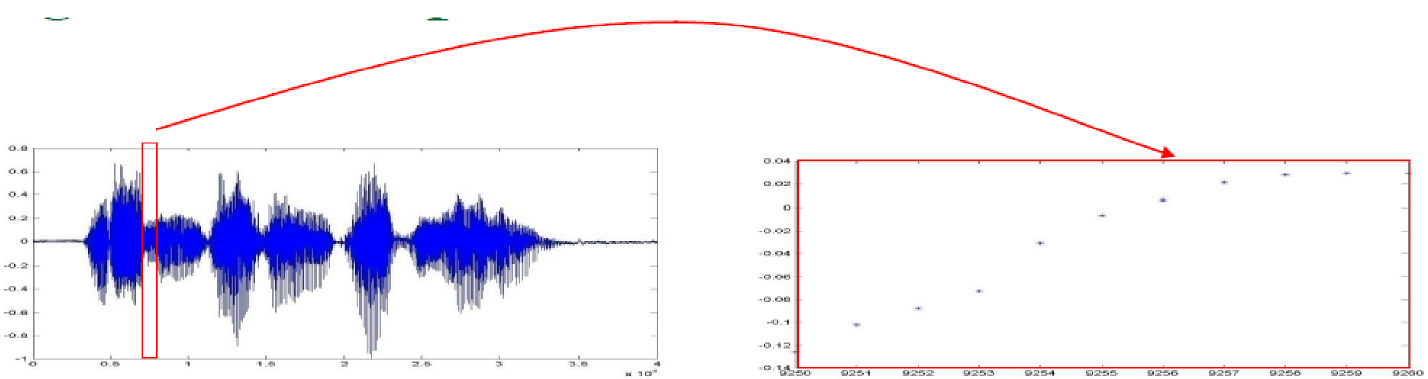


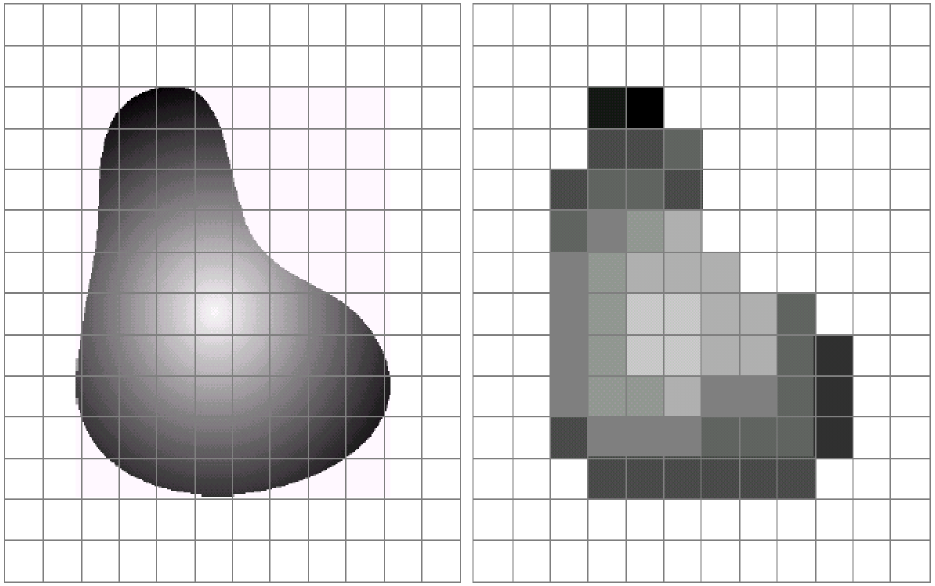
Image Sampling and Quantization

- Resolution (how much you can see the detail of the image) depends on sampling and gray levels.
- The bigger the sampling rate (n) and the gray scale (g), the better the approximation of the digitized image from the original.
- The more the quantization scale becomes, the bigger the size of the digitized image.
- The number of gray levels typically is an integer power of 2

$$L = 2^k$$

- Number of bits required to store a digitized image

$$b = M \times N \times k$$



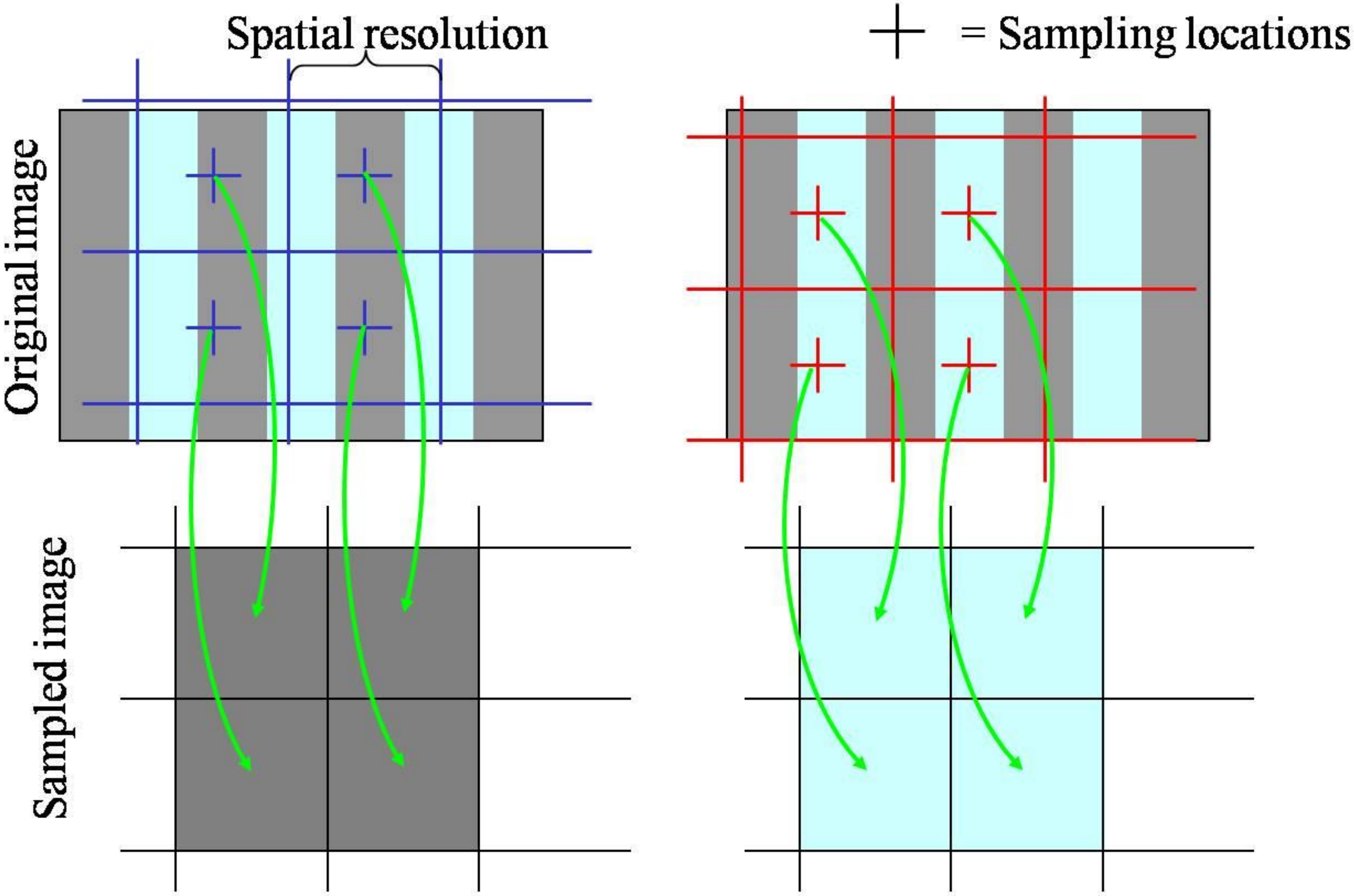
a b

FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

Spatial Resolution

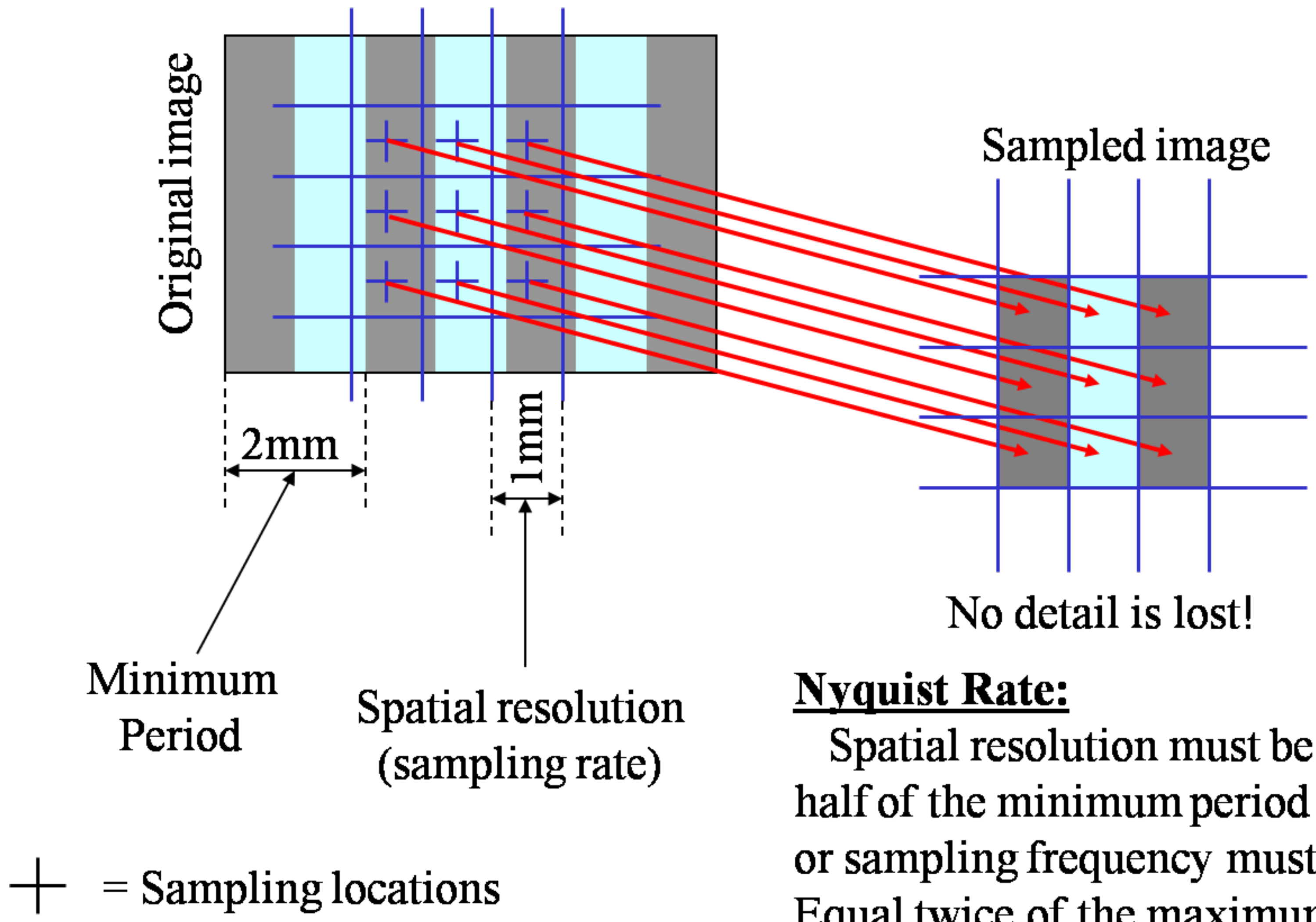
- Spatial resolution is a sampling parameter.
- The spatial resolution of an image is determined by how sampling was carried out
- Spatial resolution simply refers to the smallest discernable detail in an image
  - Vision specialists will often talk about pixel size or spatial pixel density
  - Graphic designers will talk about dots per inch (DPI)
- Most current desktop monitors display around 96 to 110 DPI, with laptops coming in slightly higher

Spatial Resolution



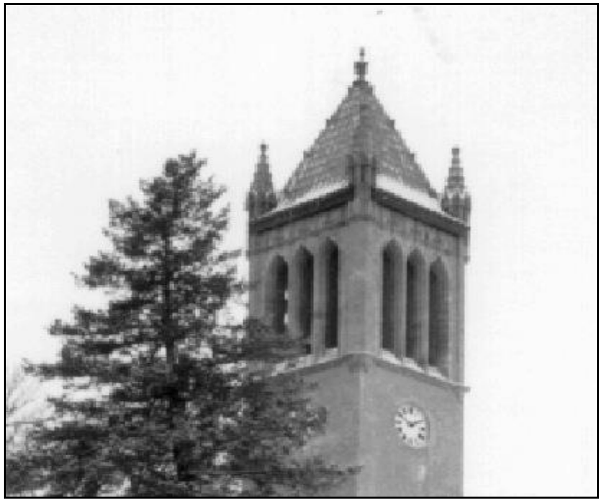
Under sampling, we lost some image details!

Spatial Resolution



Nyquist Rate:
Spatial resolution must be less or equal half of the minimum period of the image or sampling frequency must be greater or Equal twice of the maximum frequency.

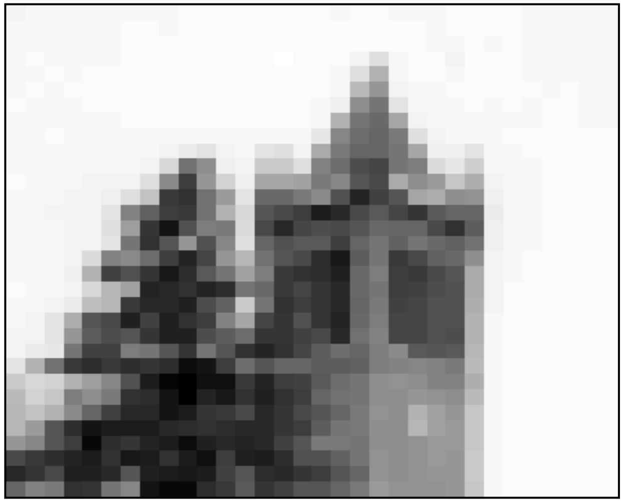
Spatial Resolution



256x256 pixels

128x128 pixels

64x64 pixels



32x32 pixels

Intensity Level Resolution

- ***Intensity level resolution***: number of intensity levels used to represent the image
  - The more intensity levels used, the finer the level of detail discernable in an image
  - Intensity level resolution usually given in terms of number of bits used to store each intensity level

Number of Bits	Number of Intensity Levels	Examples
1	2	0, 1
2	4	00, 01, 10, 11
4	16	0000, 0101, 1111
8	256	00110011, 01010101
16	65,536	1010101010101010

False Contouring

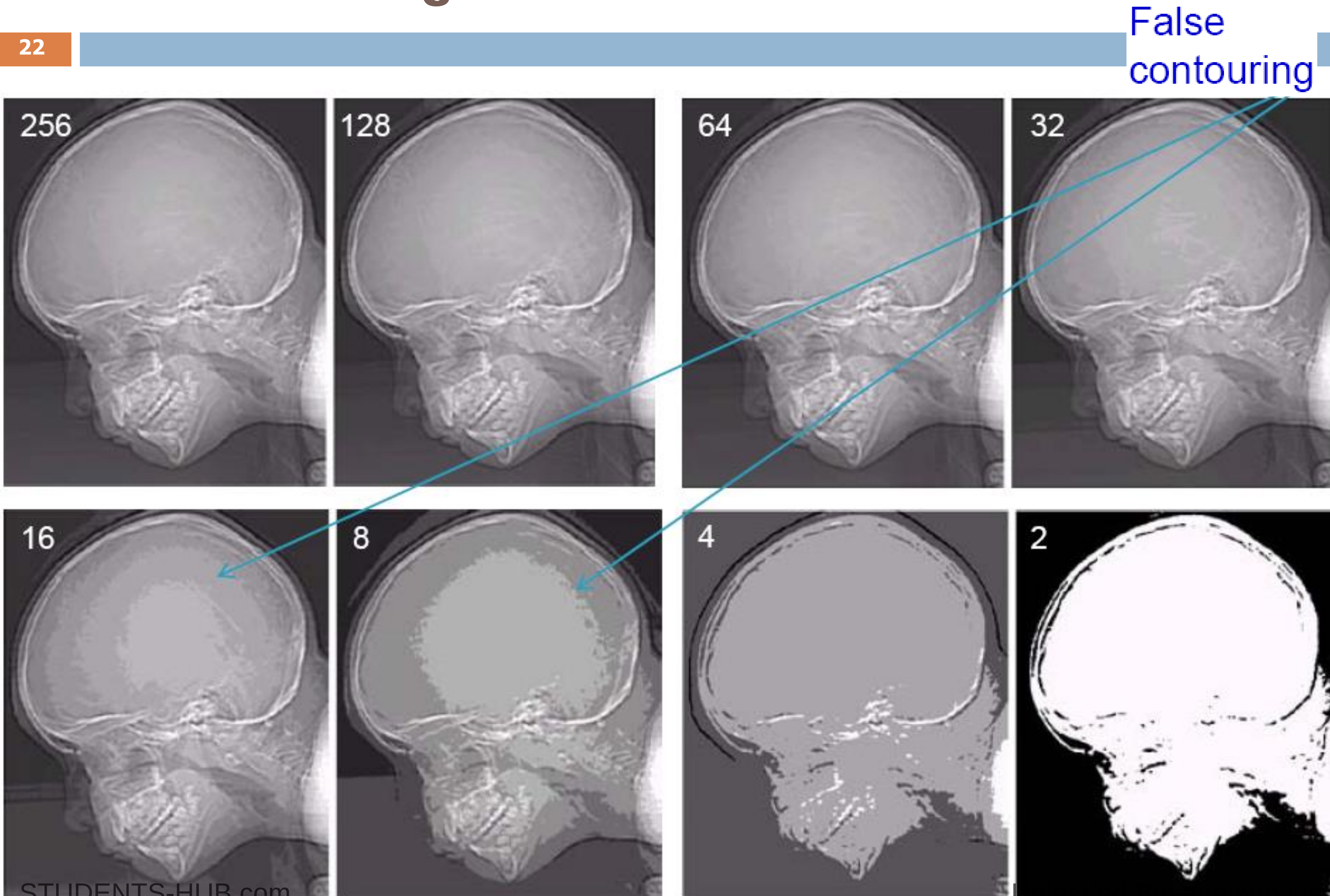


Image details and resolution

To satisfy human mind

1. For images of the same size, the low detail image may need more pixel depth.
2. As an image size increase, fewer gray levels may be needed.

Resolution: How Much Is Enough?

- The big question with resolution is always *how much is enough*?
  - Depends on what is in the image *(details)* and what you would like to do with it *(applications)*
  - Key questions:
    - Does image look aesthetically pleasing?
    - Can you see what you need to see in image?

Resolution: How Much Is Enough?

- **Example:** Picture on right okay for counting number of cars, but not for reading the number plate

Outline

- What is a digital image?
- Image Sensing and Acquisition
- Image Sampling and Quantization
- **Image Representation**
- Color Models
- Thresholding
- Image Resizing

Digital Image Representation

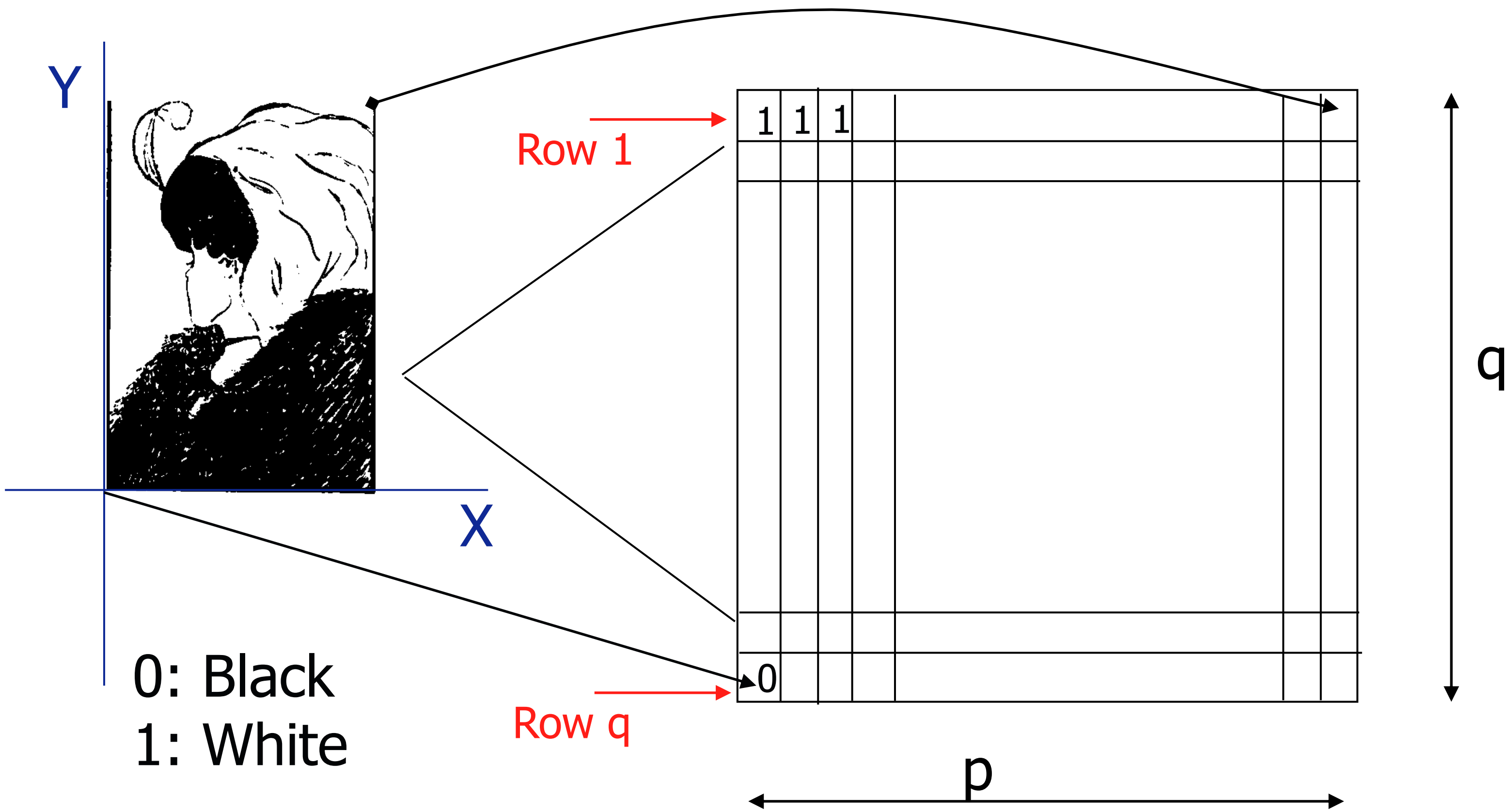
- Once the image is digitized, it can be represented as an array with M rows and N columns with each element called a pixel
- The values stored in the array elements represent the image values at that location

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & & \vdots \\ a_{M-1,0} & a_{M-1,1} & \cdots & a_{M-1,N-1} \end{bmatrix}$$

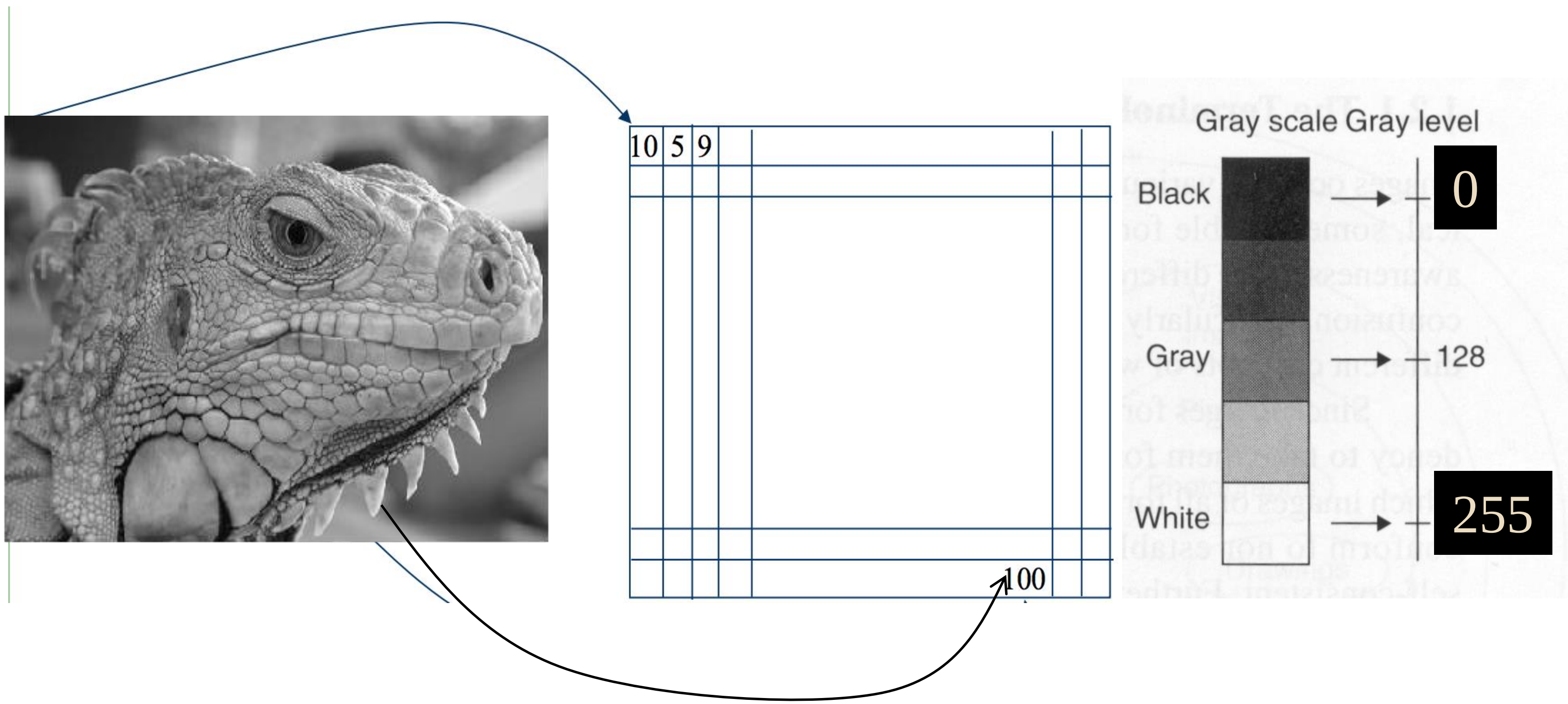
Binary Image

- A *binary image* is a digital image that has only two values (0 or 1) for each pixel, 0 = black and 1 = white



Gray scale image

- The intensity of each pixel is represented by a value between 0 and 255 (8 bits/pixel)



Vector-Valued Images

- A *vector-valued image* has more than one *channel* or *band*

  Image values $[u_1,\ldots,u_{Nchannels}]^T$ are vectors

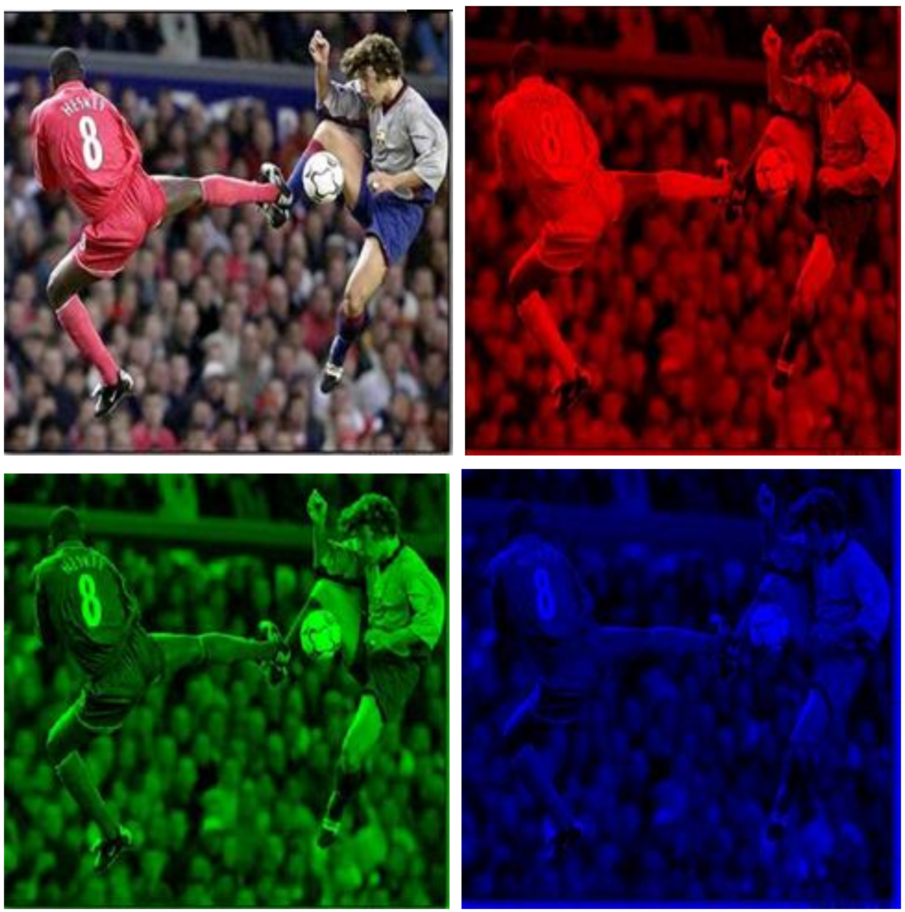
- Example: Color images in the *RGB color model* have channels for the red, green, and blue component; values u_i in each channel are in the set $\{0,1,\ldots,G_{max}\}$ (like a gray-value image)

[90, 0, 53]

[249, 215, 203]

[213, 60, 67]

Examples of RGB Color Images



Red

Green

Blue

Outline

- What is a digital image?
- Image Sensing and Acquisition
- Image Sampling and Quantization
- Image Representation
- **Color Models**
- Thresholding
- Image Resizing

Color Fundamentals

- **Why color image processing?**
  - Color is powerful in identifying and extracting objects
  - Humans can distinguish thousands of color shades and intensities when compared to only two dozens of shades of gray

- **Two major processing techniques**
  - Full color processing
    - The image is acquired using full –color sensor (TV camera, color scanner)
  - Pseudo color processing
    - Assign colors to monochromatic intensity image

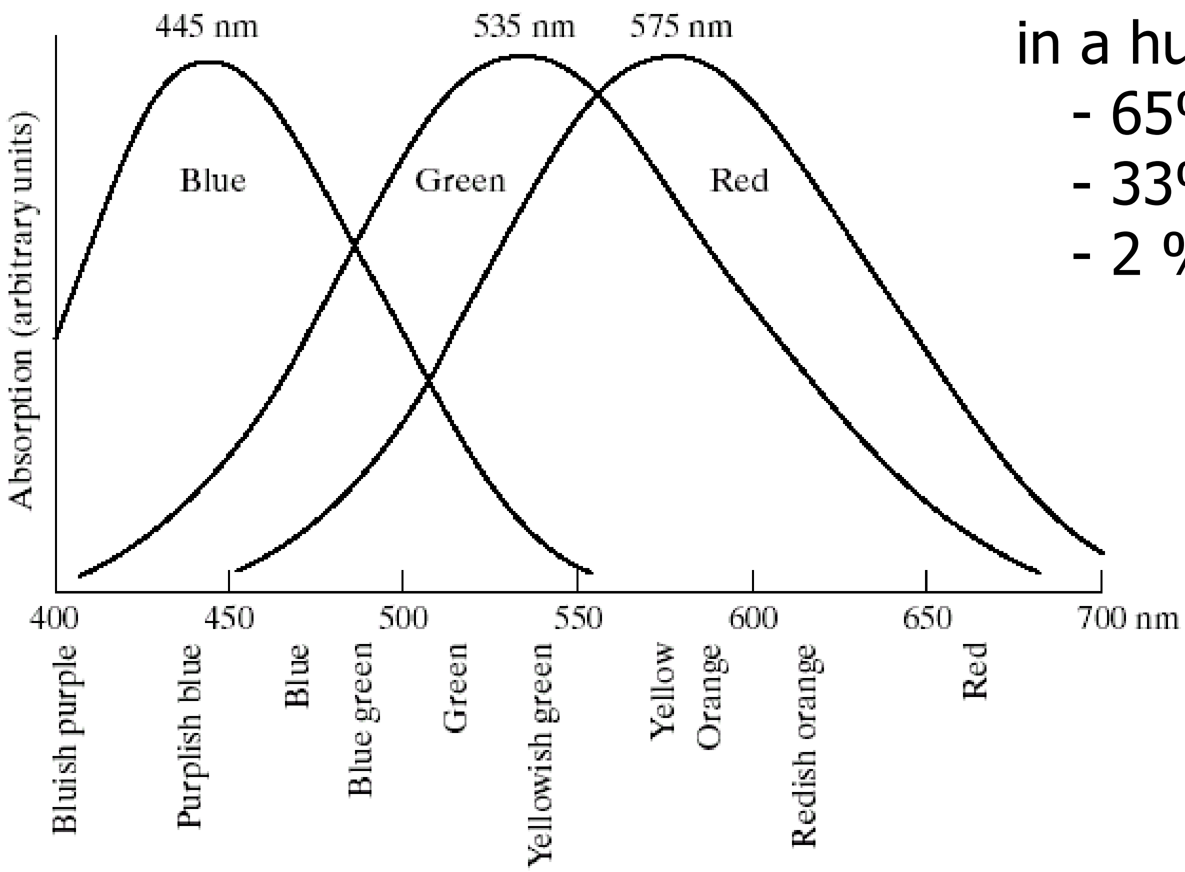
Color Fundamentals

- **Achromatic light**
  - Intensity is the only attribute that describes it
  - Light that is void of color
  - Gray level (shades of gray)
- **Chromatic light**
  - Spans the electromagnetic spectrum from approximately 400 to 700 nm
  - Quantities that describe a chromatic light source:
    - Radiance = total amount of energy flow from a light source (Watts)
    - Luminance = amount of energy received by an observer (lumens)
    - Brightness = intensity
  - Cones in the eye are responsible for color vision
    - Can be divided based on their sensitivity/absorption of light into three types: Red, Green, and Blue cones
  - Based on this experimental classification of the cones, these 3 colors are called the **primary colors**

Color Fundamentals

- **Chromatic light**
  - There is no single frequency that describe these primary colors
  - Standard values set by the CIE in 1931
    - 700 nm for Red
    - 546.1 nm for Green
    - 435.8 nm for Blue
  - Primary does not mean we can generate all colors by mixing these frequencies. Instead, we have to vary the 6 frequencies of these primary colors

Color Fundamentals

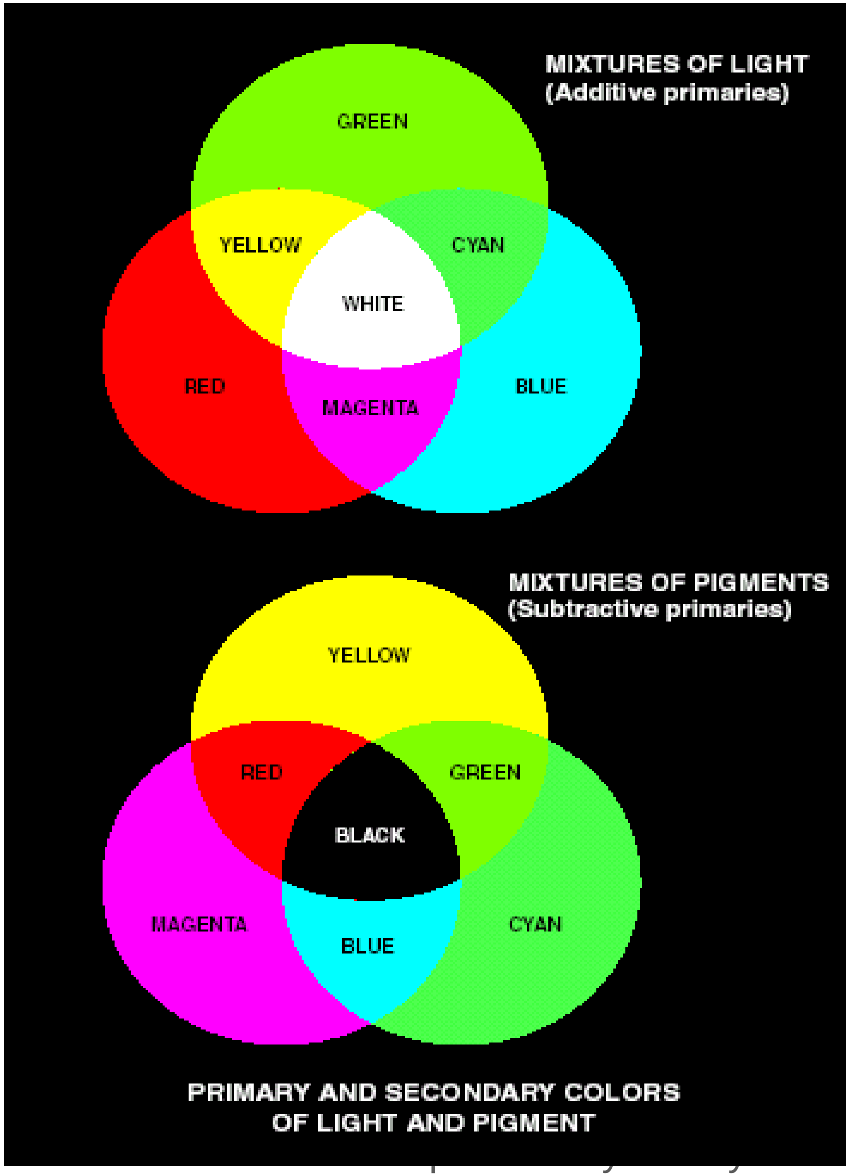


6-7 millions cones
in a human eye
- 65% sensitive to Red light
- 33% sensitive to Green light
- 2 % sensitive to Blue light

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

Color Fundamentals

- Chromatic light
  - Additive Primaries (primary colors of light)
    - Primary colors (R,G,B) can be added to produce secondary colors; magenta (M), cyan (C) , and yellow (Y)
    - Mixing the three primaries, in the right intensities, produce white
  - Subtractive Primaries (primary colors of pigment)
    - Secondary colors (RGB) can be added to produce primary colors; red, green, and blue
    - Mixing the three secondary colors, in the right intensities, produce black



Color Fundamentals

- The amount of red, green, and blue required to form any particular light are called the *tristimulus* values, X, Y, and Z, respectively.
- We can specify any color by its trichromatic coefficients

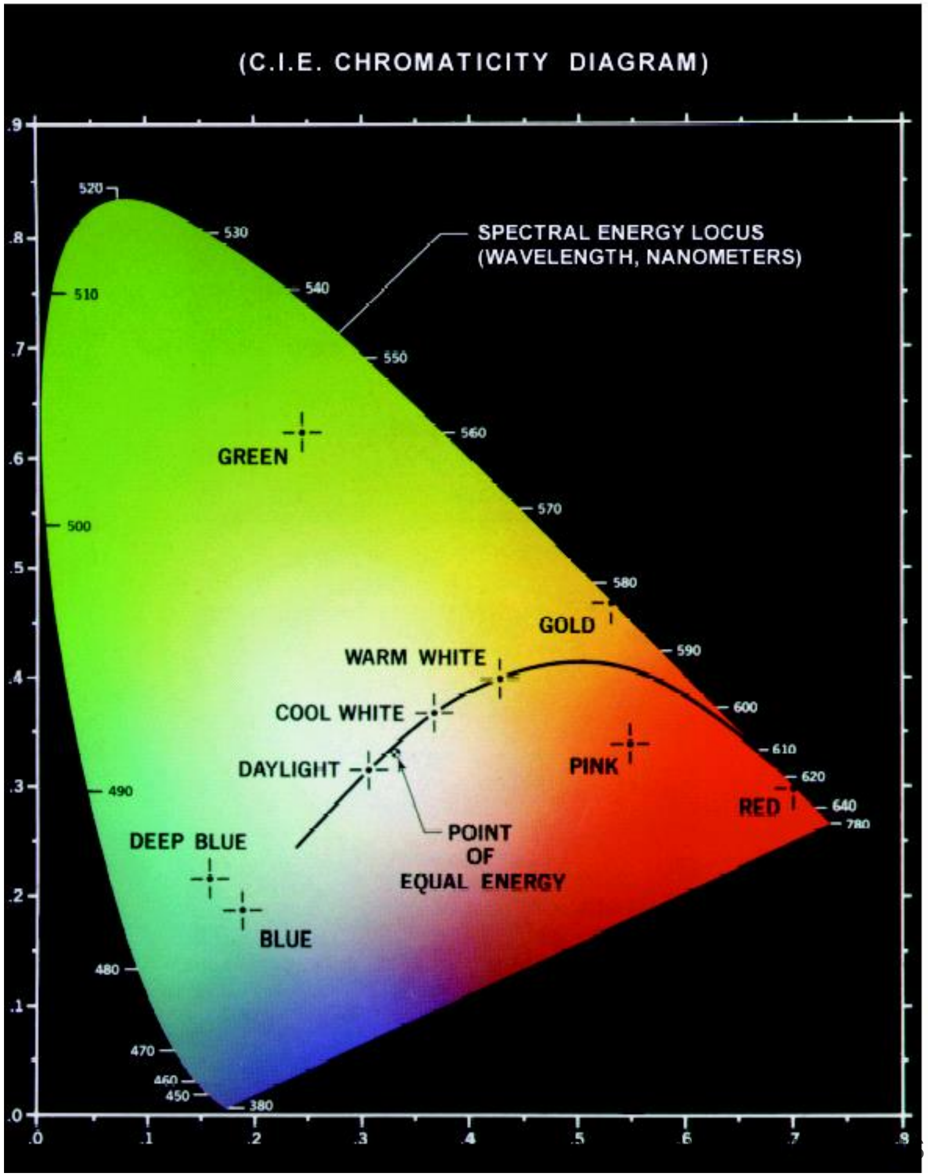
$$x = \frac{X}{X+Y+Z} \quad y = \frac{Y}{X+Y+Z} \quad z = \frac{Z}{X+Y+Z}$$

$$x + y + z = 1$$

- In order to determine the appropriate tristimulus values for any color, we use experimental tables or curves, e.g. *the chromaticity diagram*

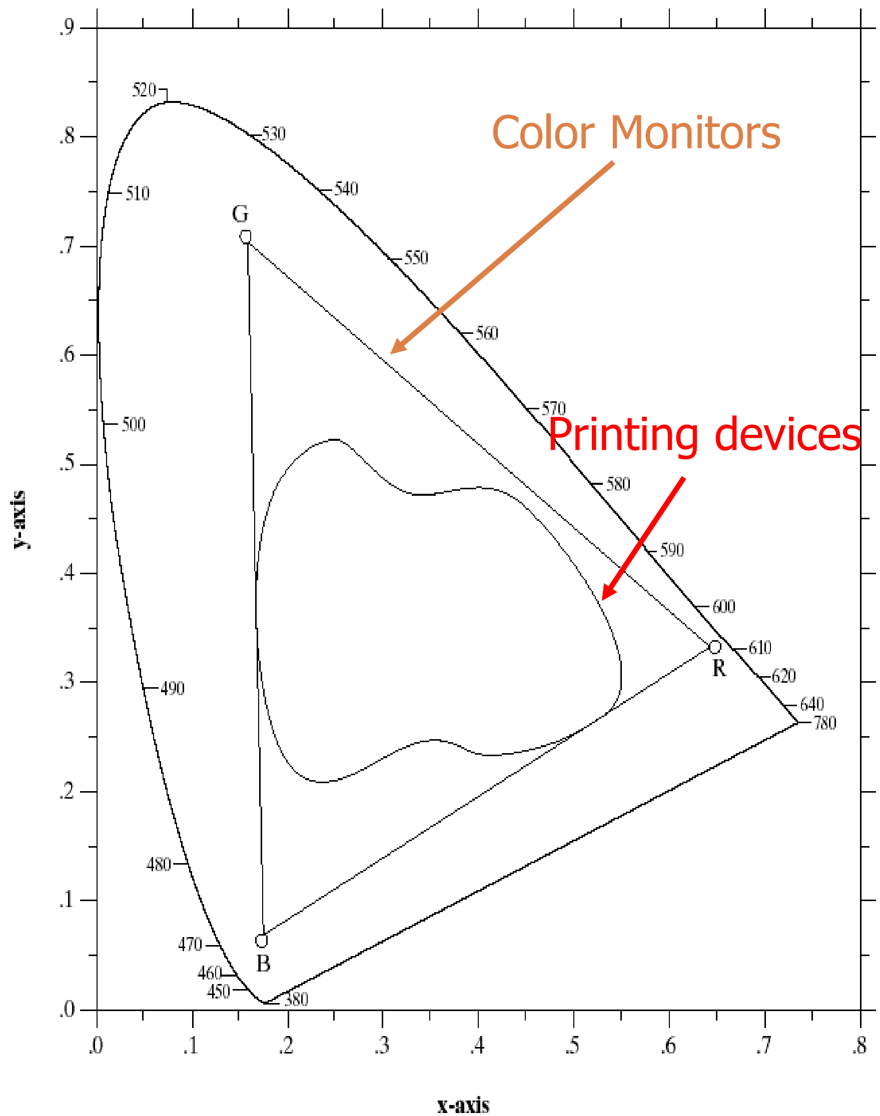
Color Fundamentals

- **The CIE Chromaticity Diagram**
  - Very useful in color mixing
  - It shows the color composition as a function of x (red) and y(green)
  - To determine z (blue) value for any color, use $z = 1 - (x+y)$
  - Colors on the boundary are fully saturated
  - Any point not on the boundary is a mix of colors
  - The point of equal energy defines color white



Color Fundamentals

- **The CIE Chromaticity Diagram**
  - Very useful in color mixing
  - A line connecting two points in the diagram defines all color variations that can be produced by combining these color additively
  - Three points in the diagram define a triangle. The point inside the triangle represent all possible colors that can be obtained by mixing different intensities of the three colors

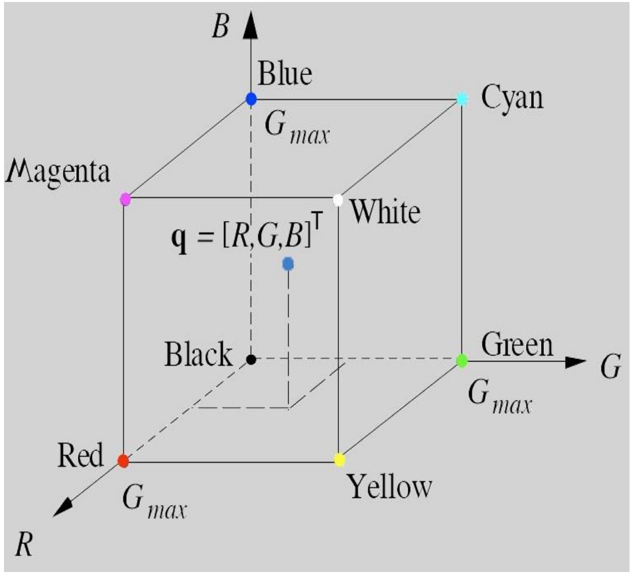


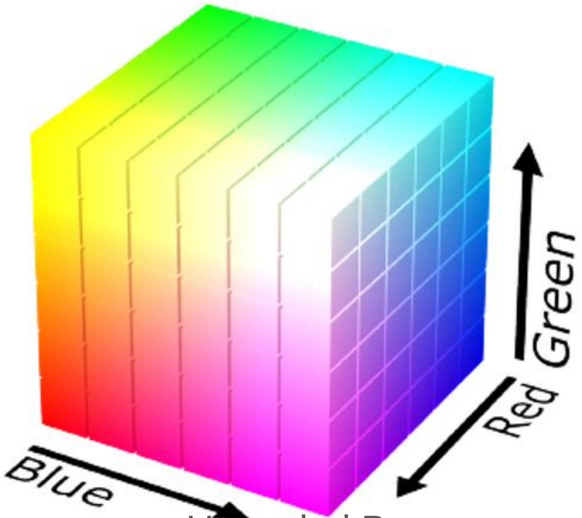
Color Models

- Color models/spaces/systems facilitate the specification of colors following some standard way
- A color model specifies a subspace within some coordinate system in which each color is represented as a point
- Classification of color models
  - Hardware-oriented
    - Generate colors in hardware
    - RGB, CMY, and CMYK
  - Software-oriented
    - The ultimate use is manipulation and processing of color images
    - HSI

The RGB Color Space

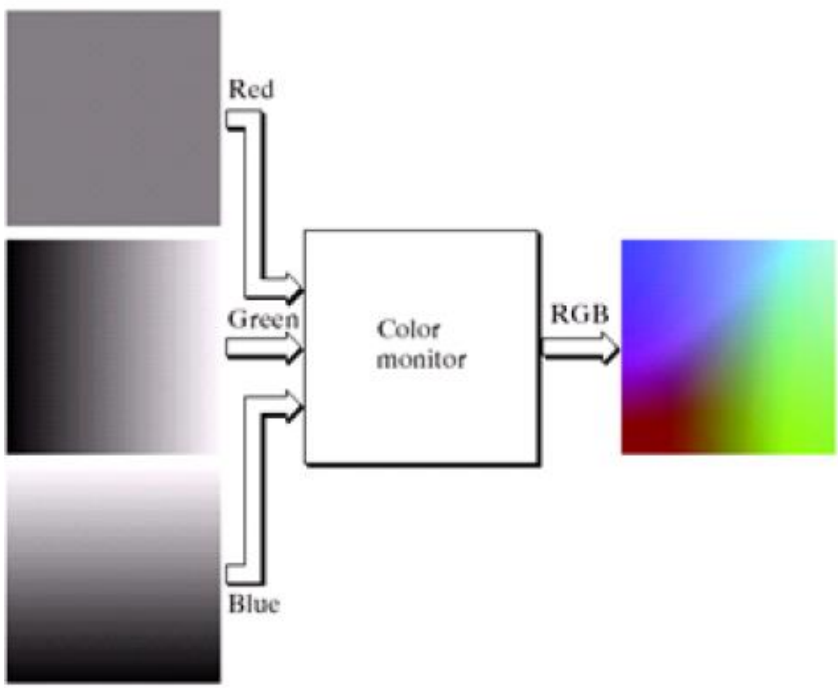
- $0 \leq R,G,B \leq G_{max}$, image I with pixel values u = (R,G,B)
- If G_{max} = 255 then 16,777,216 different colors
- u = (255,0,0) for Red, u = (255,255,0) for Yellow, and so forth
- Diagonal in cube from White at (255,255,255) to Black at (0,0,0)
- Gray-levels (u,u,u) are not colors
- q = (R,G,B) in RGB cube defines a color or a gray-level
- *Intensity* given by the mean $M = \frac{R+G+B}{3}$

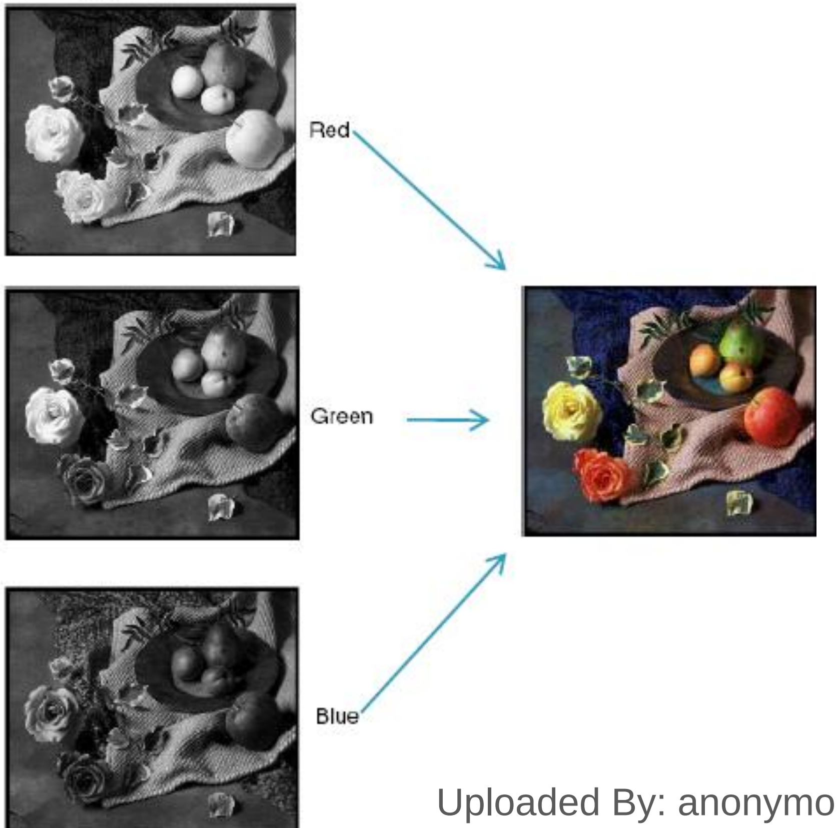




The RGB Color Model

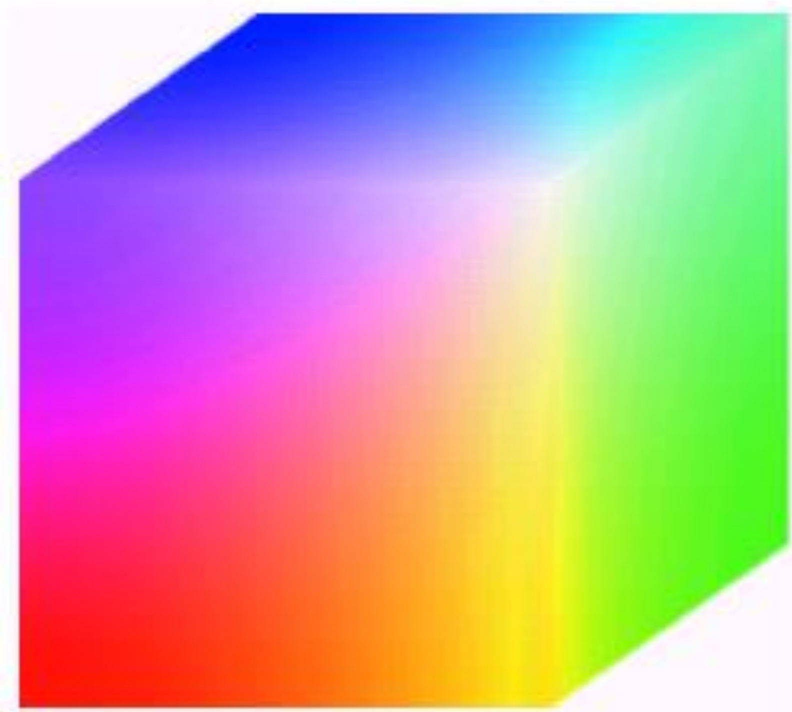
- Images represented in the RGB color model consist of three component images.
- When fed into the RGB monitor, they combine to produce the composite color image





The RGB Color Model

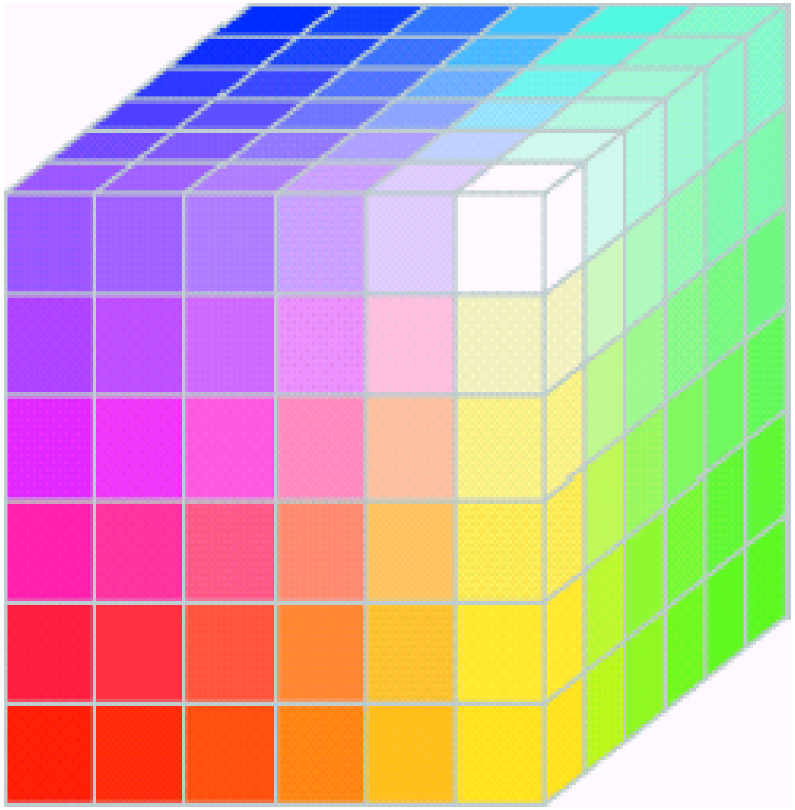
- **Full RGB Colors**
  - Each of the R,G, and B images are 8-bit,
  - The number of bits per pixel in the color image (pixel depth) is 24-bit
  - Total number of colors is 2^{24} = 16 M



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

The RGB Color Model

- **Safe RGB Colors**
  - Uses 256 colors
  - Colors are chosen such that they can be reproduced faithfully independent of hardware
  - Actually, 40 colors are processed differently by different operating systems
  - A safe color is formed by three RGB values. However, the values can be any of the following six values:: 0, 51,102,153,204, or 255.



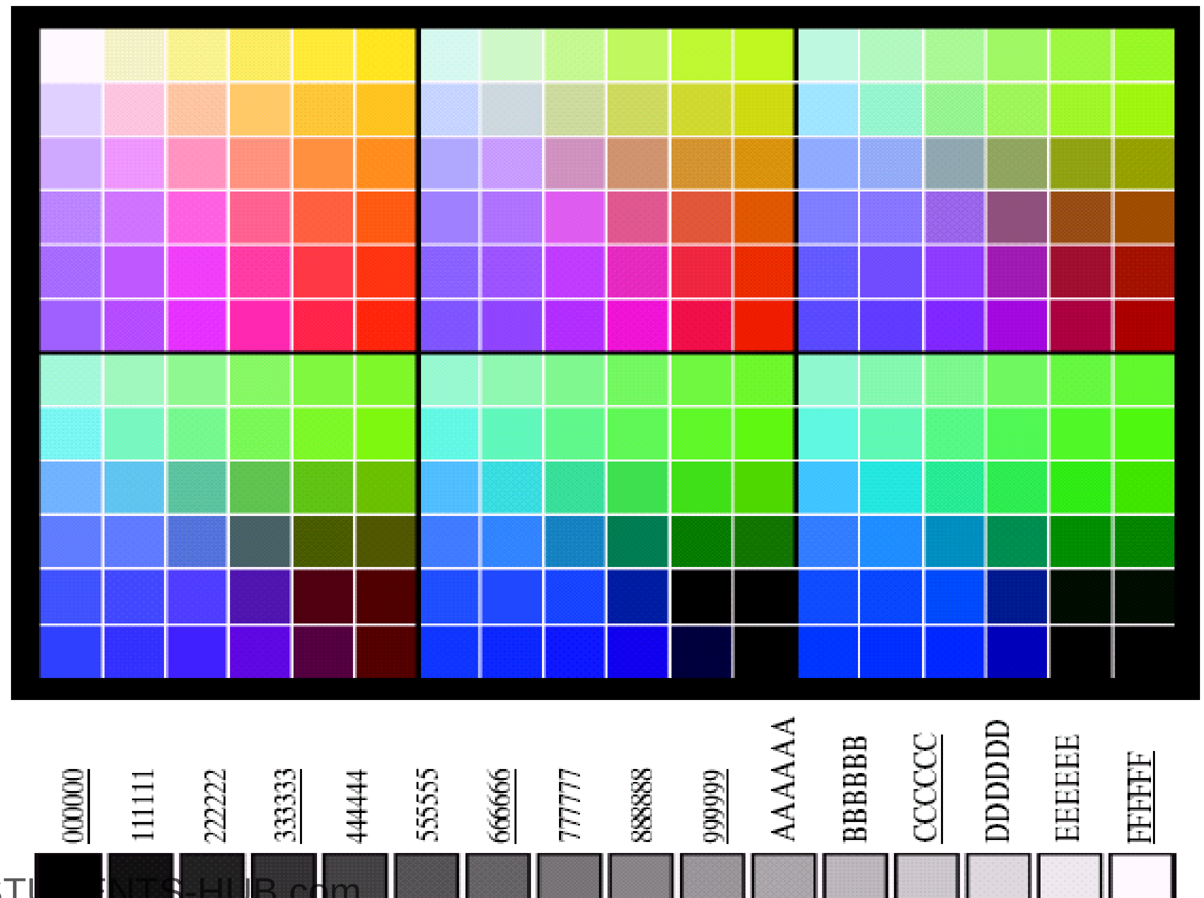
The RGB Cube is divided into 6 intervals on each axis to achieve the total $6^3 = 216$ common colors.

Valid colors are on the surface only

The RGB Color Model

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.



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).

The CMY Color Model

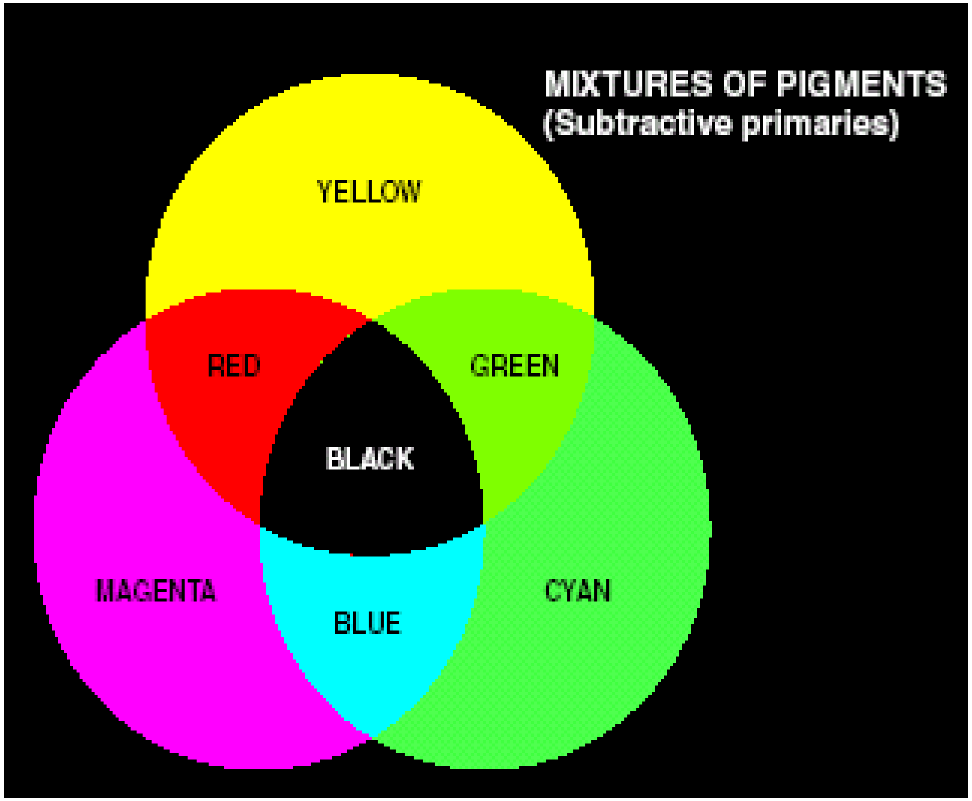
- Uses secondary colors, or the primary colors of pigments, cyan, magenta, and yellow to represent colors
- Used commonly in color printers
- Conversion between RGB and CMY

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

- Combining the three secondary colors should produce black. In practice, they produce muddy black. To produce black, a fourth color, black, is added.
- This is known as the CMYK, or four-color printing system

The CMY Color Model

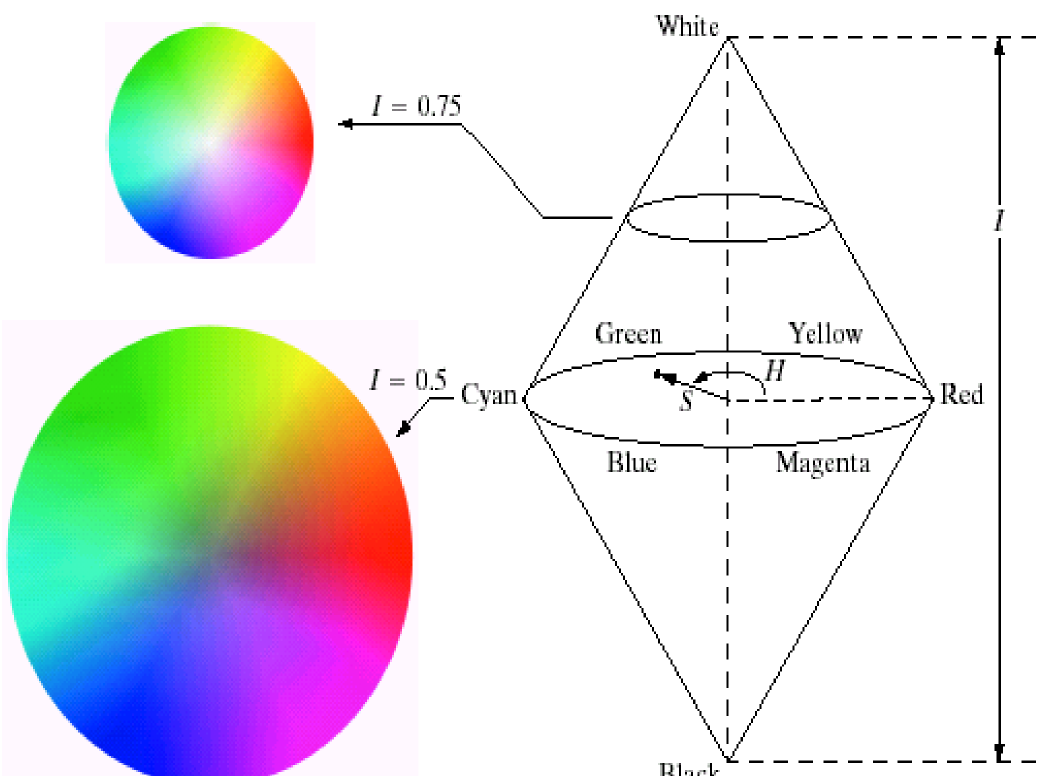
$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$



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

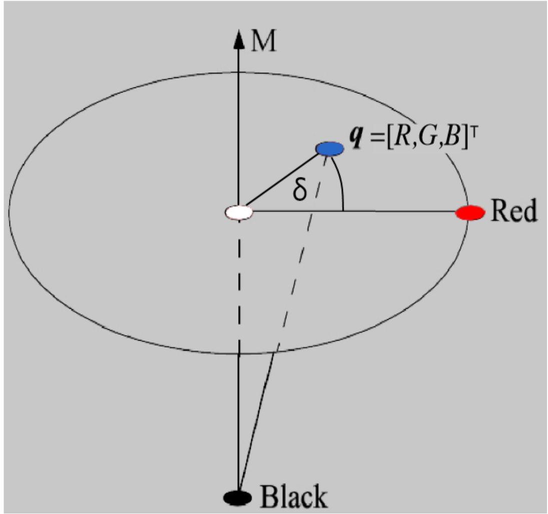
The HSI Color Model

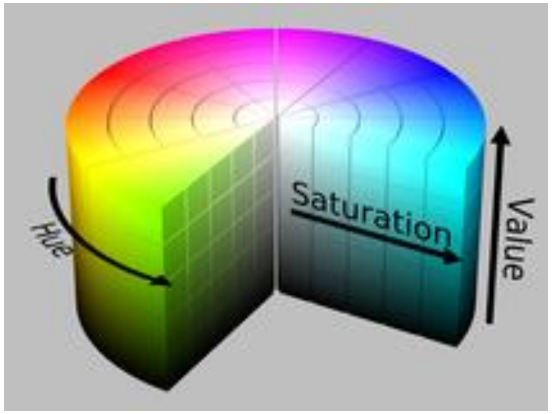
- The RGB and CMY models are well suited for hardware implementation
- It is often hard to use them in describing colors the way humans do
- Humans describe color by its hue (H), saturation (S), and intensity (I)
- These descriptors are the basis of the HSI color model



The HSV/HSI Color Model

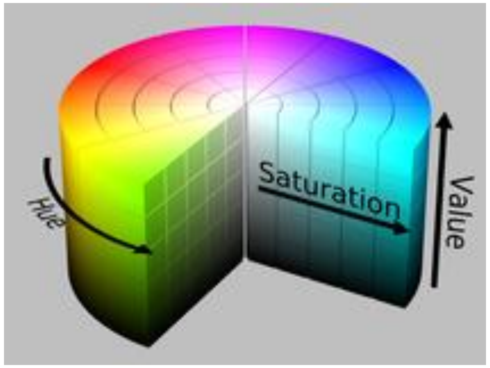
- Assume that a plane cuts RGB cube orthogonal to gray-level diagonal
- q = (R, G, B) incident with plane but not on diagonal
- The intensity axis: along the gray-level diagonal in the RGB cube
- The intersection of the plane with the intensity axis gives us the intensity component of the color
- Identify one color (here, Red) as reference color

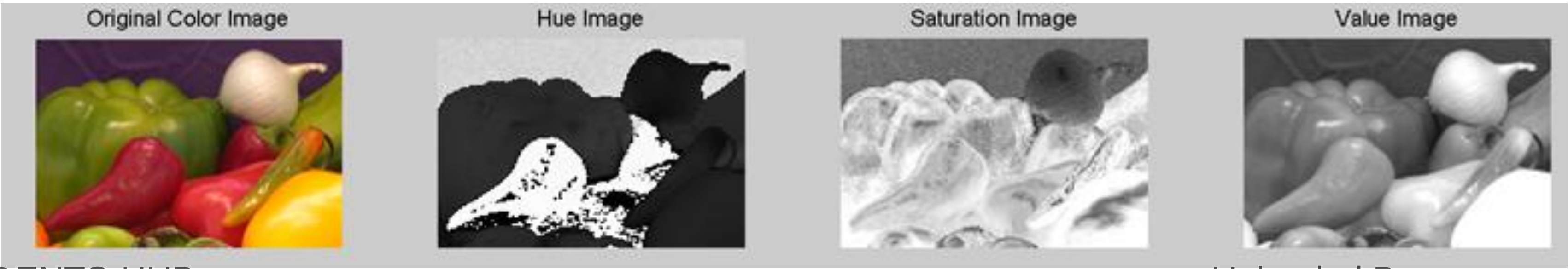




The HSV/HSI Color Model

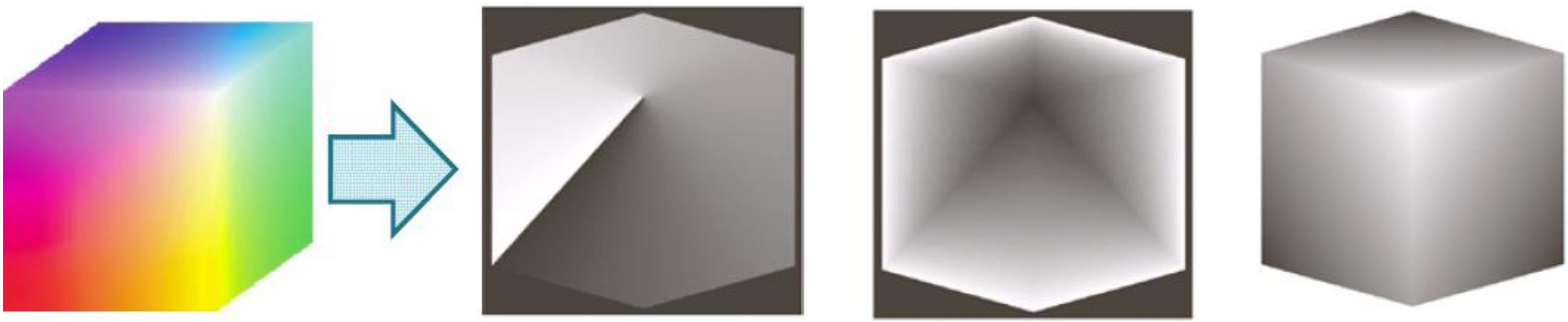
- The hue (H) of a color refers to which pure color it resembles (angle with respect to reference color)
- The saturation (S) of a color describes how white the color is
  - Sometimes, saturation appears as a range from just 0-1, where 0 is white, and 1 is a primary color.
- The value (V) or Intensity (I) of a color, also called its *brightness*, describes how dark the color is. It describes the brightness from 0-100 percent, where 0 is completely black, and 100 is the brightest and reveals the most color.



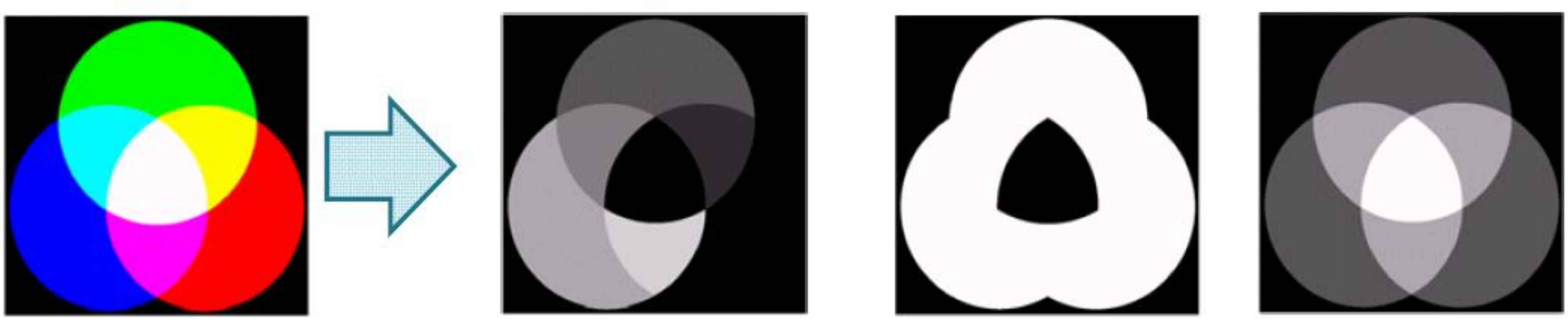


The HSI Color Model

- **Manipulating HSI Component Images**

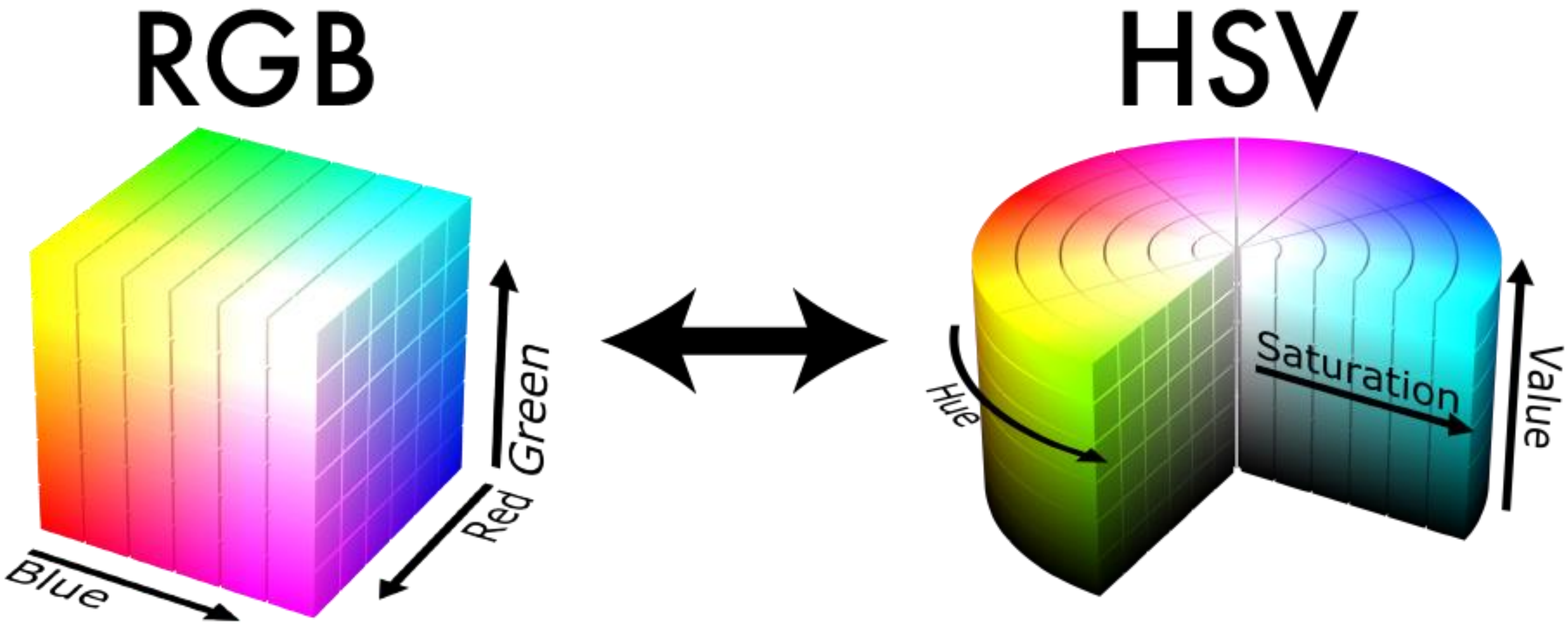


Hue Saturation Intensity



Pure colors Hue Saturation Intensity

Color Models Conversion



Color Models Conversion

- **Converting RGB colors into HSI**

  Given an image in RGB format, with normalized R, G, and B values, we can compute the HSI components by

  - The Hue Component

$$H = \begin{cases} \theta & , \text{ if } B \leq G \\ 360 - \theta & , \text{ if } B > G \end{cases} \Rightarrow \theta = cos^{-1}\left\{ \frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\left[(R-G)^2+(R-B)(R-G)\right]^{\frac{1}{2}}} \right\}$$

θ is measured with respect to the red axis

  - The Saturation Component

$$S = 1 - \frac{3}{R+G+B} min(R, G, B)$$

  - The Intensity Component

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

Color Models Conversion

- **Converting HSI colors into RGB**

  Given an image in HSI format , we have three different cases based on the value of H

RG sector ($0° \le H < 120°$)	$R = \left[1 + \frac{S \cos H}{\cos(60^o - H)}\right] I$ $B = (1 - S) I$ $G = 3I - (R + B)$
GB sector ($120° \le H < 240°$)	$H = H - 120^o \quad , \quad R = (1 - S) I$ $G = \left[1 + \frac{S \cos H}{\cos(60^o - H)}\right] I$ $B = 3I - (R + G)$
BR Sector ($240° \le H \le 360°$)	$H = H - 240^o \; , \; G = (1 - S) I$ $B = \left[1 + \frac{S \cos H}{\cos(60^o - H)}\right] I$ $R = 3I - (B + G)$

Color Image Processing

- There are 2 types of color image processes

  1. **Pseudocolor image process**: Assigning colors to gray values based on a specific criterion. Gray scale images to be processed may be a single image or multiple images such as multispectral images

  2. **Full color image process**: The process to manipulate real color images such as color photographs.

Outline

- What is a digital image?
- Image Sensing and Acquisition
- Image Sampling and Quantization
- Image Representation
- Color Models
- **Thresholding**
- Image Resizing

Thresholding

- **Thresholding** is the simplest method of image segmentation.
- From a grayscale image, thresholding can be used to create binary images
- During the thresholding process, individual pixels in an image are marked as “object” pixels if their value is greater than some threshold value (assuming an object to be brighter than the background) and as “background” pixels if their value is less than threshold value
- Typically, an object pixel is given a value of “1” while a background pixel is given a value of “0”
- The key parameter in the thresholding process is the choice of the threshold value

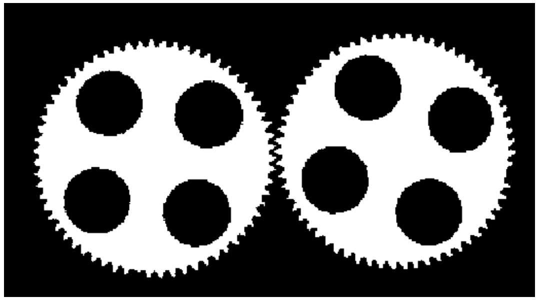
Why Thresholding

- **Simplifying Image Analysis:**
  - Reduces Complexity: Converting an image to a binary format makes it easier to analyze by reducing the amount of data.
  - Speeds Up Processing: Binary images contain less information, so algorithms that operate on these images are faster and more efficient.
- **Segmentation:**
  - Object Identification: Thresholding helps in segmenting objects of interest from the background by separating pixel intensities.
- **Feature Extraction:**
  - Region-based Analysis: It facilitates the extraction of meaningful features like area, perimeter, or shape of objects for further analysis.
- **Preprocessing for Further Algorithms:**
  - Preparation for Morphological Operations: Techniques like erosion, dilation, and other morphological operations require a binary image, for which thresholding is often the first step.

Why do we need thresholding?

- Binary image is used in a number of practical applications
- Document analysis

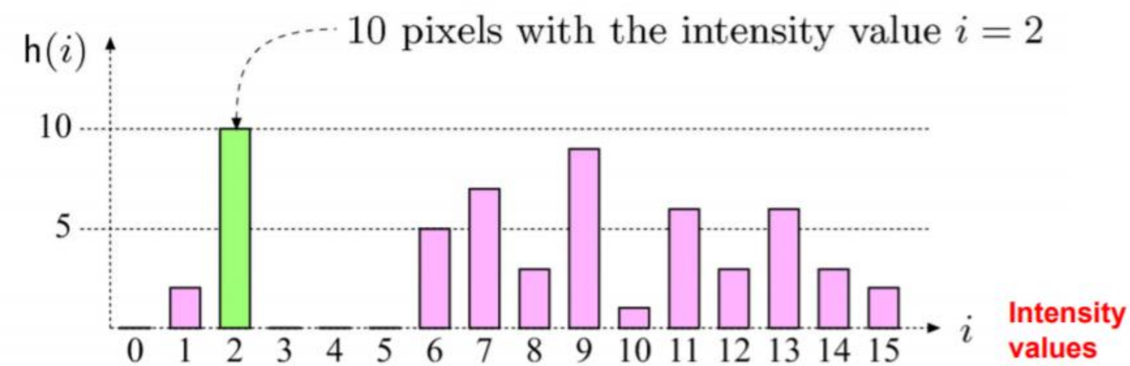
- Industrial inspection



- Medical imaging

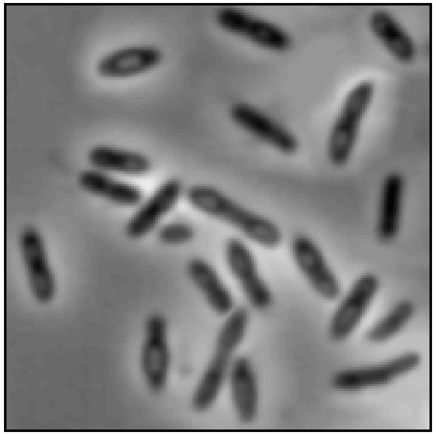
How to select threshold value?

- One idea is to use histogram.
- Histogram of an image provides the frequency of the brightness (intensity) value in the image.
- Histogram captures the distribution of gray levels in the image.
- How frequently each gray level occurs in the image



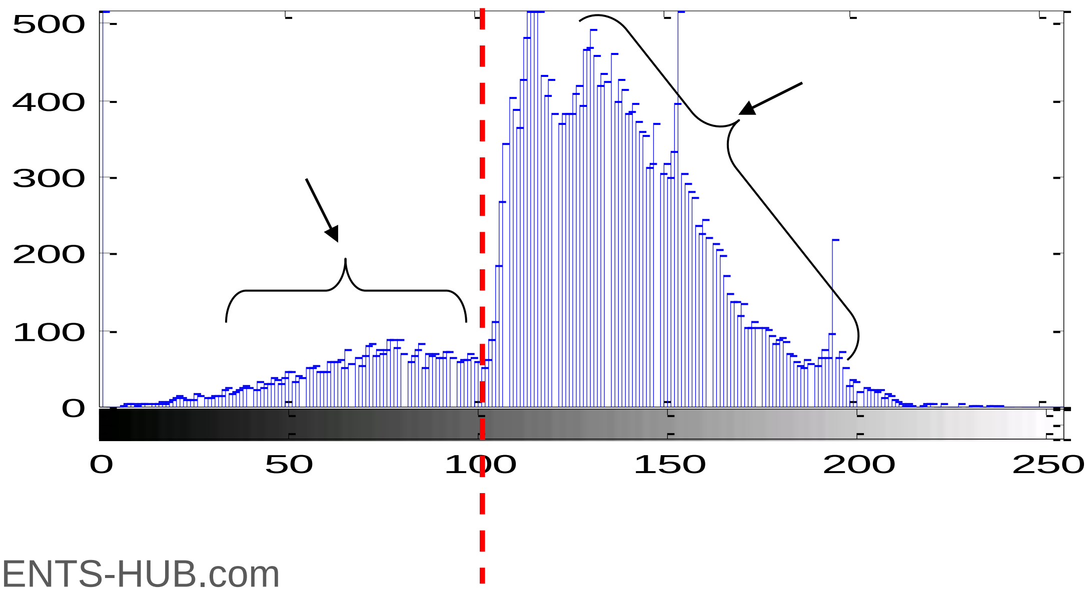
- How can we use a histogram to separate an image into 2 (or several) different regions?

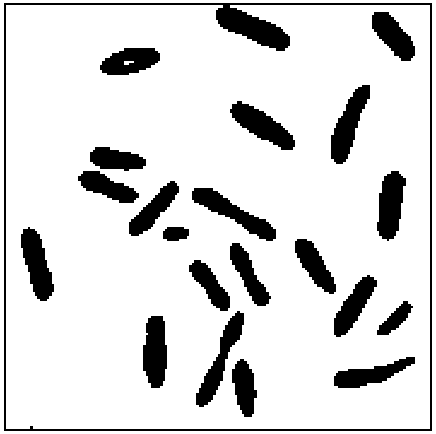
Thresholding



$$g(x,y) = \begin{cases} 1 & f(x,y) > T \\ 0 & f(x,y) < T \end{cases}$$

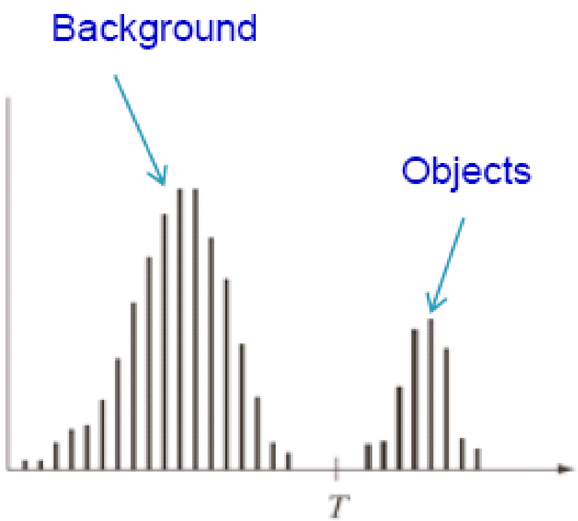
T = 102





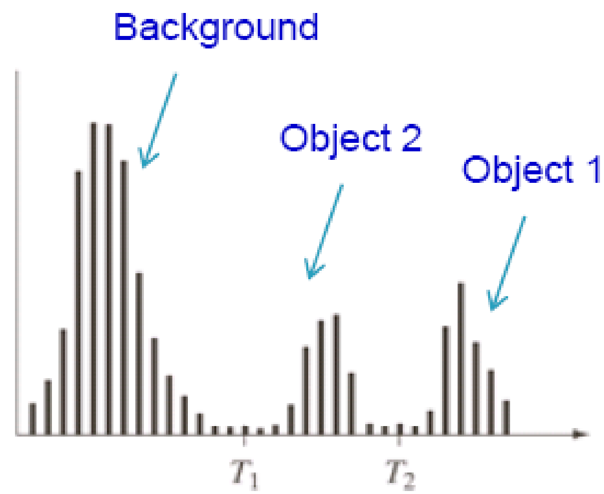
After thresholding

Thresholding



Histogram of an image with light objects and dark background . Segmenting the objects can be achieved by specifying T such that

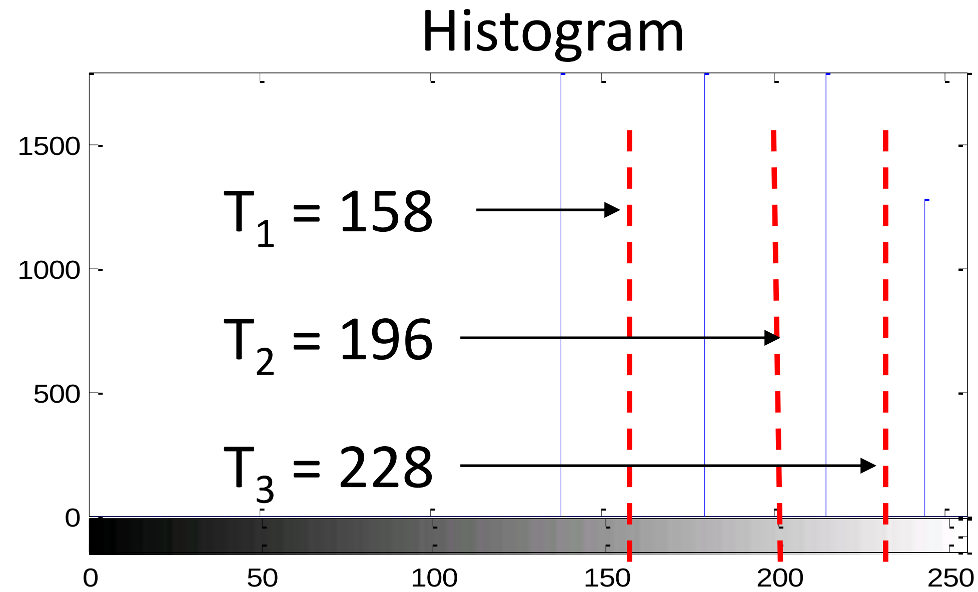
$$g(x,y)=\begin{cases} 1, & f(x,y) > T \\ 0, & f(x,y) \le T \end{cases}$$



More challenging situation where we need to specify two thresholds in order to segment the object from background (Multiple Thresholding)

$$g(x,y)=\begin{cases} a, & f(x,y) > T_2 \\ b, & T_1 < f(x,y) \le T_2 \\ c, & f(x,y) \le T_1 \end{cases}$$

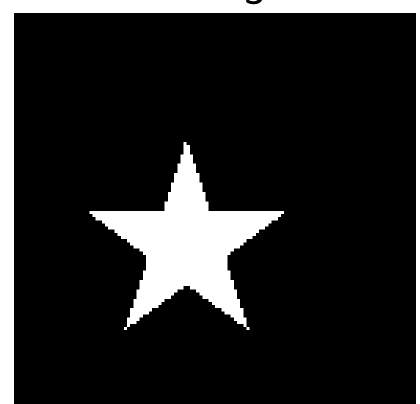
Thresholding



$T_1 < P < T_2$

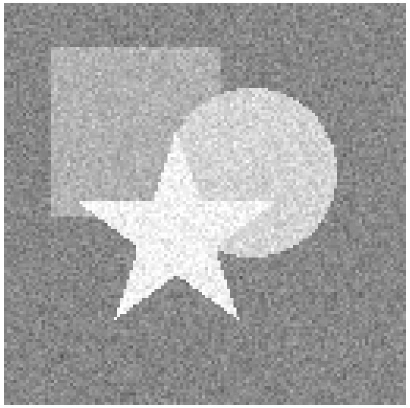
$T_2 < P < T_3$

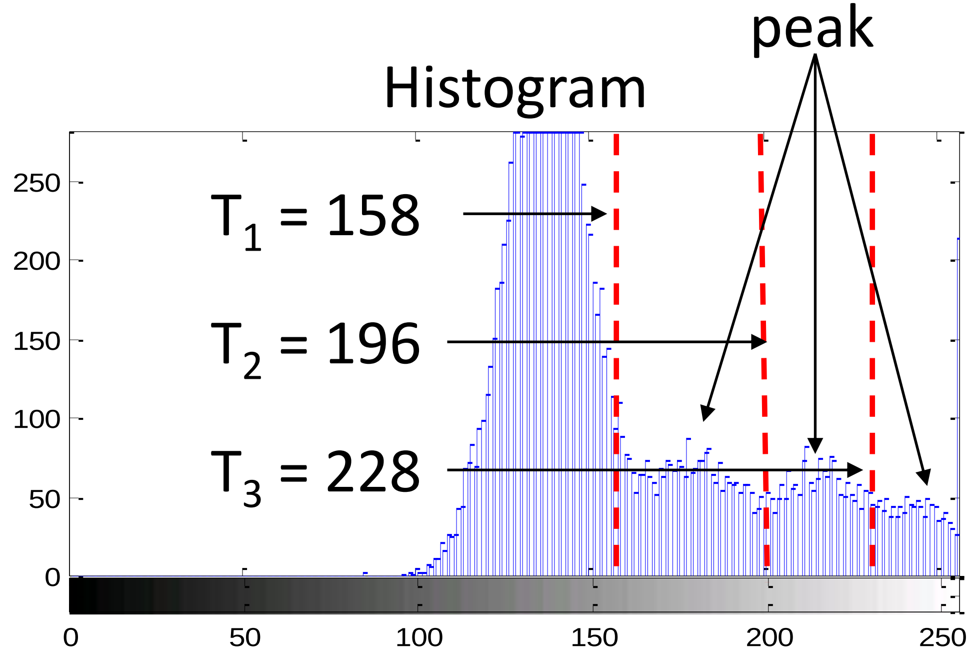
$P > T_3$



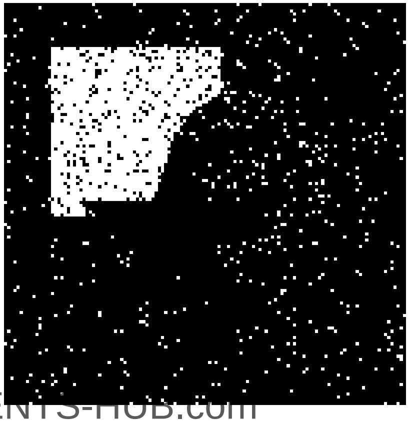
Thresholding - Effect of Noise

Image degraded by Gaussian noise (s =12)





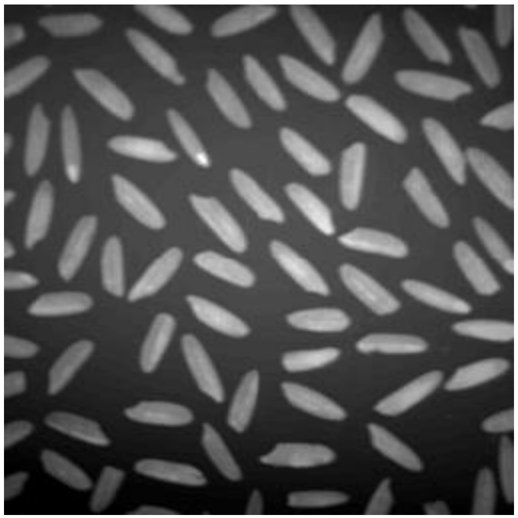
$T_1 < P < T_2$

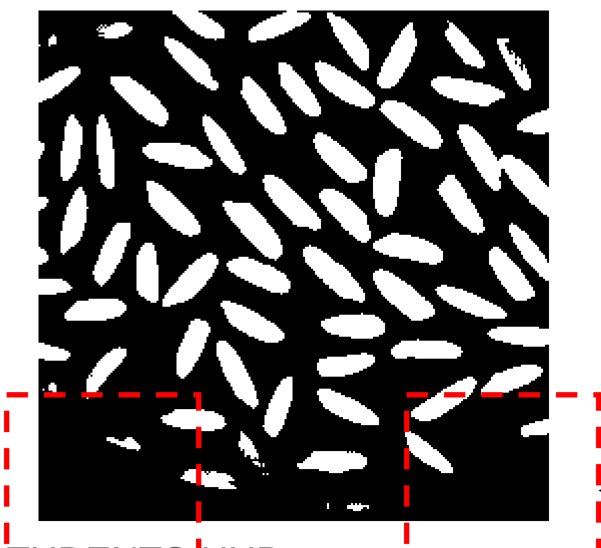


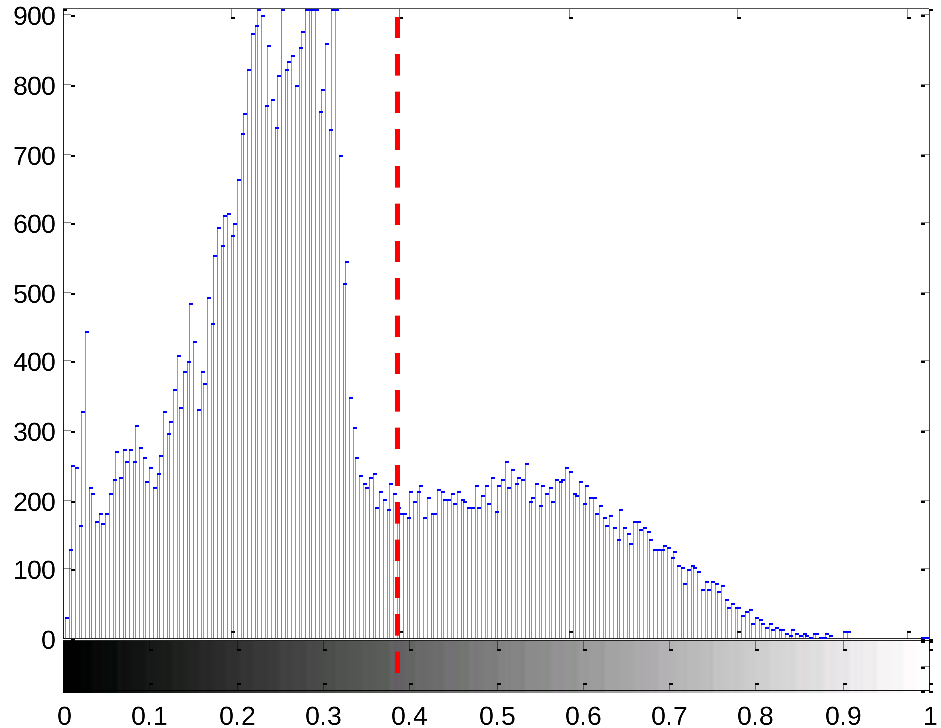
$T_2 < P < T_3$

$P > T_3$

Thresholding - Effect of Illumination







T=0.4

Error

Basic Global Thresholding

- When the intensity distribution of objects and background is sufficiently enough, it is possible to use a single global threshold value; a very common situation in practice
- To find such global threshold **automatically**

1. Select an initial estimate for the global threshold value, T
2. Segment the image using T to produce two groups of pixels; G1 containing pixels with intensity values > T and G2 with intensity values ≤ T
3. Compute the mean intensity value for each group, m_1 and m_2, and define the new threshold by

$$T = (m_1 + m_2)/2$$

4. Repeat steps 2 and 3 until the change in T is smaller than a specified value ΔT

Basic Global Thresholding

- **Example**

Original

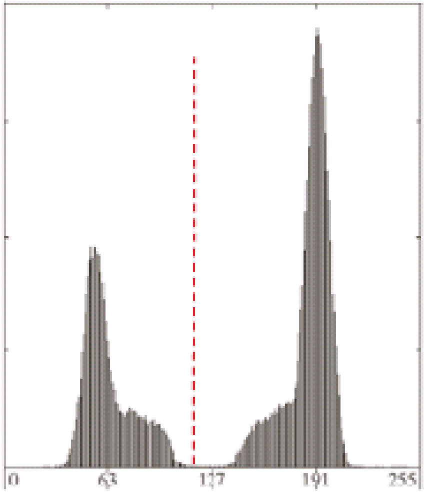


Image Histogram

Segmentation Result

Initial threshold = mean intensity of image

Optimal threshold = 125.4

Three iterations

Otsu's Optimal Global Thresholding

- Let $\{0,1,2,..L-1\}$ denote L distinct levels in MxN image . with normalized histogram whose components are $p_i = n_i/MN$
- Suppose we threshold the image with a threshold $T(k) = k$, $0<k<L-1$, to produce two classes C_1 and C_2, where C_1 is the set of pixels with intensity levels in $[0,k]$ and C_2 is the set of pixels with intensity levels in $[k+1,L-1]$
- Based on this threshold, the probability that a pixel belongs to C_1 (the probability of class C_1 occurring) is

$$P_1(k) = \sum_{i=0}^{k} p_i$$

and for C_2 is

$$P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$$

Otsu's Optimal Global Thresholding

- The mean intensity value m_1 of pixels assigned to C_1 is

$$m_1(k) = \sum_{i=0}^{k} i \times P(i / C_1) = \sum_{i=0}^{k} i \times P(C_1 / i) P(i) / P(C_1) = \frac{1}{P_1(k)} \sum_{i=0}^{k} i \times p_i$$

- Similarly, the mean intensity m_2 value of pixels assigned to C_2 is

$$m_2(k) = \sum_{i=k+1}^{L-1} i \times P(i / C_2) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} i \times p_i$$

- The average intensity of the entire image is given by

$$m_G = \sum_{i=0}^{L-1} i \times p_i$$

- The cumulative mean up to level k is $m(k) = \sum_{i=0}^{k} i \times p_i$

- We can easily verify

$$P_1 m_1 + P_2 m_2 = m_G$$
$$P_1 + P_2 = 1$$

Otsu's Optimal Global Thresholding

- In order to evaluate the goodness of the threshold at level k we use the metric (separability measure)

$$\eta(k) = \frac{\sigma_B^2(k)}{\sigma_G^2}$$

where σ_G^2 is the variance of the entire image and σ_B^2 is the between-class variance which is defined as

$$\sigma_B^2(k) = P_1(k)(m_1(k) - m_G)^2 + P_2(k)(m_2(k) - m_G)^2$$

$$= P_1(k)P_2(k)(m_1(k) - m_2(k))^2$$

$$= \frac{(m_G P_1(k) - m(k))^2}{P_1(k)(1 - P_1(k))}$$

- For optimal segmentation, we need to find the intensity level k that maximizes η as it implies larger seperability between classes

$$\eta(k^*) = \max_{0 \le k \le L-1} \sigma_B^2(k)$$

Otsu's Optimal Global Thresholding

- **Notes**
  - As the difference between m_1 and m_2 increases, σ_B^2 increases to indicate better seperability
  - The definition of the metric η assumes that $\sigma_G^2 > 0$. The variance of an image could be zero if it contains one intensity level only
  - In order to find the optimal threshold k^*, we have to compute σ_B^2 for all integer values of k and then pick the value that maximizes the between-class variance σ_B^2
  - Once the optimal threshold is found, we simply threshold the image into two classes using

$$g(x,y) = \begin{cases} 1\ ,\ f(x,y) > k^* \\ 0\ ,\ f(x,y) \leq k^* \end{cases}$$

Optimal Global Thresholding

- **Summary of Otsu's Algorithm**

1) Compute the normalized histogram of the input image and denote its components by p_i , i = 1,2,…,L-1
2) Compute the cumulative sum $P_i(k)$, for k = 0,1,2,…,L-1 using

$$P_1(k) = \sum_{i=0}^{k} p_i$$

3) Compute the cumulative means, m(k), for k = 0,1,2,…,L-1 using

$$m(k) = \sum_{i=0}^{k} i \times p_i$$

4) Compute the global intensity mean m_G using

$$m_G = \sum_{i=0}^{L-1} i \times p_i$$

5)
6) Compute the between-class variance $\sigma_B^2(k)$ for k = 0,1,2,…,L-1 using

$$\sigma_B^2(k) = \frac{(m_G P_1(k) - m(k))^2}{P_1(k)(1 - P_1(k))}$$

7) Pick Otsu's threshold k* that gives the maximum value for $\sigma_B^2(k)$
8) Obtain the separability measure at k* using $\eta(k^*) = \dfrac{\sigma_B^2(k^*)}{\sigma_G^2}$

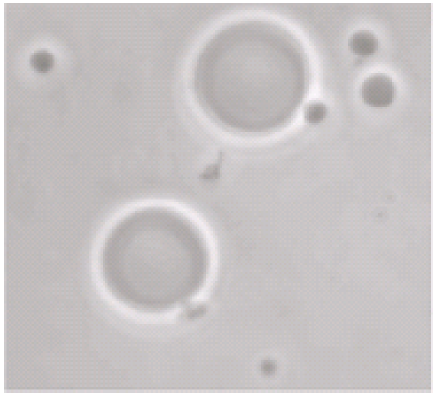
Otsu's thresholding: Example

original image

pixels above threshold

Optimal Global Thresholding vs. Otsu's

- **Example:** segmentation of the molecules from background



Polymersome cells image

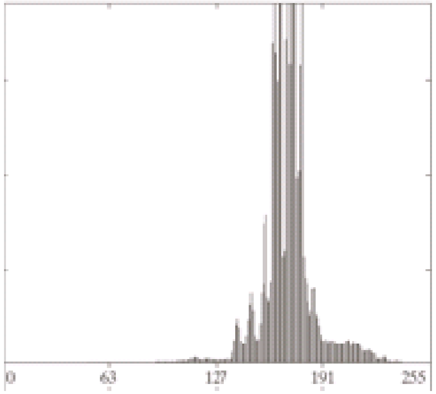


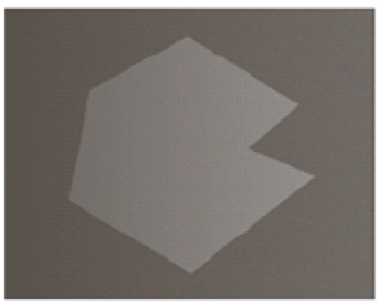
Image Histogram

Segmentation using basic global thresholding T* = 169

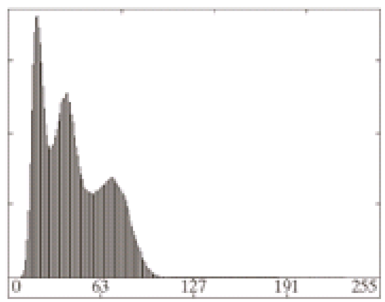
Segmentation using Otsu's global thresholding k* = 181

Variable Thresholding

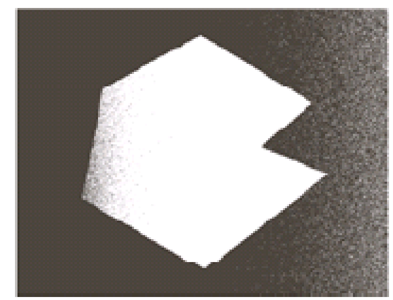
- **Variable Thresholding Based on Image Partitioning**
  - Subdivide the image into nonoverlapping rectangles to compensate in non-uniformities in illumination and/or reflectance. Threshold each subimage separately
  - The method works well when the objects and the background occupy regions of reasonably comparable size



Noisy image with nonuniform illumination



Histogram



Otsu's segmentation result

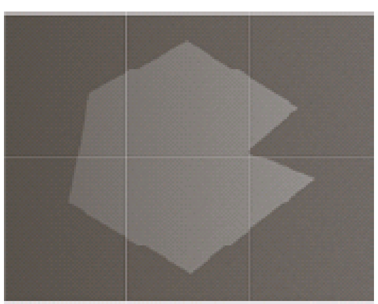
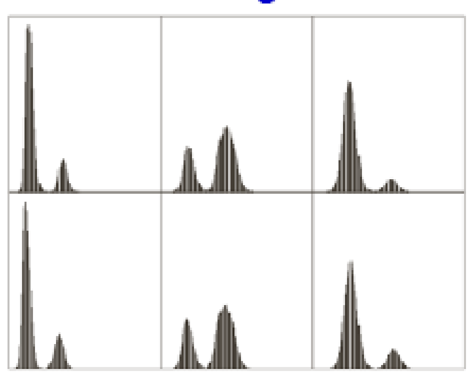


Image partitioning



Histogram of each subimage

Segmentation result after applying Otsu's method to each subimage

Other Thresholding/Binarization Techniques

- Niblack's method
- local-variance-based method by Sauvola
- Local adaptive method proposed by Bernsen
- Entropy-based method By Kapur
- learning framework for the optimization of the binarization methods by Cheriet

Outline

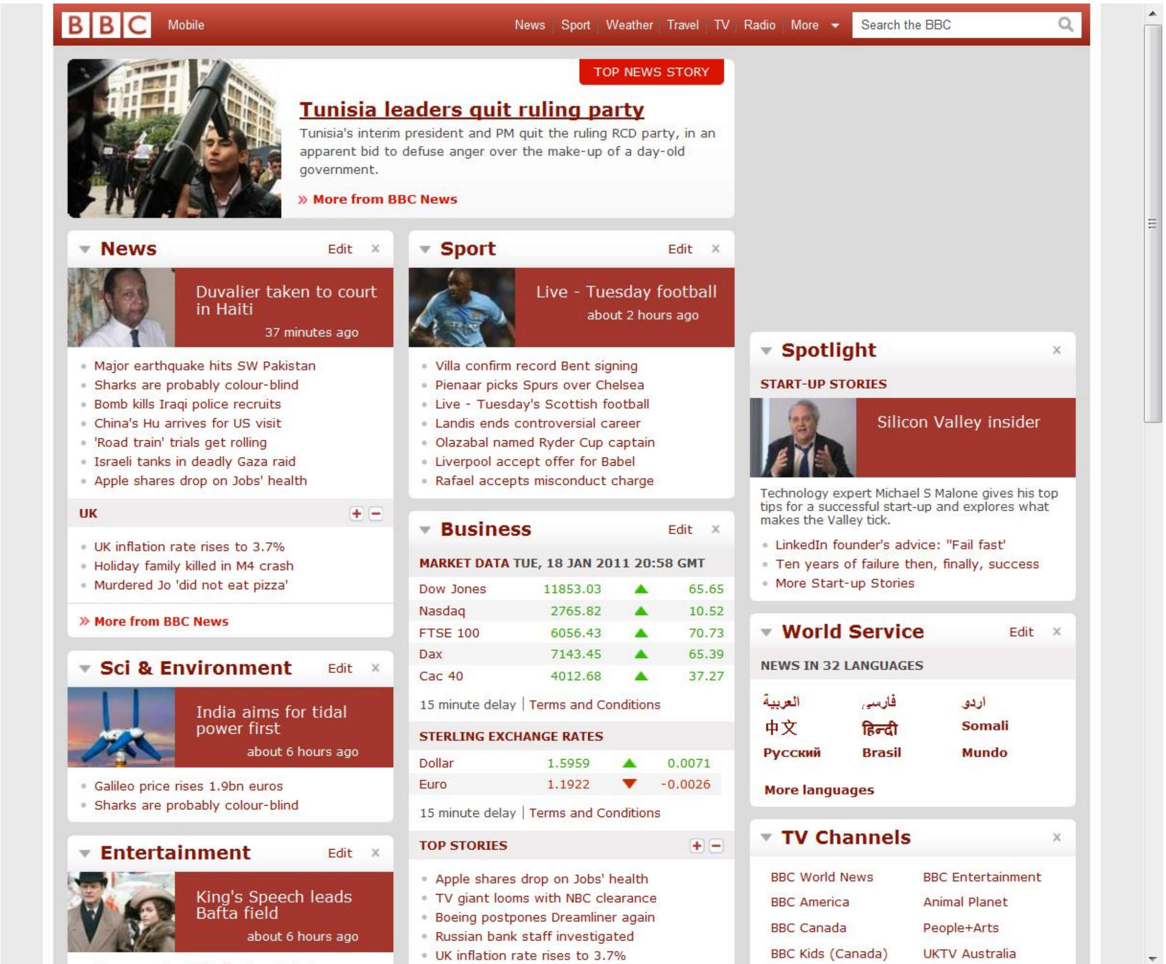
- What is a digital image?
- Image Sensing and Acquisition
- Image Sampling and Quantization
- Image Representation
- Color Models
- Thresholding
- **Image Resizing**

Image Resizing

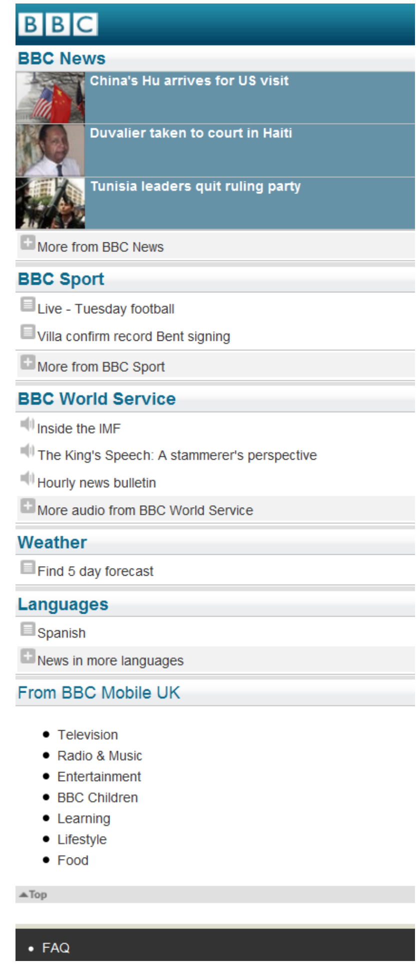
- Various display devices such as televisions, notebooks, personal digital assistants (PDAs), and cell phones, however, have different resolutions and aspect ratios.
- When optimizing websites for faster page load time, one of the most common errors we see is that images aren't the proper size, requiring the browser to resize the image every time the page loads. Since images can often make up the bulk of the bytes needed to load a web page, everything we can do to make our images smaller and easier to load will pay dividends in increased site speed.
- In such scenarios, image/video resizing is a natural choice, and thus the image/video resizing techniques are urgently important for display device manufacturers and web page designers.

Display Devices

Content Retargeting

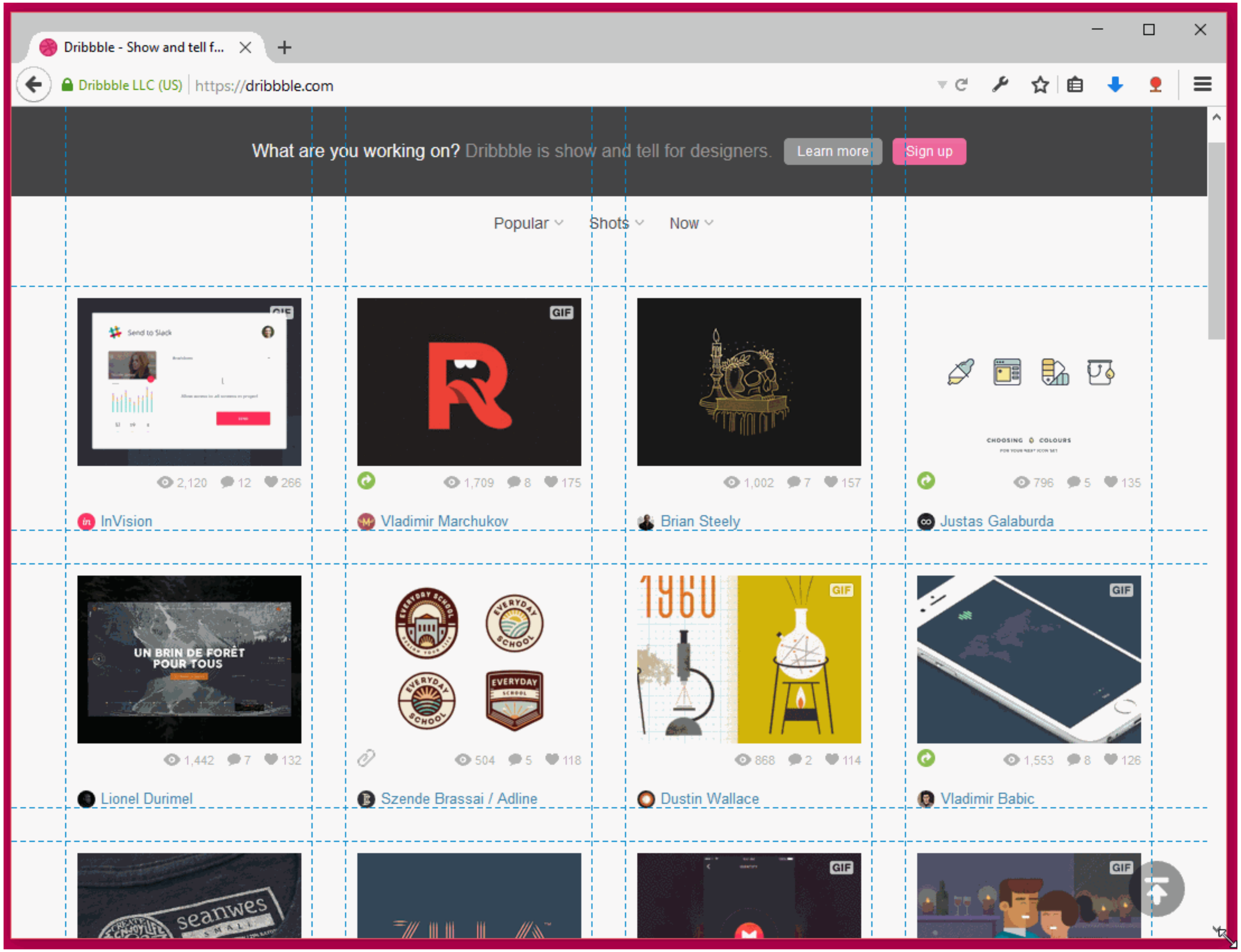


PC



iPhone

Page Layout


Dribbble - Show and tell f...
Dribbble LLC (US) https://dribbble.com
What are you working on? Dribbble is show and tell for designers. Learn more Sign up
Popular Shots Now
InVision
Vladimir Marchukov
Brian Steely
Justas Galaburda
Lionel Durimel
Szende Brassai / Adline
Dustin Wallace
Vladimir Babic

Simple Media Retargeting Operators

Image resizing approaches

- Traditional methods:
  - **Scaling/Interpolation**
- Content aware resizing
  - **Seam Carving**

Image Interpolation

- Interpolation works by using known data to estimate values at unknown points.
- Image interpolation works in two directions, and tries to achieve a best approximation of a pixel's intensity based on the values at surrounding pixels.
- An important tool used in zooming, shrinking, rotation, and geometric corrections
  - Zooming: enlarge a MxN image to OxP image
    - Can be simply done by row/column replication
  - Shrinking: reduce a MxN image to QxR image
    - Can be simply done by row/column deletion
- Interpolation methods
  - Nearest neighbor
  - Bilinear
  - Bicubic

Image Shrinking by Pixel Deletion

Original

Image shrinked by 2x2

Image shrinked by 4x4

Image shrinked by 16x16

Zooming by Pixel Replication

Image zoomed by factor of 4x4

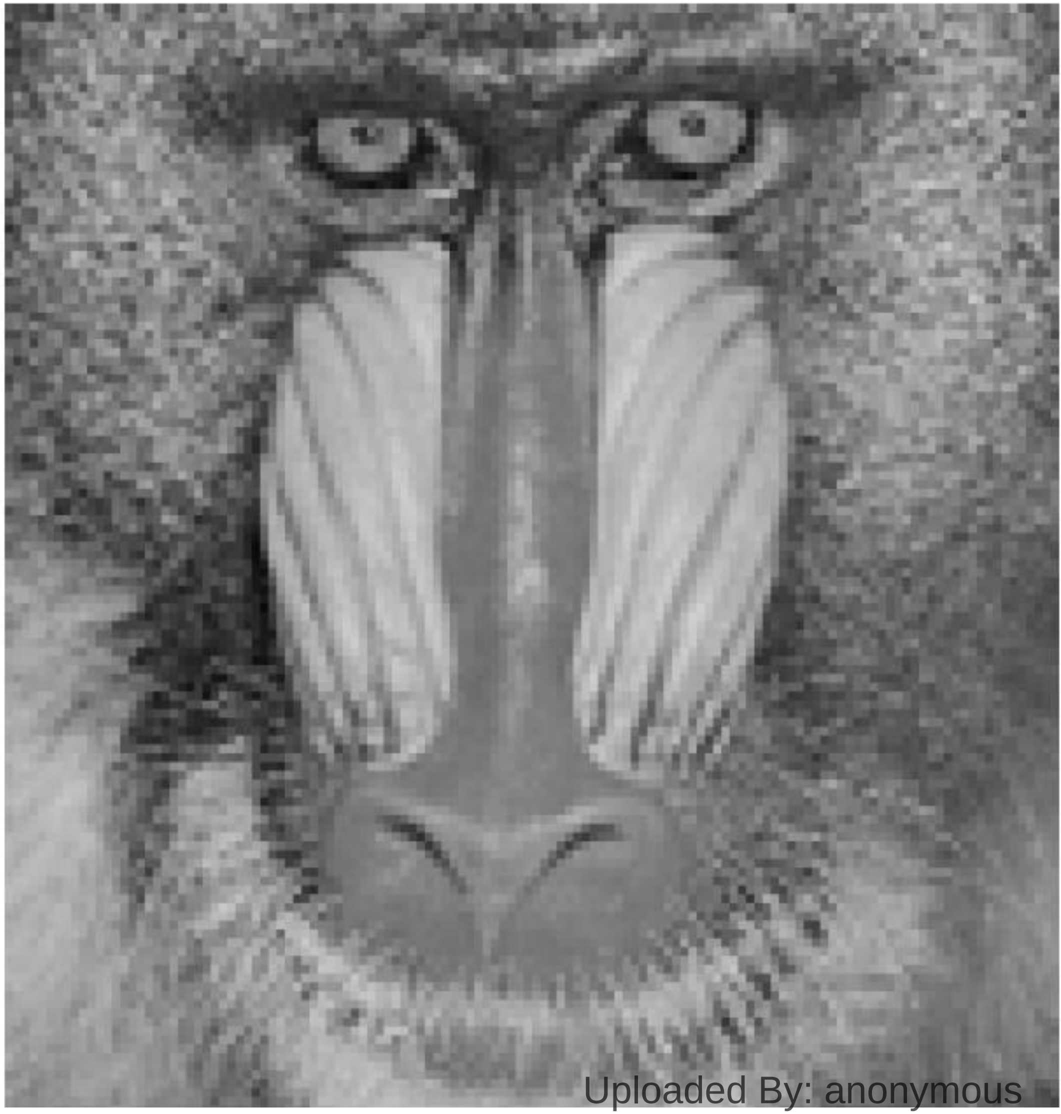
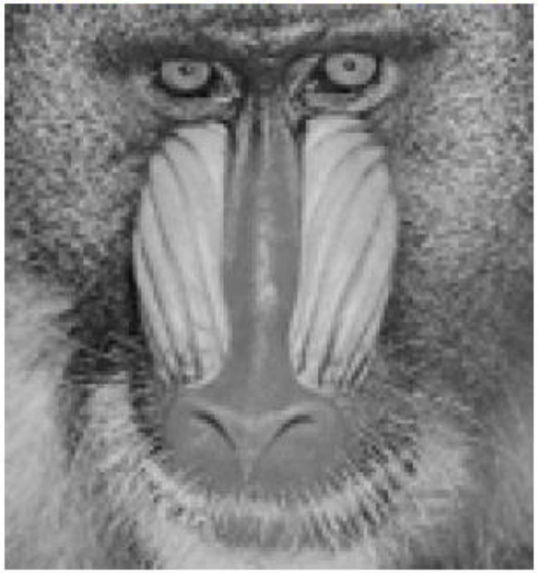


Image zoomed by factor of 2x2



Original

Image Interpolation

- We need to assign values to each pixel in the output image
- So scan through the output image; at each pixel calculate the value from the input image at the corresponding location
- If not at an integer position in the input image, we need to interpolate

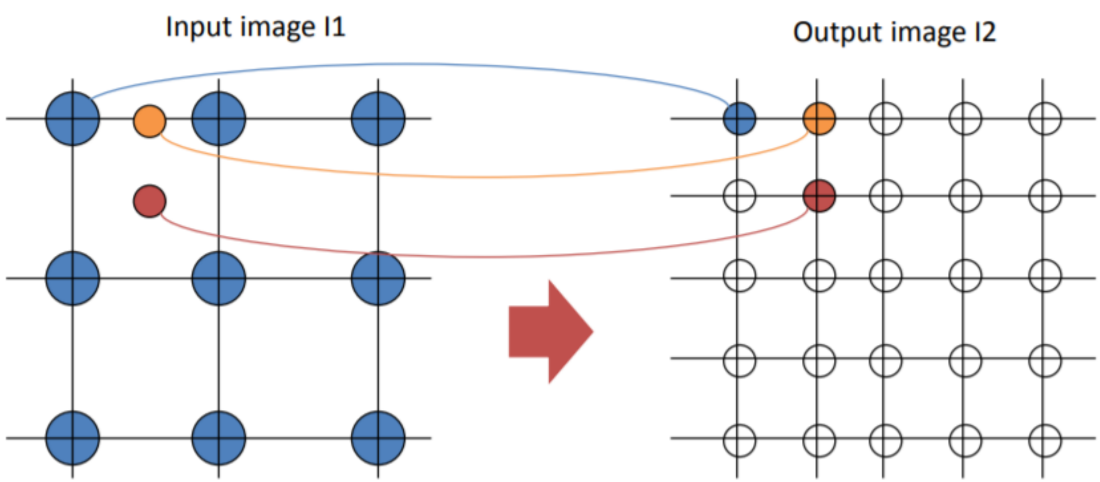
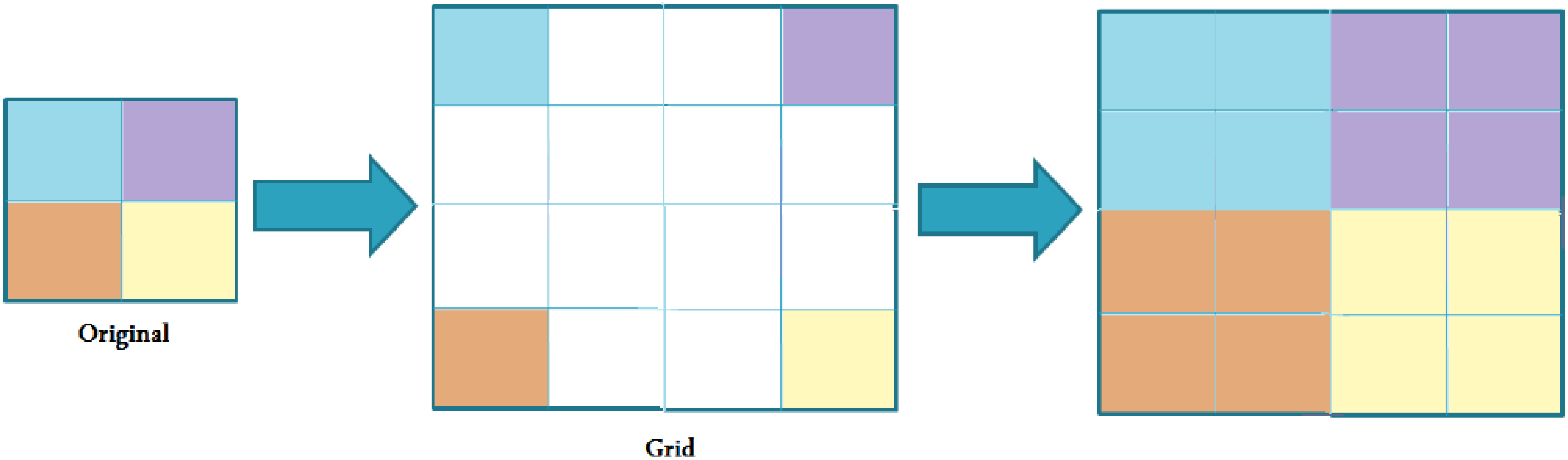


Image Interpolation - Nearest Neighbor

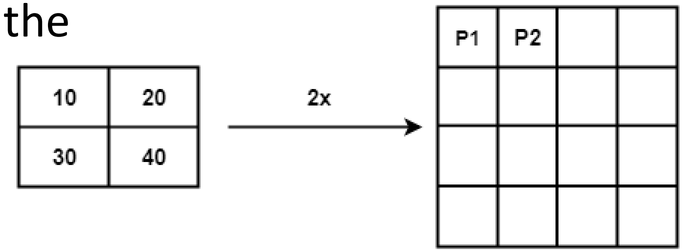
- Generate a grid with the new size and assign available pixels to new locations
- For missing pixels, assign to it the value of the closest known pixel
- Fast but produces artifacts



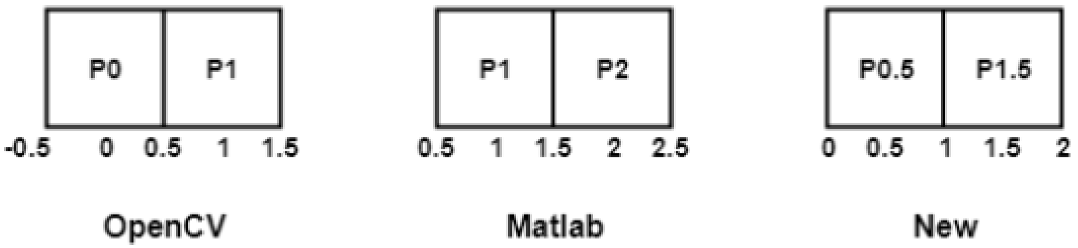
Nearest Neighbor Interpolation

- We assign the unknown pixel to the nearest known pixel.
  - Example: Suppose, we have a 2×2 image and let's say we want to upscale this by a factor of 2 as shown below.

    - ✧ Let's pick up the first pixel (denoted by 'P1') in the unknown image.
    - ✧ **Assumption: a pixel is always represented by its center value.**



    - ✧ Indexing in OpenCV starts from 0 while in matlab it starts from 1. But for the sake of simplicity, we will start indexing from 0.5 which means that our first pixel is at 0.5 next at 1.5 and so on as shown below.



10:(0.5,0.5), 20:(1.5,0.5), 30:(0.5,1.5), 40:(1.5,1.5).

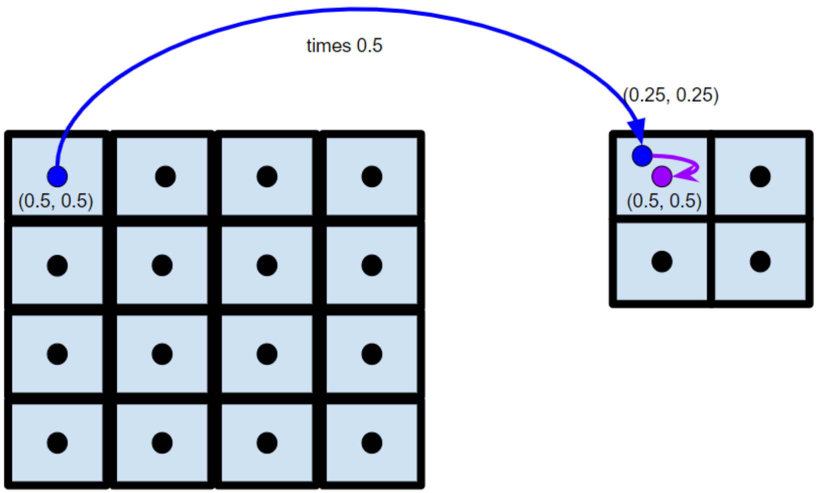
Nearest Neighbor Interpolation - Algorithm

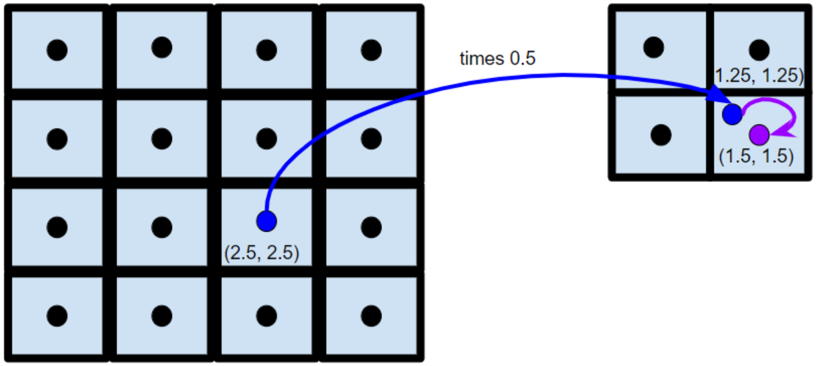
1. **Calculate Scaling Factors:**
   - Determine the scaling factors for both the horizontal and vertical dimensions. For example, if you are doubling the size of the image, the scaling factors would be 2 for both dimensions.
2. **Calculate New Image Dimensions:**
   - Calculate the dimensions of the new (resized) image by multiplying the original dimensions by the scaling factors.
3. **Iterate Over Each Pixel in the Resized Image:**
   - For each pixel in the resized image, calculate its corresponding position in the original image using the inverse of the scaling factors.
4. **Assign Pixel Value:**
   - Assign the value of the nearest pixel in the original image to the corresponding pixel in the resized image.

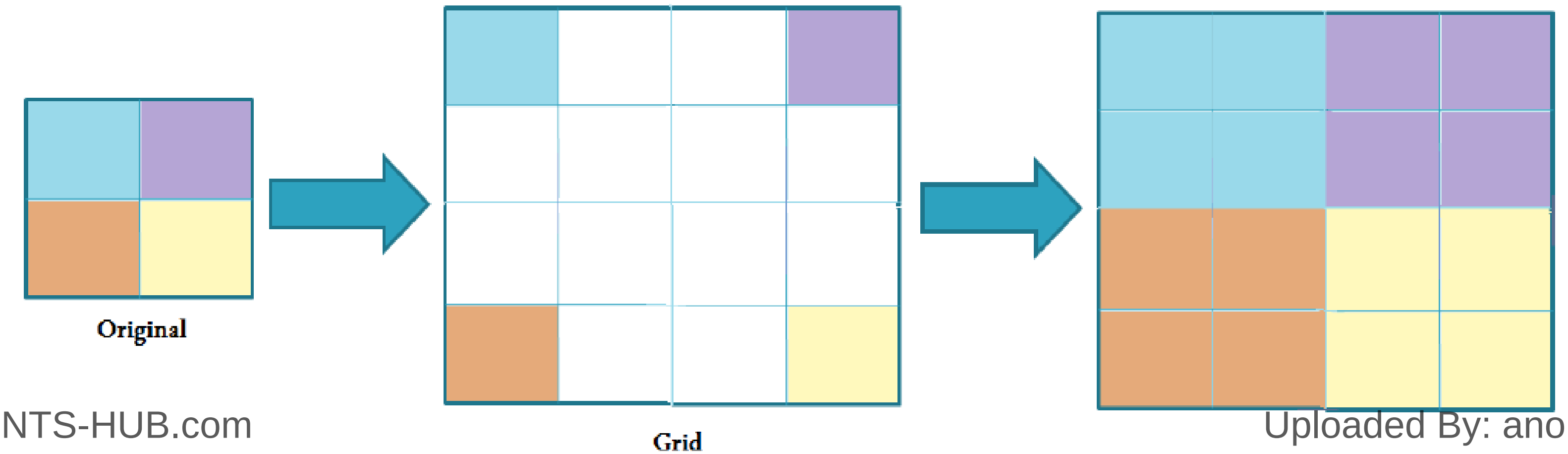
- **Example: Pixel P1**
  - P1 in the enlarged image has coordinates (0.5,0.5).
  - Our image has a scale ratio of 2/4 (the scale ratio is calculated by (in_dimension/out_dimension) in the x and y direction, so we'll divide P1's x and y values by 2, giving us (0.25,0.25).
  - Looking at our original image, (0.25,0.25) is closest to (0.5,0.5). Meaning P1 gets a value of 1.

Nearest Neighbor Interpolation

Assumption: a pixel is always represented by its center value.

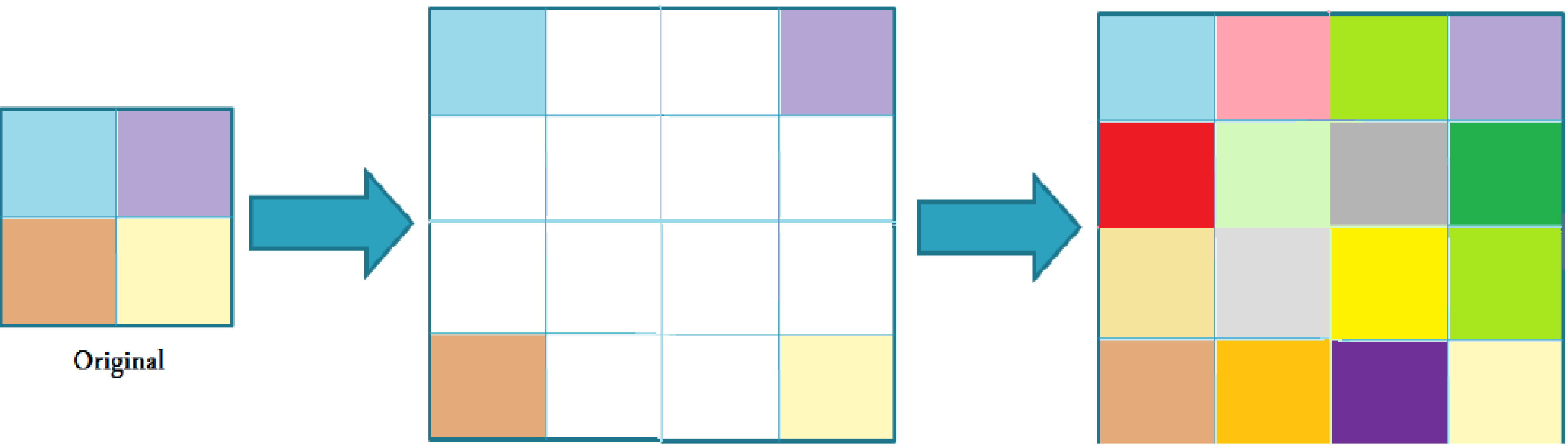






Bilinear Interpolation

- Bilinear image interpolation is a method for resizing or rotating an image while preserving its quality. It works by calculating the weighted average of the four nearest pixels to the desired output pixel.



Bilinear interpolation - Algorithm

1. **Calculate the scaling factors for the x and y directions.**
2. **Calculate Coordinates:**
   - Given a target location (x, y) in the resized image, calculate the corresponding coordinates (u, v) in the original image.
   - These coordinates are typically calculated using a scaling factor.

$$u = \frac{x}{\text{scaling factor}}$$
$$v = \frac{y}{\text{scaling factor}}$$

2. **Identify Nearest Pixels:**
   - Identify the four nearest pixels (p1, p2, p3, p4) surrounding the coordinates (u, v) in the original image.
     1. Find the nearest integer values to the x and y coordinates.
     2. The four nearest pixels in the input image are the pixels with the following coordinates: (u_nearest, v_nearest), (u_nearest + 1, v_nearest), (u_nearest, v_nearest + 1), (u_nearest + 1, v_nearest + 1).
3. **Calculate Distances:**
   - Calculate the distances (d1, d2, d3, d4) (using for example Euclidean distance) between the target location and each of the nearest pixels.
   - These distances are typically expressed as a fraction of the distance between adjacent pixels.

Bilinear interpolation - Algorithm

5. **Calculate Weights:**
   1. Calculate the weights (w1, w2, w3, w4) based on the distances.
   2. The weights are usually proportional to the inverse of the distances. The idea is to assign higher weights to pixels that are closer to the target location and lower weights to pixels that are farther away. This reflects the concept that closer pixels should contribute more to the interpolated value.
6. **Calculate Weighted Average:**
   1. For each color channel (e.g., red, green, blue), calculate the weighted average using the formula:

$$Pixel\_value = w1 \times p1 + w2 \times p2 + w3 \times p3 + w4 \times p4$$

7. **Repeat for Each Channel:**
   1. Repeat the weighted average calculation for each color channel to obtain the final color value for the target pixel.

Image Interpolation

- Generate a grid with the new size and assign available pixels to new locations
- For missing pixels, assign to it the value of by

$$v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$$

- The constants are determined by solving sixteen equations for the closest sixteen neighbors

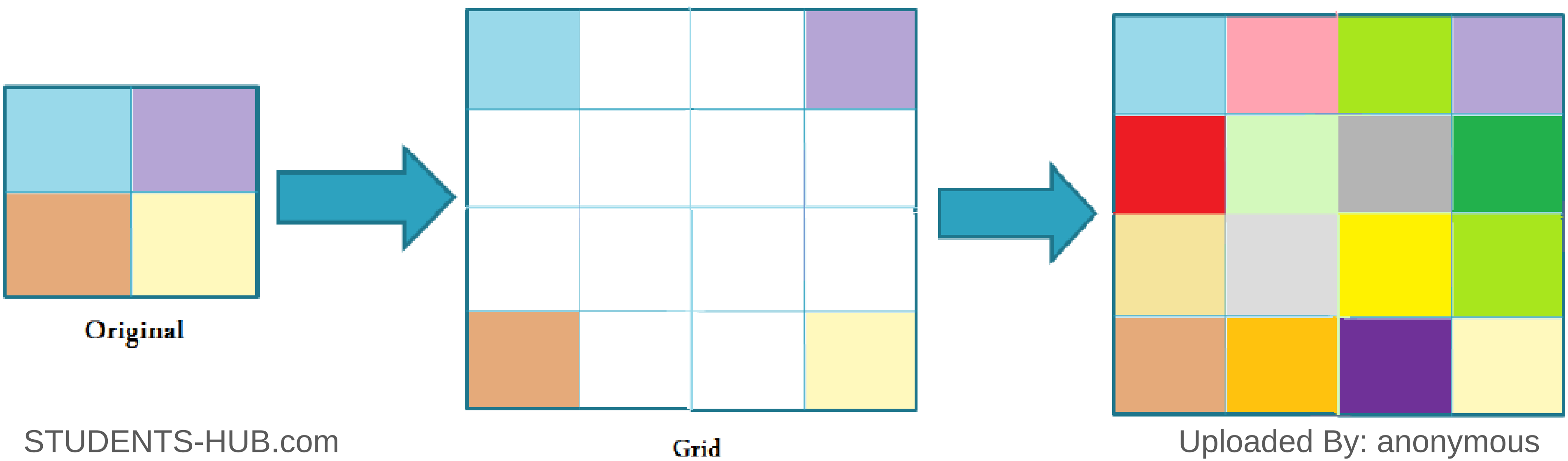


Image Interpolation

a b c

FIGURE 2.27 (a) Image reduced to 72 dpi and zoomed back to its original 930 dpi using nearest neighbor interpolation. This figure is the same as Fig. 2.23(d). (b) Image reduced to 72 dpi and zoomed using bilinear interpolation. (c) Same as (b) but using bicubic interpolation.

Example

Nearest Neighbor

Bilinear

Bicubic

128 → 1024

64 → 1024

Interpolation Summary

- Image scaling can be performed in real time and the global visual effects can be preserved when interpolation methods are employed.
- However, these interpolation scaling methods can bring artifacts, such as an artificial block and aliasing.
- Scaling causes obvious distortion if the aspect ratio of the input image is obviously different from that of the output image.

Acknowledgement

- The material in these slides are based on:
  - Digital Image Processing: Rafael C. Gonzalez, and Richard
  - Forsythe and Ponce: Computer Vision: A Modern Approach
  - Rick Szeliski's book: Computer Vision: Algorithms and Applications
  - cs131@ Stanford University
  - cs131n@ Stanford University
  - CS198-126@ University of California, Berkely
  - CAP5415@ University of Central Florida
  - CSW182 @ University of California, Berkely
  - 11-785@ Carnegie Mellon University
  - CSCI1430@ Brown University
  - Computer Vision@ Bonn University
  - ICS 505@ KFUPM
  - Digital Image Processing@ University of Jordan