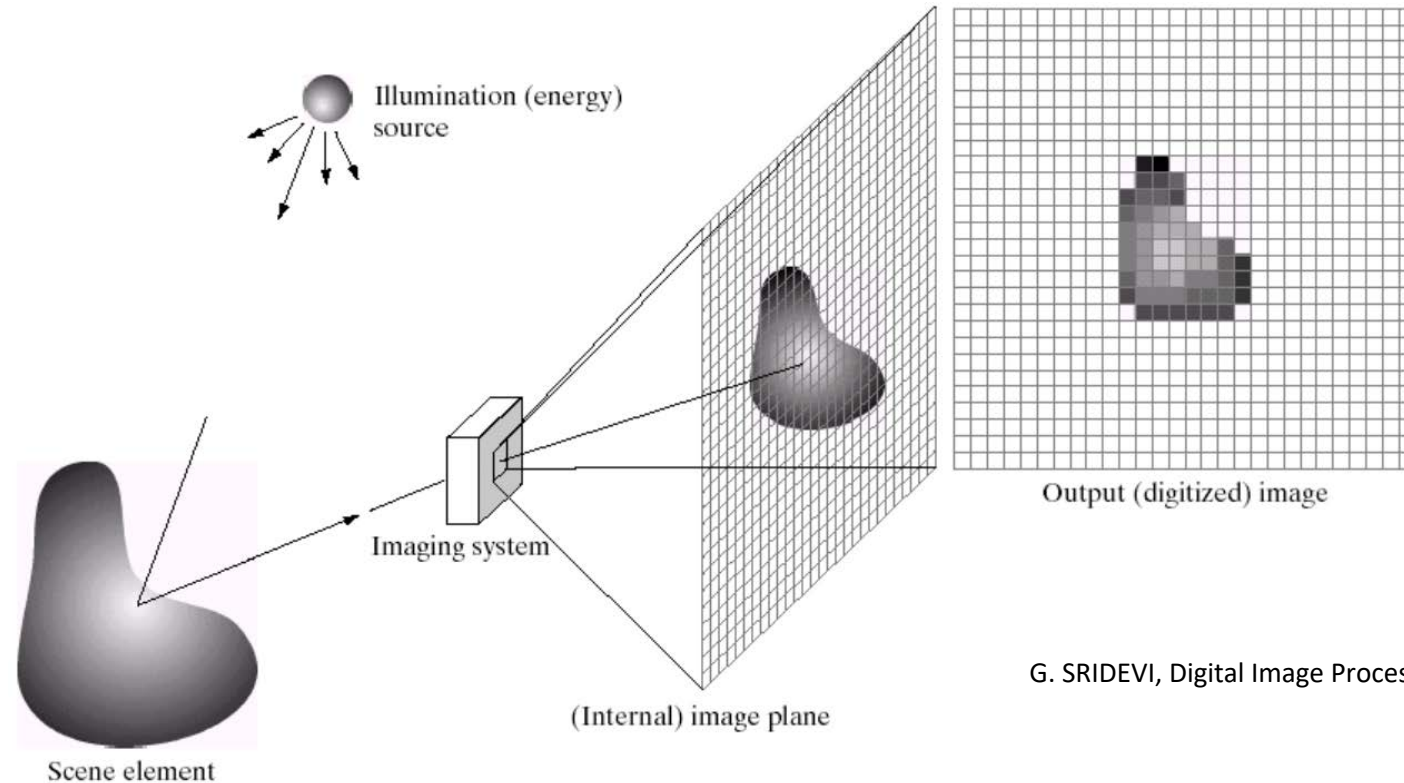


Chapter 3: Image Representation

Faculty of Digital Engineering
Institute of Digital Technology
2026

What is a Digital Image?

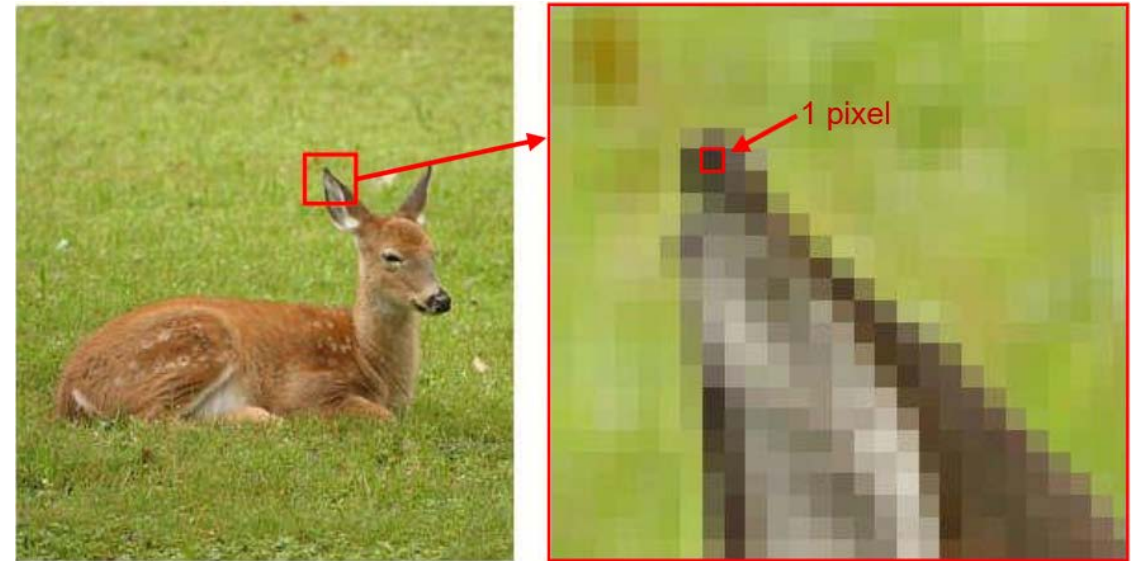
- a representation of a two-dimensional signal as a finite set of digital values, called picture elements or **pixels**



G. SRIDEVI, Digital Image Processing

What is a Digital Image?

- An image is a two-dimensional function $f(x, y)$,
 - where x and y are **spatial coordinates**
 - the amplitude of f at any pair of coordinates (x, y) is called the **intensity** of the image at that point
- Digital image: x , y , and the intensity values of f are usually all finite, discrete quantities.



Note: digitization implies that a digital image is an approximation of a real scene

Image Coordinate System

- The coordinate system is usually flipped in the vertical direction
 - the y-coordinate runs from top to bottom and
 - the origin lies in the upper left corner

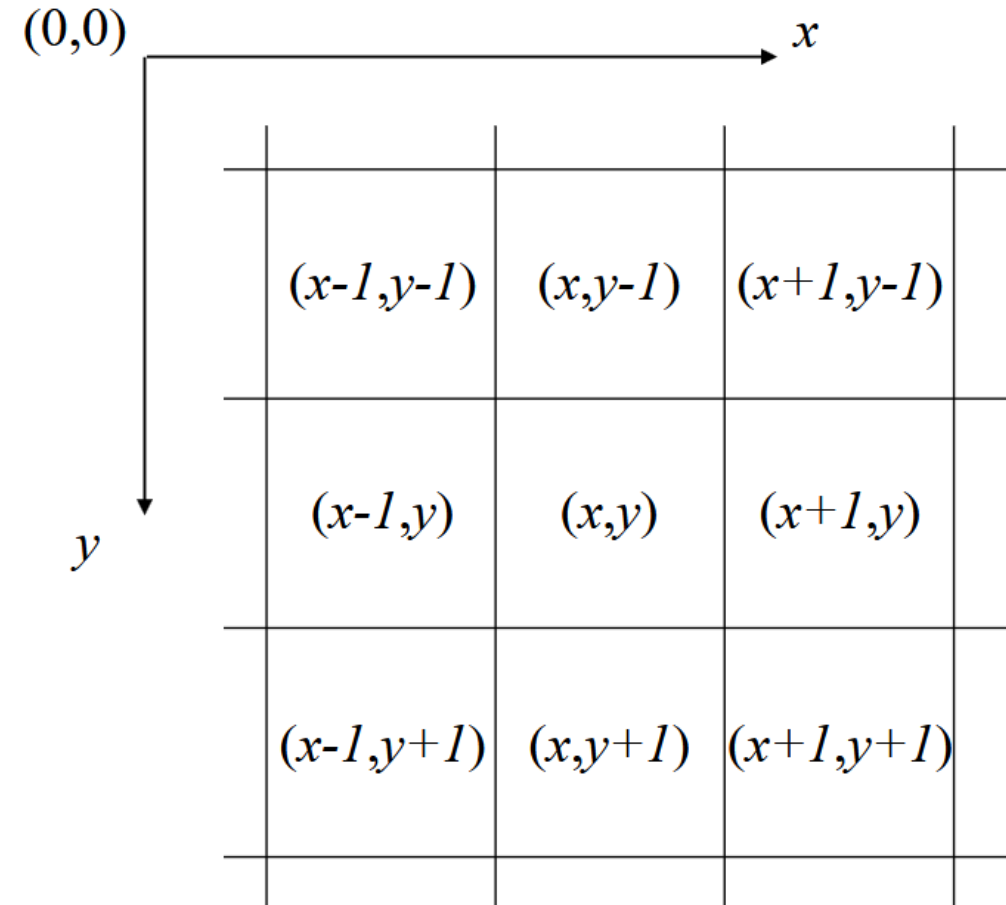


Image Types

- The information within a pixel depends on the data type used to represent it.
- Pixel values are practically always binary words of length b so that a pixel can represent any of 2^b different values.
- The value b is called the bit depth (or just “depth”) of the image.
 - The exact bit-level layout of an individual pixel depends on the kind of image
 - binary
 - grayscale
 - RGB color.



Grayscale Images

- Grayscale images are referred to as monochrome, or one-color image.
 - consist of a single channel that represents the intensity, brightness, or density of the image. In most cases,
- A typical grayscale image uses $b = 8 \text{ bits}$ (1 byte) per pixel
 - intensity values in the range $0, \dots, 255$
 - 0 represents the minimum brightness (black)
 - 255 the maximum brightness (white)

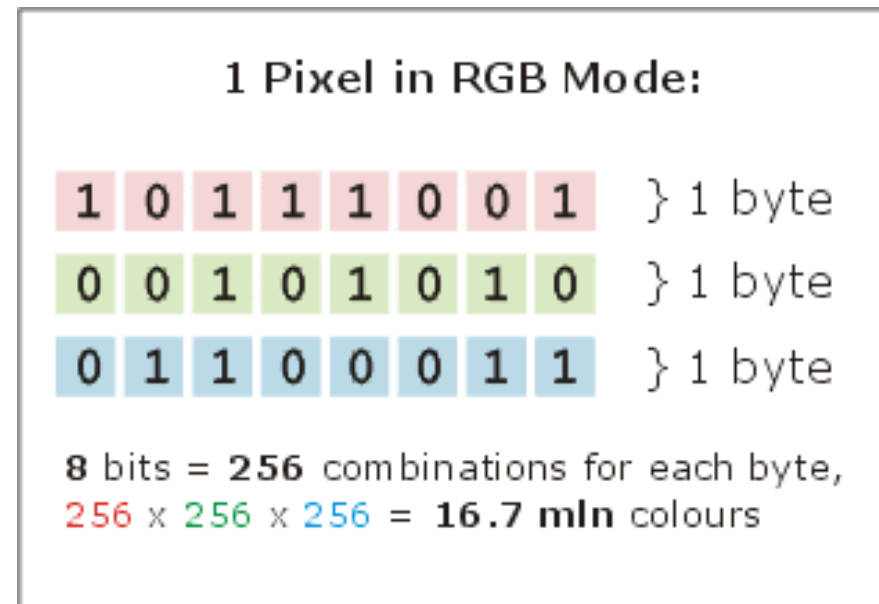
Binary Images

- A binary image is referred to as a 1 bit/pixel image
 - pixels can only take on one of two values, black or white (0 or 1)
- Binary images are often created from grayscale images via a threshold value
 - White if above threshold or black otherwise

$$\begin{bmatrix} 122 & 128 & 145 & 133 & 132 \\ 129 & 134 & 144 & 139 & 140 \\ 120 & 122 & 139 & 136 & 132 \\ 119 & 123 & 122 & 131 & 137 \end{bmatrix} \xrightarrow{\text{Threshold} = 134} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Color Images

- Color image can be modeled as three band monochrome image data,
 - based on the primary colors red, green, and blue RGB
 - 8 bits for each color component



Important Concepts

- Image size
 - Described by horizontal (Width) and vertical (Height) pixel count
 - Total pixel count = Width x Height
- Color Depth
 - Each pixel of the image contains unique color information described in the unit of bits.
 - b = number of bits.
 - 2^b = number of possible displayed colors
- Raw file size: ***Width x Height x Depth***

Image Formats

- What Are Image Formats?
 - File types used to store digital images, balancing quality, size, and compatibility.
- Common Image Formats:
 - JPEG (JPG): Great for photographs, lossy compression reduces file size.
 - PNG: Supports transparency, lossless compression, suitable for graphics.
 - GIF: Used for animations, limited to 256 colors, lossless for simple images.
 - BMP: Uncompressed, high-quality but large file sizes.
 - TIFF: High-quality, supports lossless/lossy options, common in professional use.
 - WEBP: Modern format, combines transparency and compression efficiency.



Image Compression

- What Is Image Compression?
 - Reducing file size by removing unnecessary data while maintaining visual quality.
- Types of Compression:
 - Lossless Compression (e.g., PNG, GIF, TIFF): No data lost; original quality preserved.
 - Lossy Compression (e.g., JPEG, WEBP): Reduces quality by removing details imperceptible to the human eye.
- Why Compress Images?
 - Faster loading times (web use).
 - Saves storage space.
 - Enhances performance without noticeable quality loss (lossy).

Exercise

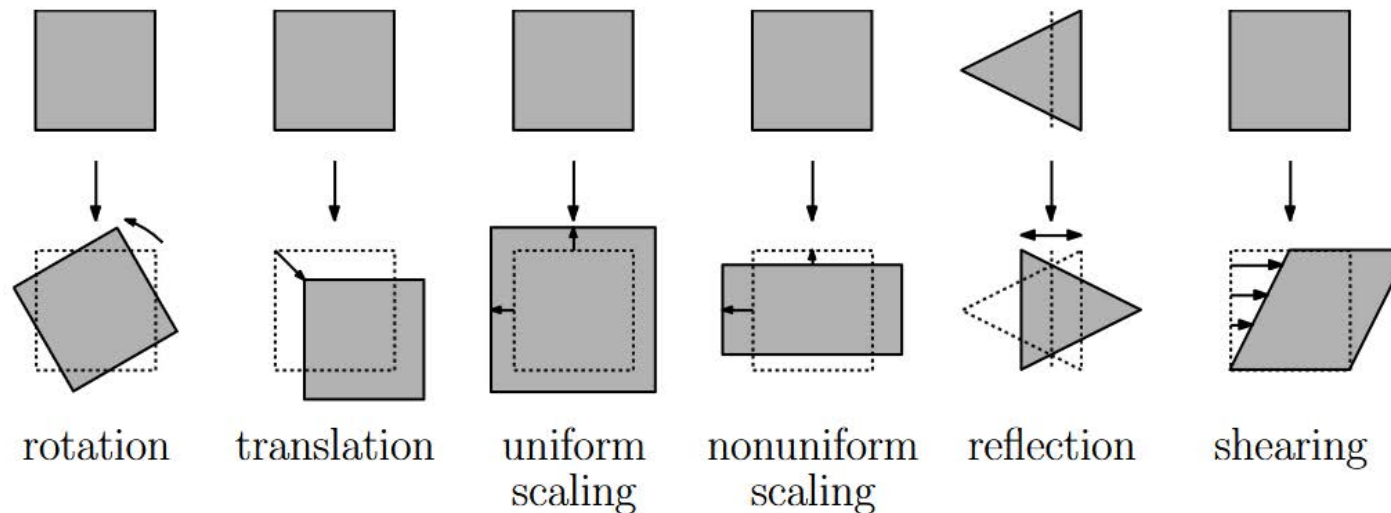
- A PC plays back video data recorded at 500 x 400 pixels per frame, 30 frames per second, and 3 bytes per pixel while reading it from a peripheral device. When the data transfer rate between the peripheral device and the PC is 1×10^6 bytes per second, in order to play back the video without delay, the data must be compressed to one n -th ($1/n$) in advance.
- What is the minimum value of n ? Here, the time to expand the compressed data and other impacts can be ignored.

Solution

- Frame: 500 x 400 pixels
- 1 second: 30 frames
- 1 pixel: 3 bytes
- Transfer rate: 1×10^6 bytes/s
- Compress rate = $1/n$
- Video size for 1s: $30 \times (500 \times 400) \times 3 = 18 \times 10^6$ bytes
- In order to play back the video without delay: *transmit rate* $\leq$ *receive rate*
- Compress the data at least 18 times

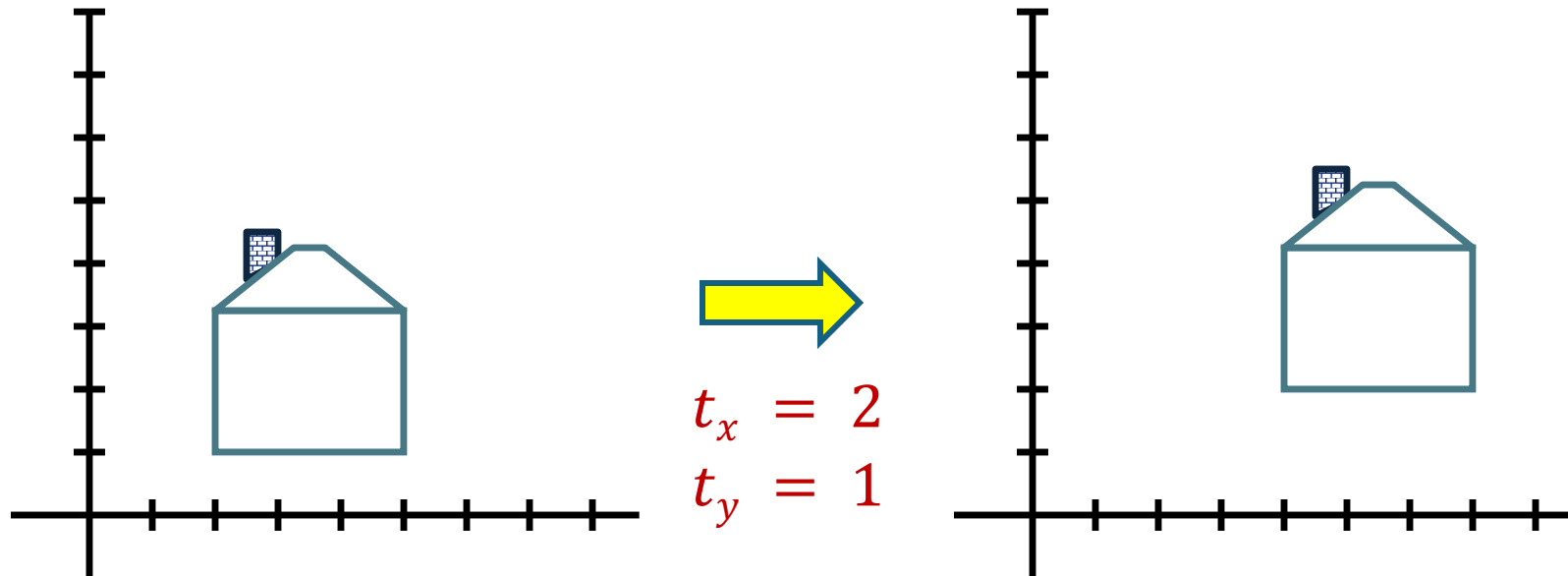
Image Transformation

- What is image transformation?
 - The process of altering an image's geometry while preserving its essential features.
- **Affine Transformation:** A linear mapping that preserves points, straight lines, and parallelism.



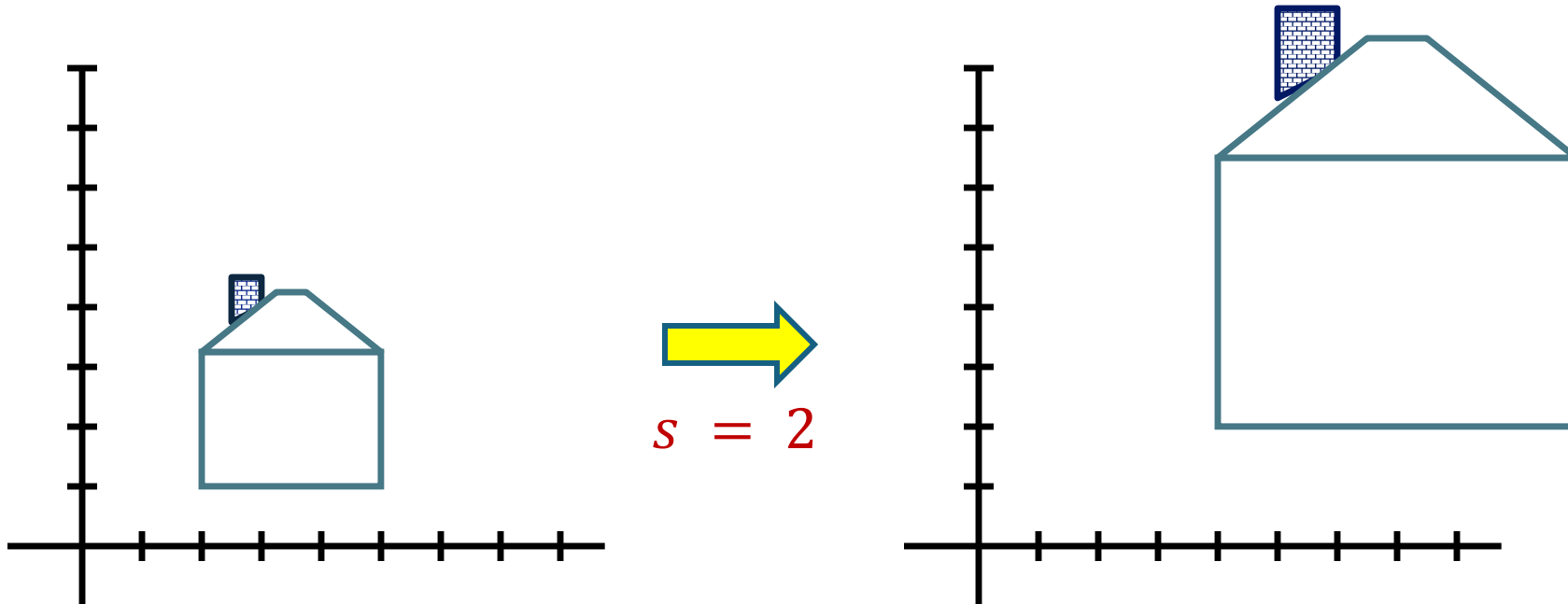
Translation

- $x' = x + t_x$
 $y' = y + t_y$



Uniform scaling

$$\begin{aligned}x' &= s * x \\ y' &= s * y\end{aligned}$$

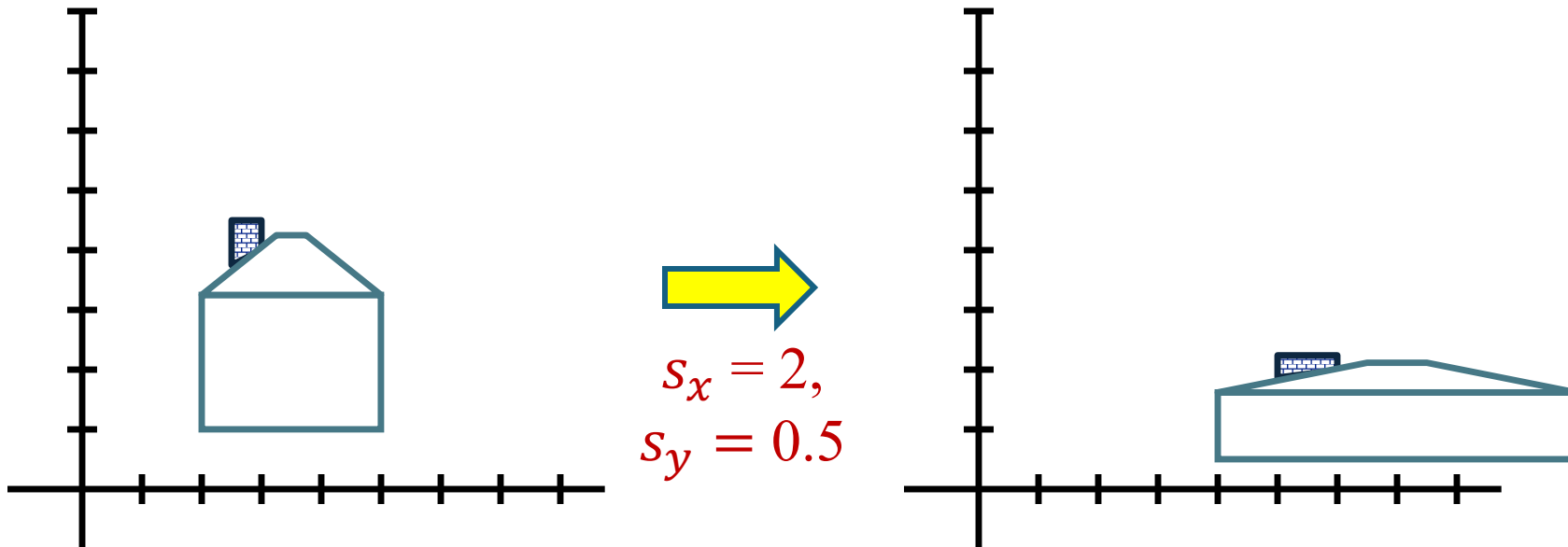


Non-uniform scaling

$$\begin{aligned}x' &= s_x * x \\ y' &= s_y * y\end{aligned}$$

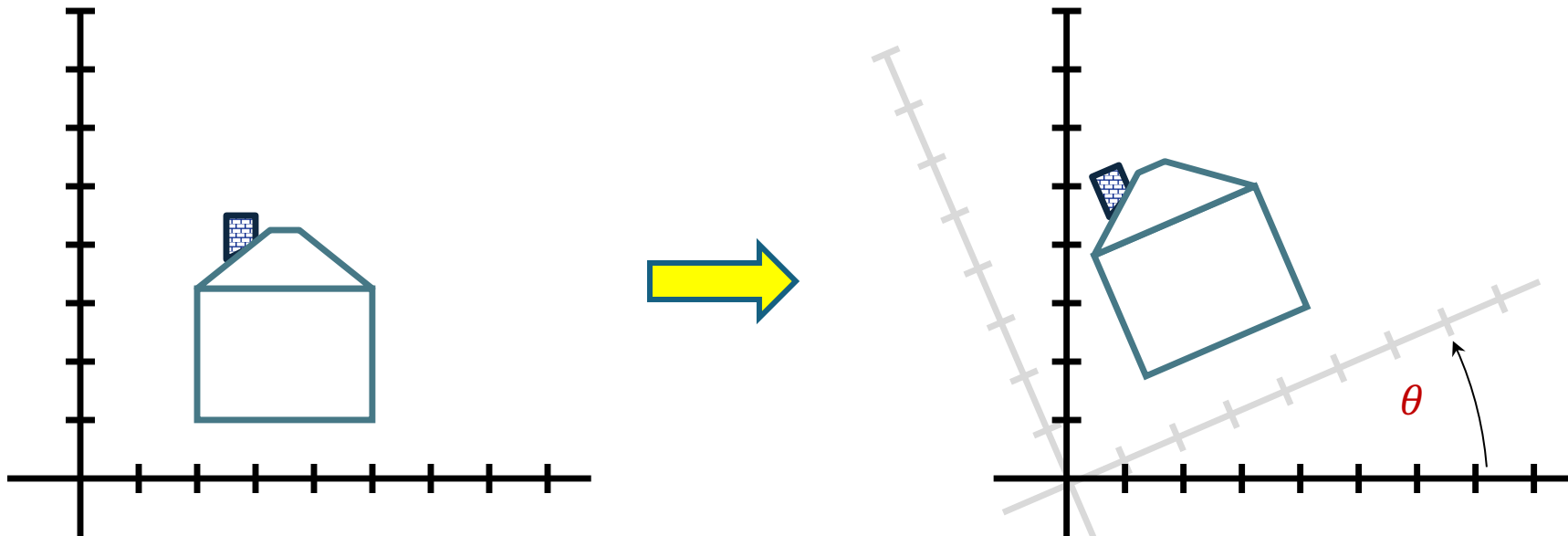
In matrix form:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



Rotation

- Rotate the image by an angle of θ about the origin:

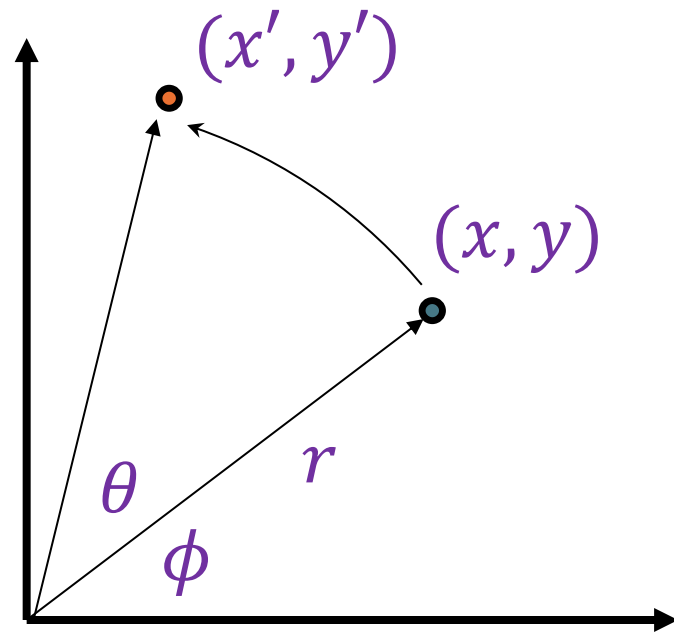


Rotation

- Rotate the image by an angle of θ about the origin:

$$x' = r \cos(\phi + \theta) = r \cos(\phi) \cos(\theta) - r \sin(\phi) \sin(\theta) = x \cos(\theta) - y \sin(\theta)$$

$$y' = r \sin(\phi + \theta) = r \sin(\phi) \cos(\theta) + r \cos(\phi) \sin(\theta) = x \sin(\theta) + y \cos(\theta)$$



Apply trig identities

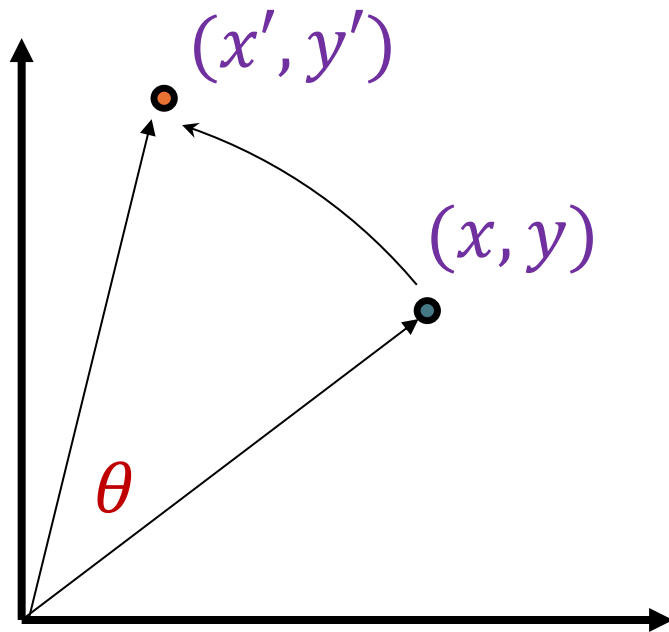
$$\begin{aligned} x &= r \cos(\phi) \\ y &= r \sin(\phi) \end{aligned}$$

Substitute

$$\begin{aligned} \cos(a+b) &= \cos(a)\cos(b) - \sin(a)\sin(b) \\ \sin(a+b) &= \sin(a)\cos(b) + \cos(a)\sin(b) \end{aligned}$$

Rotation

- Rotate the image by an angle of θ about the origin:



$$x' = x \cos(\theta) - y \sin(\theta)$$

$$y' = x \sin(\theta) + y \cos(\theta)$$

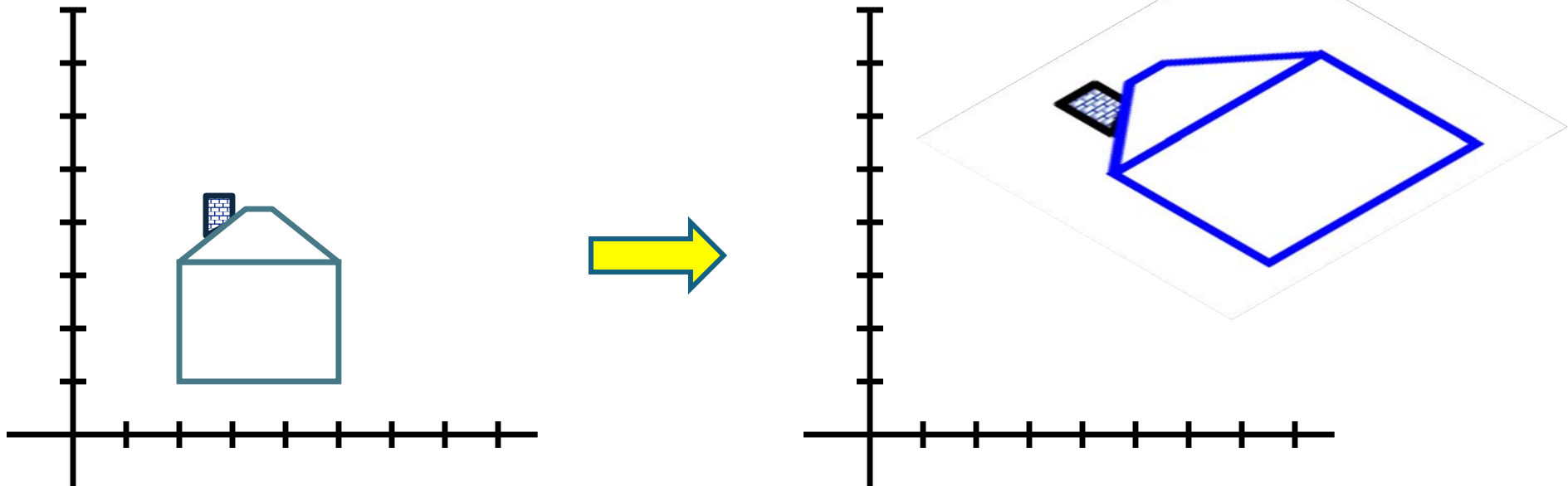
In matrix form:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Affine transformation

- Combination of translation, non-uniform scaling, rotation, and *shear*

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$



Types of Affine Transformations

- **Translation:** Shifts the image in X or Y direction by (tx, ty)
 - Example: Moving an object from one part of the image to another.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix}$$

- **Scaling:** Changes the size of the image by factors sx and sy .
 - Example: Enlarging or shrinking an image.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} sx & 0 \\ 0 & sy \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Types of Affine Transformations

- **Rotation:** Rotates the image around a specified point (often the center) by angle θ .

- Example: Turning an image clockwise or counterclockwise.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **Shearing:** Slants the image along X or Y axis by factors sh_x or sh_y .

- Example: Creates a "tilted" effect.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Example - Scaling

- Given a 3x3 grayscale image I and scaling transformation matrix S where $S_x = 2$, $S_y = 2$. Find the new pixel positions and resized image.

$$I = \begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix}$$

$$M = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

Example - Scaling

1. Scaling Transformation Formula:

- $x' = S_x \cdot x$, $y' = S_y \cdot y$

2. Calculate New Positions:

- $(0, 0) \rightarrow (0, 0)$; $(0, 1) \rightarrow (0, 2)$; $(0, 2) \rightarrow (0, 4)$
- $(1, 0) \rightarrow (2, 0)$; $(1, 1) \rightarrow (2, 2)$; $(1, 2) \rightarrow (2, 4)$
- $(2, 0) \rightarrow (4, 0)$; $(2, 1) \rightarrow (4, 2)$; $(2, 2) \rightarrow (4, 4)$

3. Construct the Scaled Image

10	10	20	20	30	30
10	10	20	20	30	30
40	40	50	50	60	60
40	40	50	50	60	60
70	70	80	80	90	90
70	70	80	80	90	90

Example - Translation

- A pixel is located at $(x, y) = (5, 7)$. The image undergoes a translation transformation where $t_x = 3$, $t_y = 2$. Find the new coordinates of the pixel after translation.

- $x' = x + t_x ; y' = y + t_y \rightarrow x' = 8 ; y' = 9$

Example - Rotation

- A pixel is located at $(x, y) = (4, 2)$. Rotate it by **90° counterclockwise** about the origin using the **rotation matrix**, where $\theta=90^\circ$. Find the new coordinates (x', y') after rotation.

$$R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

Example - Rotation

1. Scaling Transformation Formula:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

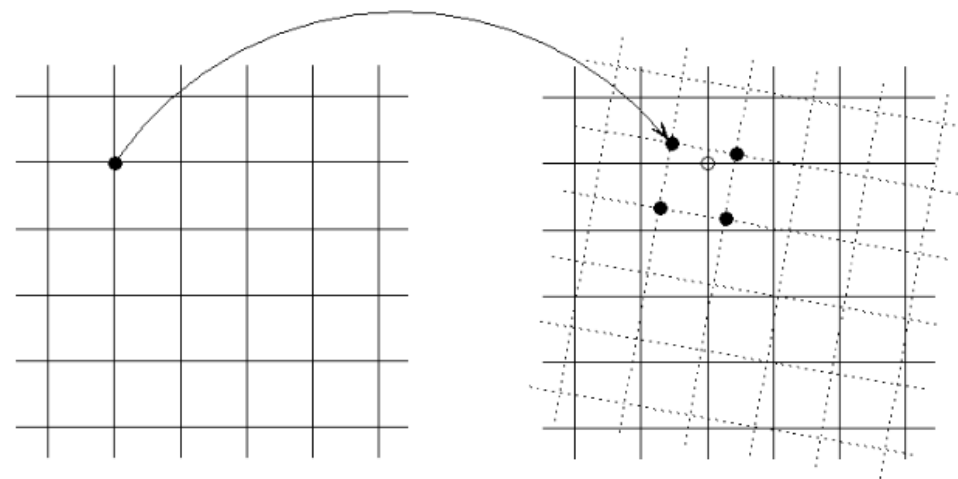
2. $x' = x \cos(\theta) - y \sin(\theta); y' = x \sin(\theta) + y \cos(\theta)$

$$x' = (4)(0) - (2)(1) = -2$$

$$y' = (4)(1) + (2)(0) = 4$$

Interpolation in Image Transformation

- Image Interpolation refers to the process of resizing images by estimating pixel intensities at interpolated points using different algorithms.
- Why Interpolation?
 - Transformations can result in non-integer pixel positions.
 - Interpolation estimates the pixel value for these positions.



Types of Interpolation

- **Nearest Neighbor:** selects the closest pixel value.

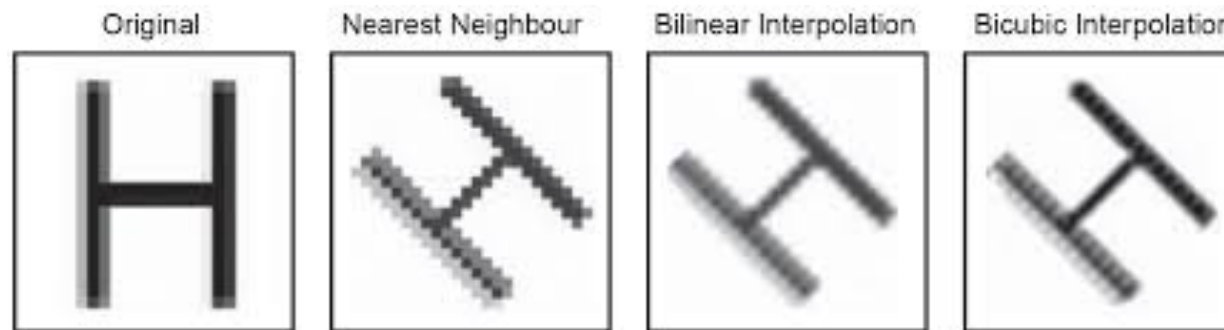
$$I(x', y') = I(\text{round}(x), \text{round}(y))$$

- **Bilinear:** Considers the 4 nearest pixels and computes a weighted average.

$$I(x', y') = (1 - dx)(1 - dy)I(x_1, y_1) + \dots$$

where dx, dy are distances from the nearest pixel.

- **Bicubic:** Uses 16 nearest pixels for smooth results.



Example - Nearest Neighbor

- Given a 3x3 grayscale image. A pixel is sampled at **(2.6, 1.3)** using **nearest neighbor interpolation**. Find the estimated pixel value.

$$I = \begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix}$$

Example - Nearest Neighbor

- Given a 3x3 grayscale image. A pixel is sampled at **(2.6, 1.3)** using **nearest neighbor interpolation**. Find the estimated pixel value.

$$I = \begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix}$$

- $x' = \text{round}(x), y' = \text{round}(y) \rightarrow (x', y') = (3, 1)$
- The estimated pixel value is 30

Example – Bilinear Interpolation

- Given a pixel located at non-integer coordinates (2.5, 3.5) in an image.
- The four neighbor pixel values are:
 - 100 (2, 3), 120 (3, 3), 90 (2, 4), 110 (3, 4)
- Perform bilinear interpolation using the formula.

$$I(x, y) = I_{00}(1 - dx)(1 - dy) + I_{10}dx(1 - dy) + I_{01}(1 - dx)dy + I_{11}dxdy$$

where:

- $dx = x - x_0, dy = y - y_0.$

Example – Bilinear Interpolation

- $I_{00} = 100$ at (2,3).
- $I_{10} = 120$ at (3,3).
- $I_{01} = 90$ at (2,4).
- $I_{11} = 110$ at (3,4).
- $dx = 2.5 - 2 = 0.5$.
- $dy = 3.5 - 3 = 0.5$.

$$I(2.5, 3.5) = 100(1 - 0.5)(1 - 0.5) + 120(0.5)(1 - 0.5) + 90(1 - 0.5)(0.5) + 110(0.5)(0.5)$$

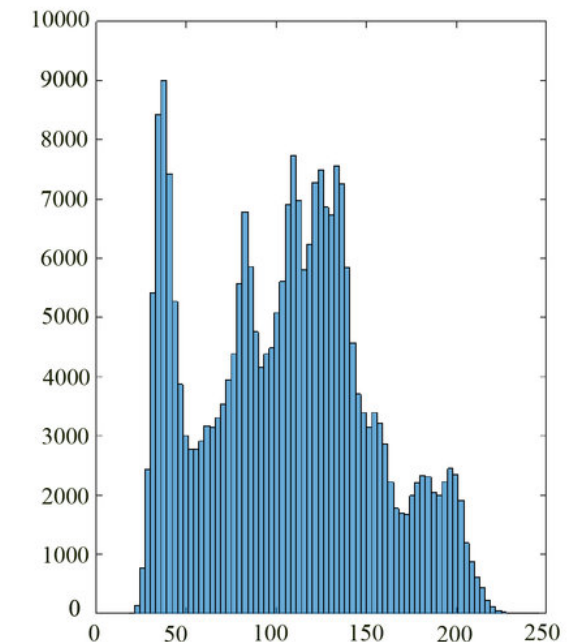
$$I(2.5, 3.5) = 100(0.5)(0.5) + 120(0.5)(0.5) + 90(0.5)(0.5) + 110(0.5)(0.5)$$

$$I(2.5, 3.5) = 25 + 30 + 22.5 + 27.5 = 105$$

Image Histogram

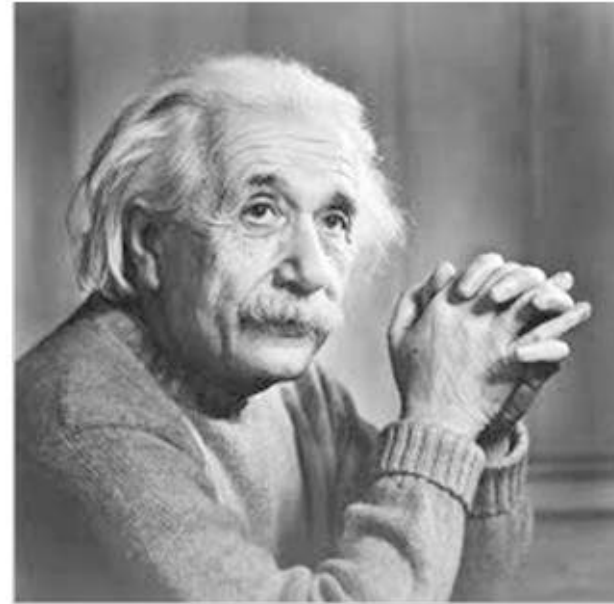
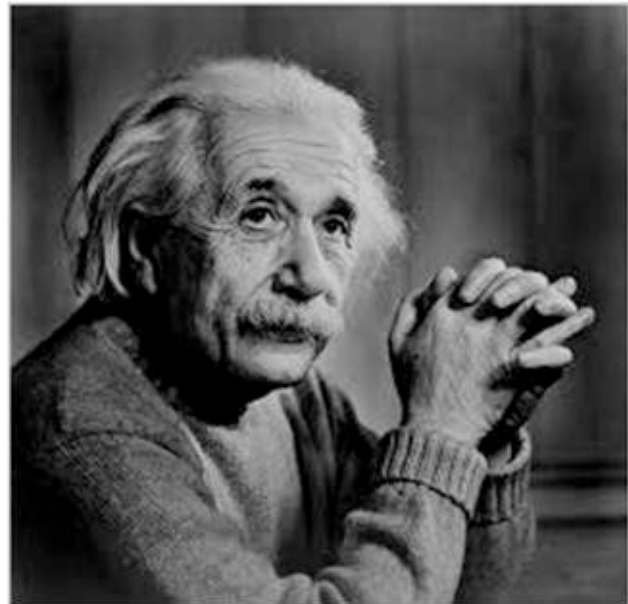
- A graphical representation of the distribution of pixel intensity values in an image.
 - X-axis: Pixel intensity (0–255 for grayscale images).
 - Y-axis: Number of pixels with each intensity value.
- Histogram for grayscale images:
 - Count the number of pixels at each intensity level:

$$H(i) = \text{Number of pixels with intensity } i$$
- Color Histogram
 - A separate histogram is calculated for each color channel RGB



Brightness and Contrast

- Brightness is a measure of an image's overall lightness or darkness.
- Contrast is the difference in brightness between the lightest and darkest parts of an image.



Brightness and Contrast

- Brightness: Controls how light or dark an image appears.

- Achieved by adding a constant b to all pixel values:

$$I'(x, y) = I(x, y) + b$$

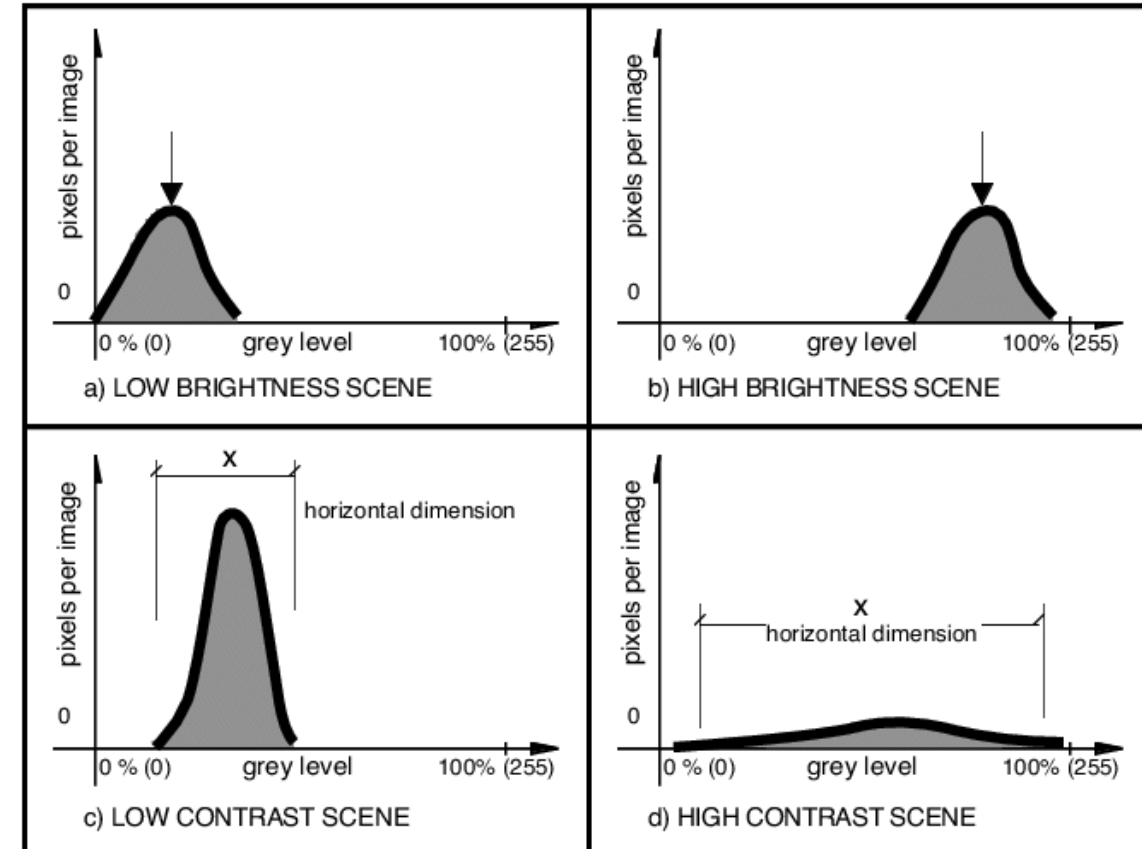
- Shifts the histogram to the right (increased brightness) or left (decreased brightness).

- Contrast: Controls the difference between light and dark areas.

- Achieved by multiplying pixel values by a constant α :

$$I'(x, y) = \alpha \cdot I(x, y)$$

- Stretches or compresses the histogram depending on the scaling factor α .



Demers, Claude. QUALITIES OF LIGHT AND SPACE: contrast as a global integrator.

