

# *Computer Vision*

## Course 2

## **Fundamental Steps in Image Processing/Computer Vision**

- methods whose input and output are images
- methods whose inputs are images but whose outputs are attributes extracted from those images

# *Computer Vision*

---

## **Course 2**

---

### **Outputs are images**

- image acquisition
- image filtering and enhancement
- image restoration
- color image processing
- wavelets and multiresolution processing
- compression
- morphological processing

# *Computer Vision*

---

## **Course 2**

---

### **Outputs are attributes**

- morphological processing
- segmentation
- representation and description
- object recognition

**Image acquisition** - may involve preprocessing such as scaling

**Image enhancement**

- manipulating an image so that the result is more suitable than the original for a specific operation
- enhancement is problem oriented
- there is no general ,theory' of image enhancement
- enhancement use subjective methods for image improvement
- enhancement is based on human subjective preferences regarding what is a „good” enhancement result

### **Image restoration**

- improving the appearance of an image
- restoration is objective - the techniques for restoration are based on mathematical or probabilistic models of image degradation

### **Color image processing**

- fundamental concept in color models
- basic color processing in a digital domain

**Wavelets and multiresolution processing**

- representing images in various degree of resolution

**Compression**

- reducing the storage required to save an image or the bandwidth required to transmit it

**Morphological processing**

- tools for extracting image components that are useful in the representation and description of shape
- a transition from processes that output images to processes that output image attributes

**Segmentation**

- partitioning an image into its constituents' parts or objects
- autonomous segmentation is one of the most difficult tasks of DIP
- the more accurate the segmentation, the more likely recognition is to succeed

**Representation and description** (almost always follow segmentation)

- segmentation produces either the boundary of a region or all the points in the region itself
- converting the data produced by segmentation to a form suitable for computer processing

# *Computer Vision*

---

## **Course 2**

---

- boundary representation: the focus is on external shape characteristics such as corners or inflections
- complete region: the focus is on internal properties such as texture or skeletal shape
- description is also called feature extraction – extracting attributes that result in some quantitative information of interest or are basic for differentiating one class of objects from another

### **Object recognition** (Machine Learning techniques)

- the process of assigning a label (e.g. „vehicle”) to an object based on its descriptors

### **Knowledge database**

# Computer Vision

## Course 2

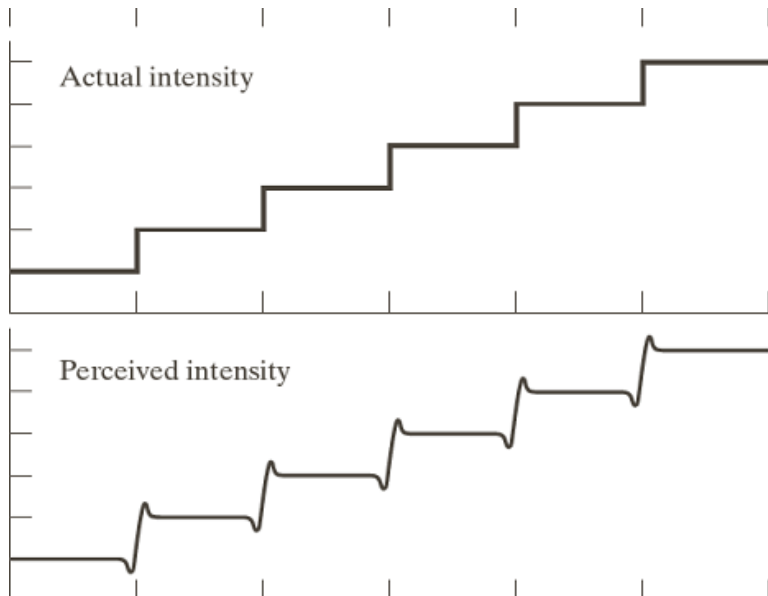
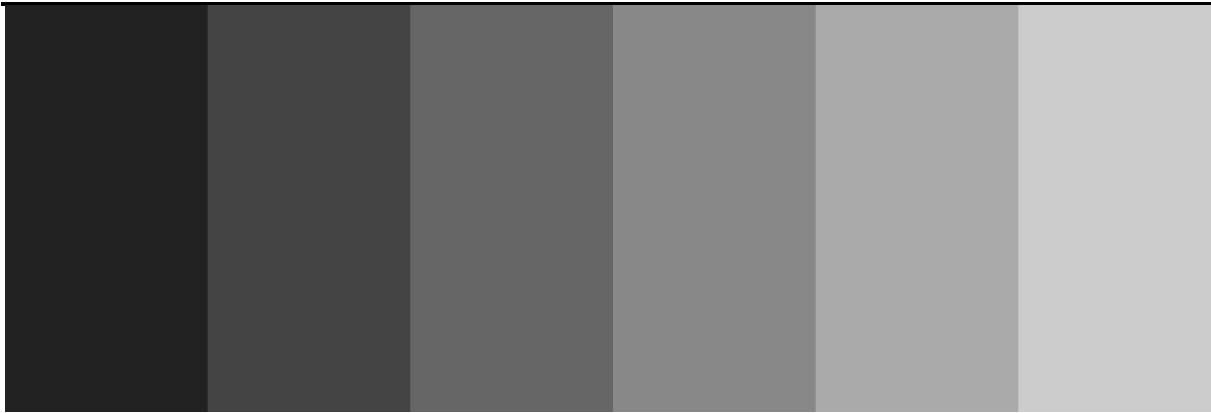


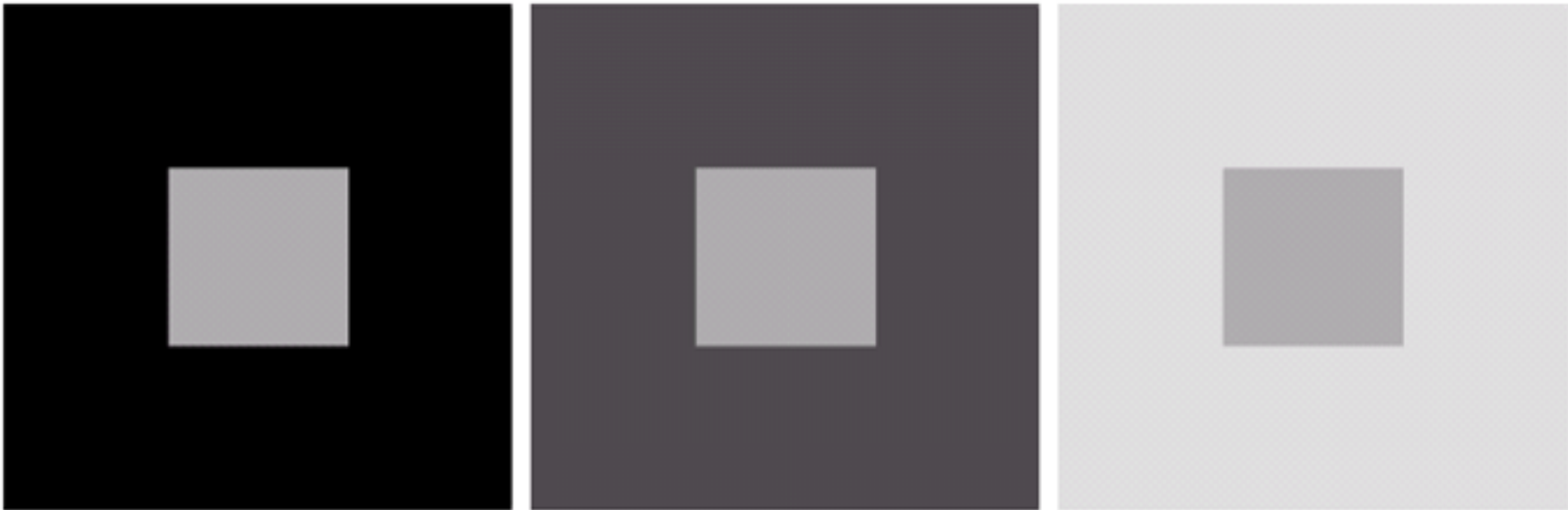
Illustration of Mach band effect  
Perceived intensity is not a simple  
function of the actual intensity

# *Computer Vision*

---

## **Course 2**

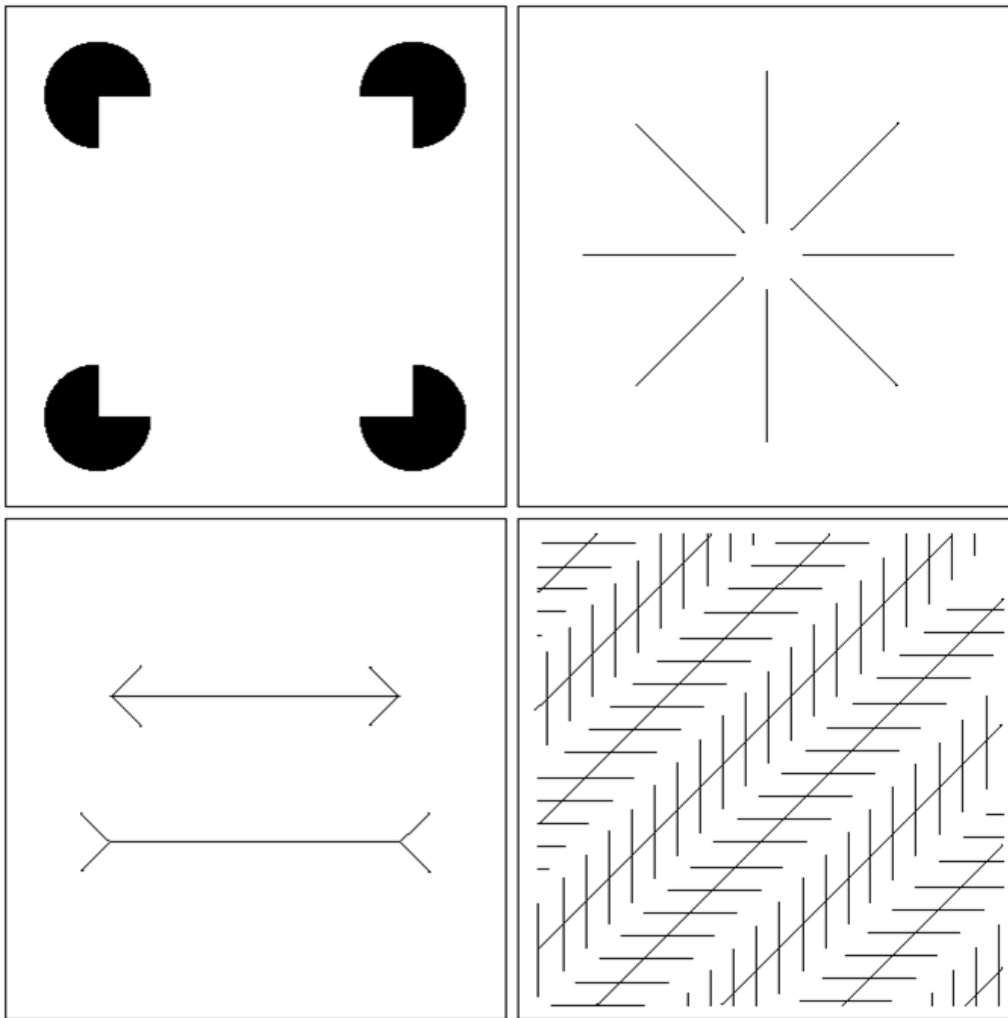
---



All the inner squares have the same intensity, but they appear progressively darker as the background becomes lighter

# Computer Vision

## Course 2



Optical illusions

**ALBASTRU**

**VERDE**

**GALBEN**

**ROSU**

**PORTOCALIU**

**GALBEN**

**VISINIU**

**ALB**

**BLUE**

**GREEN**

**YELLOW**

**RED**

**ORANGE**

**YELLOW**

**BURGUNDY**

**WHITE**

## **Image Sampling and Quantization**

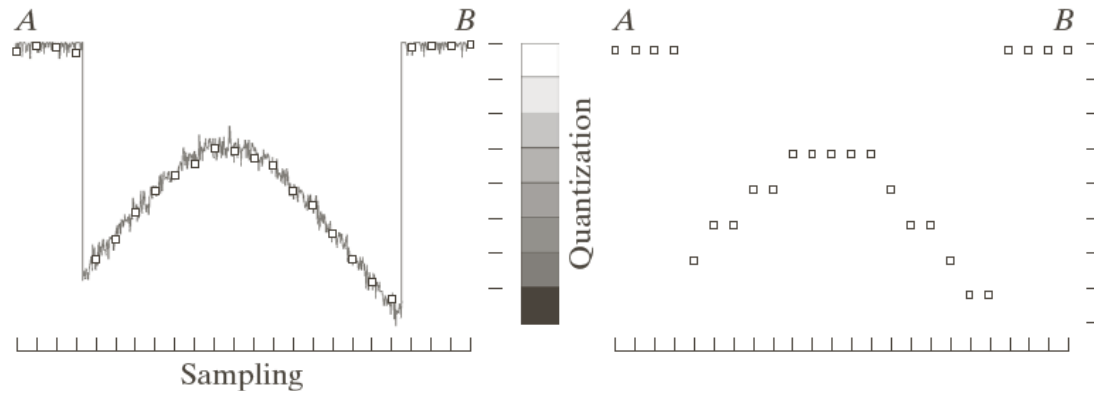
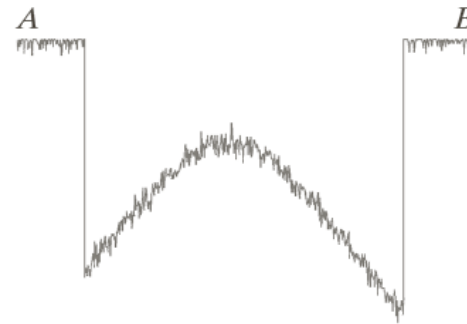
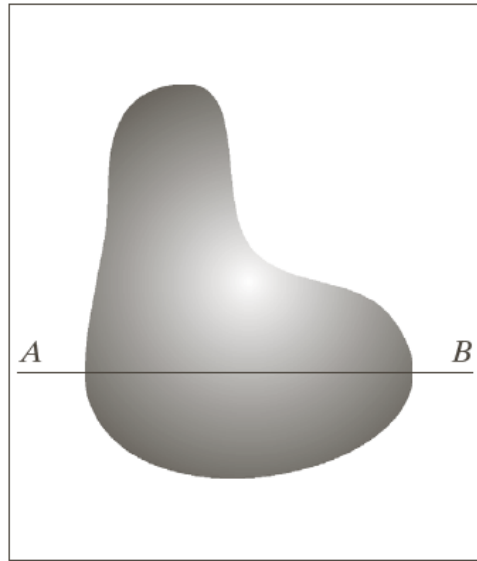
- the output of the sensors is a continuous voltage waveform related to the sensed scene

→ converting a continuous image  $f$  to digital form

1. digitizing  $(x, y)$  is called *sampling*
2. digitizing  $f(x, y)$  is called *quantization*

# Computer Vision

## Course 2

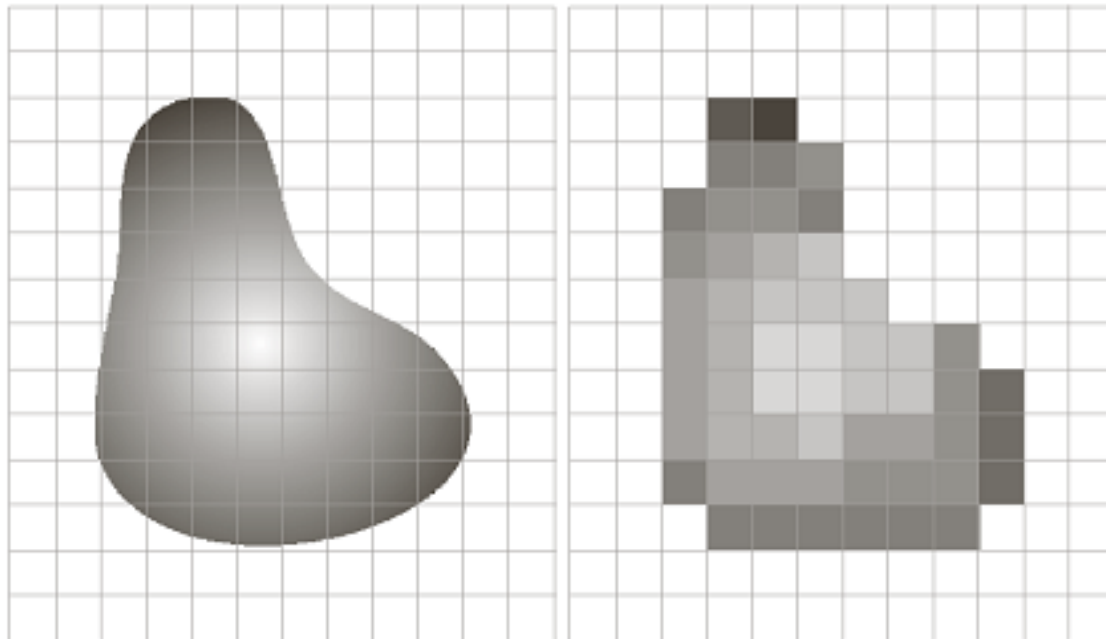


# *Computer Vision*

---

## Course 2

---



Continuous image projected onto a sensor array    Result of image sampling and quantization

# Computer Vision

---

## Course 2

---



### Representing Digital Images

$(x,y) \rightarrow x = 0,1,\dots,M-1, y = 0,1,\dots,N-1$  – *spatial variables or spatial coordinates*

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

$$A = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & & \vdots \\ a_{M-1,0} & a_{M-1,1} & \cdots & a_{M-1,N-1} \end{bmatrix} \in \mathbb{R}^{M \times N}, \quad \begin{aligned} a_{i,j} &= f(x=i, y=j) = f(i,j) \\ a_{i,j} &\text{ – image element, pixel} \end{aligned}$$

$f(0,0)$  – the upper left corner of the image

# Computer Vision

---

## Course 2

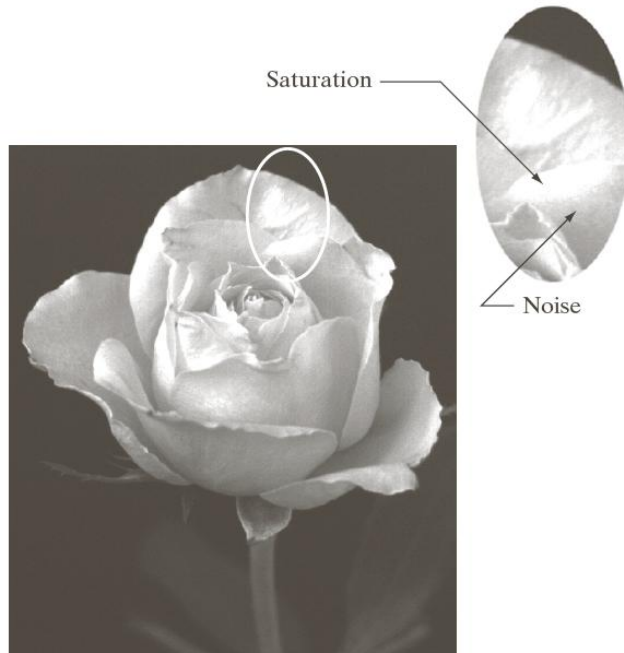
---

$$M, N \geq 0, \quad L=2^k$$

$$a_{i,j} \in \mathbb{N}, \quad a_{i,j} \in [0, L-1]$$

**Dynamic range** of an image = the ratio of the maximum measurable intensity to the minimum detectable intensity level in the system

Upper limit – determined by *saturation*, lower limit - *noise*



# *Computer Vision*

---

## **Course 2**

---

[http://xahlee.info/img/image\\_editing.html](http://xahlee.info/img/image_editing.html)



# *Computer Vision*

---

## **Course 2**

---



# *Computer Vision*

---

## Course 2

---



# *Computer Vision*

---

## **Course 2**

---



# *Computer Vision*

---

## **Course 2**

---



# *Computer Vision*

---

## **Course 2**

---



# *Computer Vision*

---

## **Course 2**

---



# Computer Vision

---

## Course 2

---

Number of bits required to store a digitized image:

$$b = M \times N \times k, \quad \text{for } M = N, \quad b = N^2 k$$

When an image can have  $2^k$  intensity levels, the image is referred as a *k-bit image*  
256 discrete intensity values – 8-bit image

**TABLE 2.1**

Number of storage bits for various values of  $N$  and  $k$ .

$N/k$	1 ( $L = 2$ )	2 ( $L = 4$ )	3 ( $L = 8$ )	4 ( $L = 16$ )	5 ( $L = 32$ )	6 ( $L = 64$ )	7 ( $L = 128$ )	8 ( $L = 256$ )
32	1,024	2,048	3,072	4,096	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,384	32,768	49,152	65,536	81,920	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	393,216	458,752	524,288
512	262,144	524,288	786,432	1,048,576	1,310,720	1,572,864	1,835,008	2,097,152
1024	1,048,576	2,097,152	3,145,728	4,194,304	5,242,880	6,291,456	7,340,032	8,388,608
2048	4,194,304	8,388,608	12,582,912	16,777,216	20,971,520	25,165,824	29,369,128	33,554,432
4096	16,777,216	33,554,432	50,331,648	67,108,864	83,886,080	100,663,296	117,440,512	134,217,728
8192	67,108,864	134,217,728	201,326,592	268,435,456	335,544,320	402,653,184	469,762,048	536,870,912

**Spatial and Intensity Resolution**

*Spatial resolution* – the smallest discernible detail in an image

Measures: *line pairs per unit distance, dots (pixels) per unit distance*

Image resolution = the largest number of discernible line pairs per unit distance

(e.g. 100 line pairs per mm)

Dots per unit distance are commonly used in printing and publishing

In U.S. the measure is expressed in *dots per inch (dpi)*

(newspapers are printed with 75 dpi, glossy brochures at 175 dpi)

*Intensity resolution* – the smallest discernible change in intensity level

The number of intensity levels ( $L$ ) is determined by hardware considerations

$L=2^k$  – most common  $k = 8$

Intensity resolution, in practice, is given by  $k$  (number of bits used to quantize intensity)

# Computer Vision

## Course 2



Fig.1 Reducing spatial resolution: 1250 dpi(upper left), 300 dpi (upper right)  
150 dpi (lower left), 72 dpi (lower right)

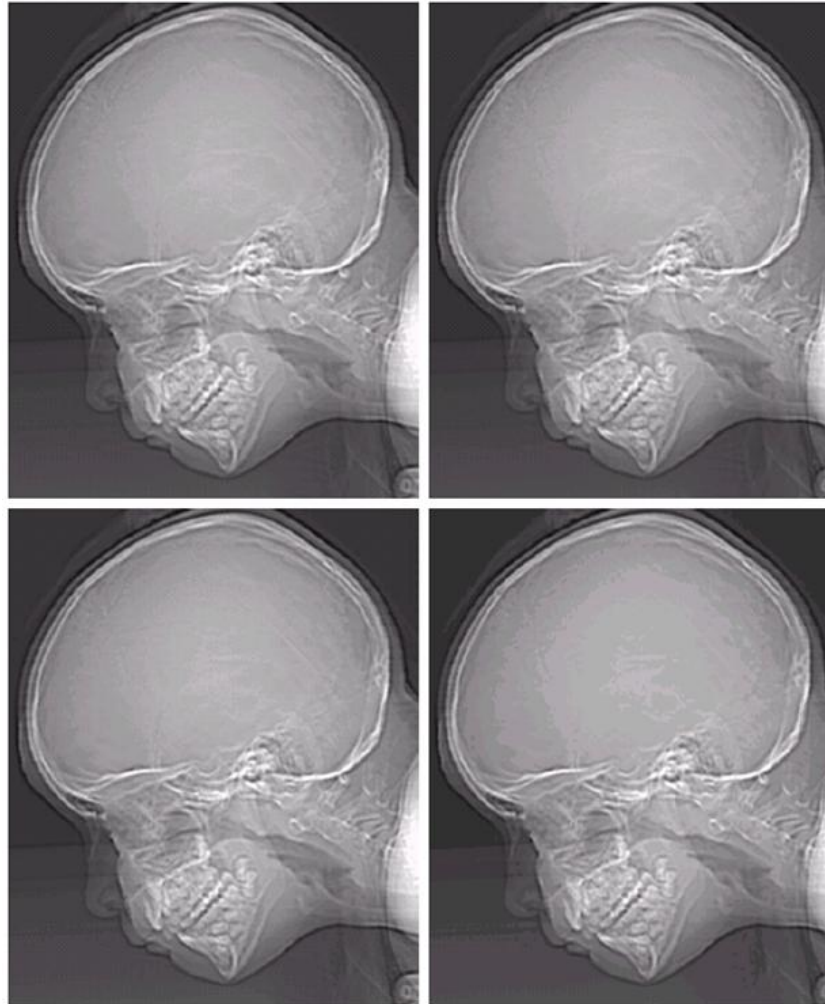
# *Computer Vision*

---

## **Course 2**

---

Reducing the number of gray levels: 256, 128, 64, 32



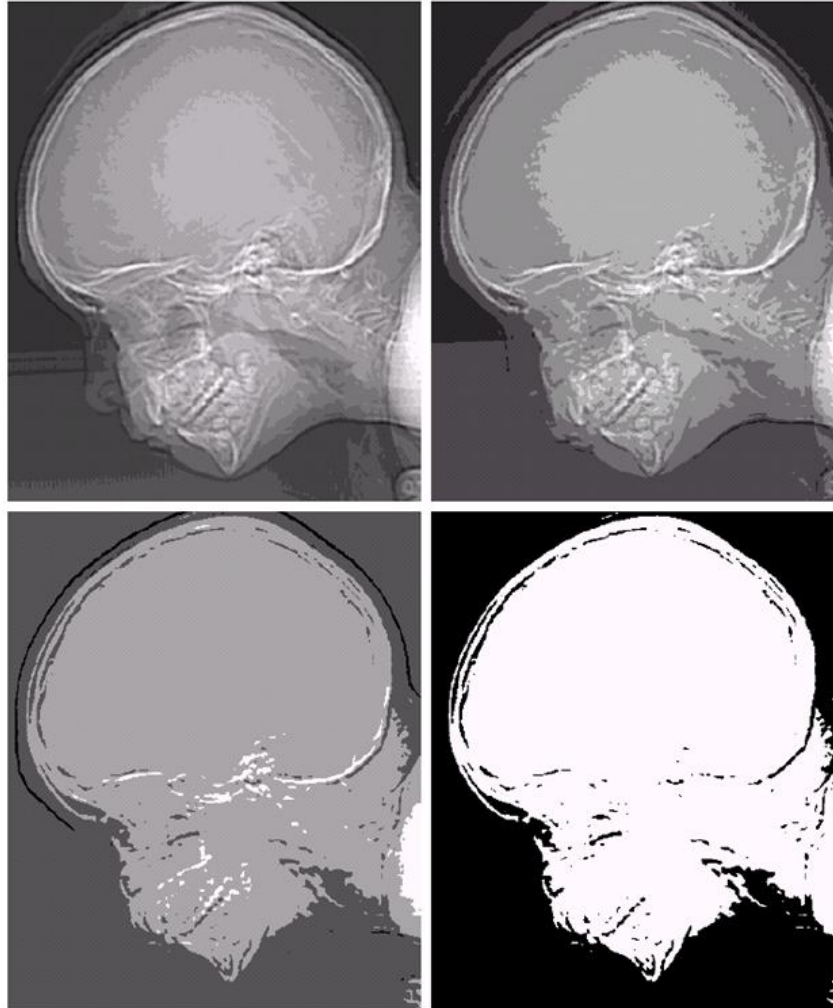
# *Computer Vision*

---

## **Course 2**

---

Reducing the number of gray levels: 16, 8, 4, 2



**Image Interpolation**

- used in zooming, shrinking, rotating, and geometric corrections

Shrinking, zooming – image resizing – image resampling methods

*Interpolation* is the process of using known data to estimate values at unknown locations

Suppose we have an image of size  $500 \times 500$  pixels that has to be enlarged 1.5 times to  $750 \times 750$  pixels. One way to do this is to create an imaginary  $750 \times 750$  grid with the same spacing as the original, and then shrink it so that it fits exactly over the original image. The pixel spacing in the  $750 \times 750$  grid will be less than in the original image.

*Problem:* assignment of intensity-level in the new  $750 \times 750$  grid



**Nearest neighbor interpolation:** assign for every point in the new grid ( $750 \times 750$ ) the intensity of the closest pixel (nearest neighbor) from the old/original grid ( $500 \times 500$ ). This technique has the tendency to produce undesirable effects, like severe distortion of straight edges.

**Bilinear interpolation** – assign for the new  $(x, y)$  location the following intensity:

$$v(x, y) = a x + b y + c x y + d$$

where the four coefficients are determined from the 4 equations in 4 unknowns that can be written using the 4 nearest neighbors of point  $(x, y)$ .

Bilinear interpolation gives much better results than nearest neighbor interpolation, with a modest increase in computational effort.

**Bicubic interpolation** – assign for the new  $(x, y)$  location an intensity that involves the 16 nearest neighbors of the point:

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

The coefficients  $c_{i,j}$  are obtained solving a **16x16** linear system:

$$\sum_{i=0}^3 \sum_{j=0}^3 c_{i,j} x^i y^j = \text{intensity levels of the 16 nearest neighbors of } (x, y)$$

Generally, bicubic interpolation does a better job of preserving fine detail than the bilinear technique. Bicubic interpolation is the standard used in commercial image editing programs, such as Adobe Photoshop and Corel Photopaint.

Figure 2 (a) is the same as Fig. 1 (d), which was obtained by reducing the resolution of the 1250 dpi in Fig. 1(a) to 72 dpi (the size shrank from **3692** × **2812** to **213** × **162**) and

## *Computer Vision*

---

### **Course 2**

---

then zooming the reduced image back to its original size. To generate Fig. 1(d) nearest neighbor interpolation was used (both for shrinking and zooming).

Figures 2(b) and (c) were generated using the same steps but using bilinear and bicubic interpolation, respectively. Figures 2(d)+(e)+(f) were obtained by reducing the resolution from 1250 dpi to 150 dpi (instead of 72 dpi)

# Computer Vision

## Course 2

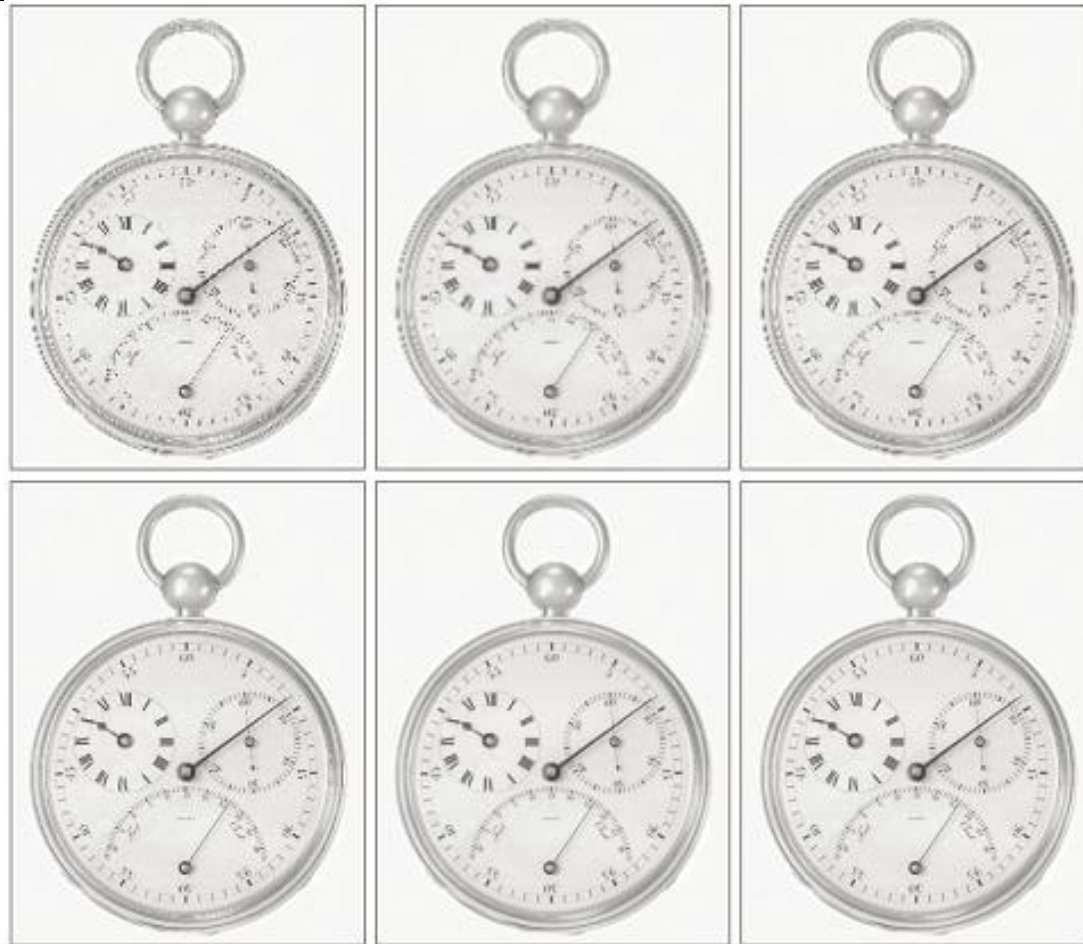


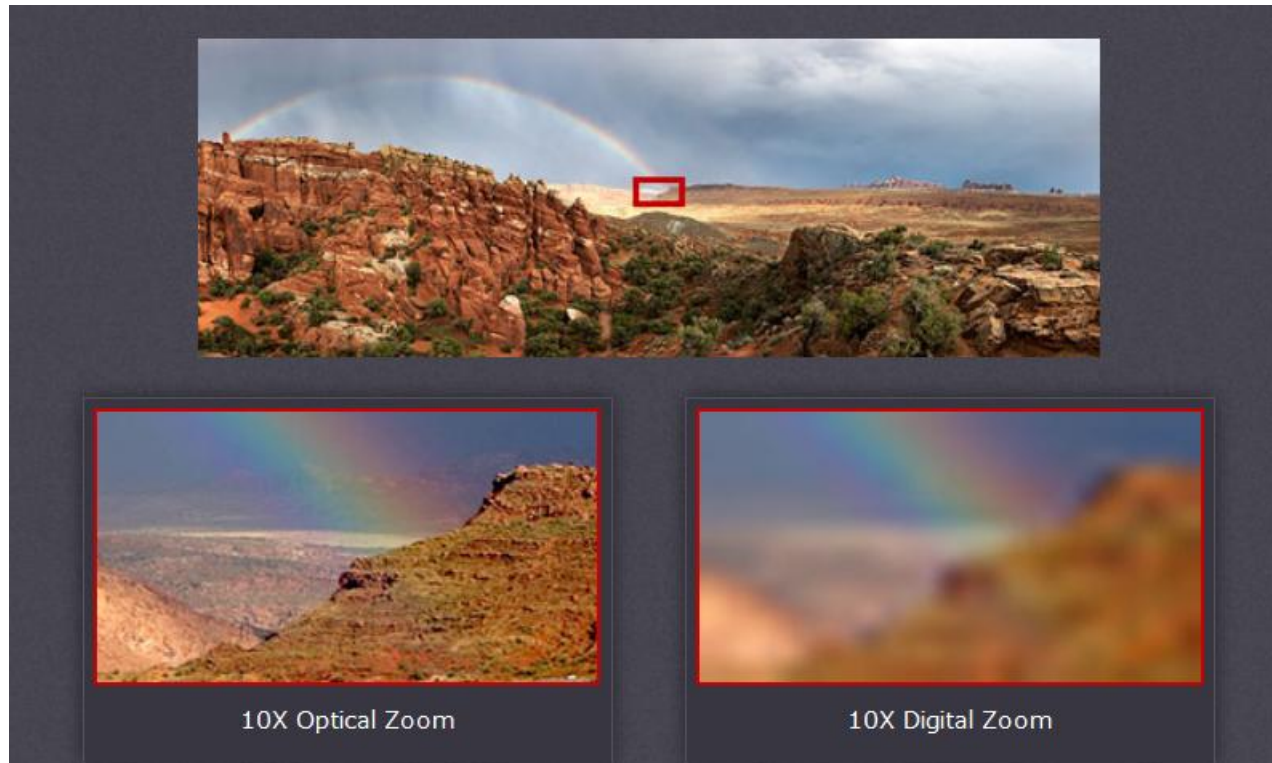
Fig. 2 – Interpolation examples for zooming and shrinking (nearest neighbor, linear, bicubic)

# Computer Vision

---

## Course 2

---



<https://www.cambridgeincolour.com/tutorials/image-interpolation.htm>

**Neighbors of a Pixel**

A pixel  $p$  at coordinates  $(x, y)$  has 4 *horizontal* and *vertical* neighbors:

horizontal:  $(x + 1, y)$ ,  $(x - 1, y)$  ; vertical:  $(x, y + 1)$ ,  $(x, y - 1)$

This set of pixels, called the *4-neighbors* of  $p$ , denoted by  $N_4(p)$ .

The 4 *diagonal* neighbors of  $p$  have coordinates:

$(x + 1, y + 1)$ ,  $(x + 1, y - 1)$ ,  $(x - 1, y + 1)$ ,  $(x - 1, y - 1)$

are denoted  $N_D(p)$ .

The horizontal, vertical and diagonal neighbors are called the *8-neighbors* of  $p$ , denoted  $N_8(p)$ .

If  $(x, y)$  is on the border of the image some of the neighbor locations in  $N_D(p)$  and  $N_8(p)$  fall outside the image.

### Adjacency, Connectivity, Regions, Boundaries

Denote by  $V$  the set of intensity levels used to define adjacency.

3. in a binary image  $V \subseteq \{0,1\}$  ( $V=\{0\}$ ,  $V=\{1\}$ )

4. in a gray-scale image with 256 possible gray-levels,  $V$  can be any subset of  $\{0,255\}$

We consider 3 types of adjacency:

(a) *4-adjacency* : two pixels  $p$  and  $q$  with values from  $V$  are *4-adjacent* if  $q \in N_4(p)$

(b) *8-adjacency* : two pixels  $p$  and  $q$  with values from  $V$  are *8-adjacent* if  $q \in N_8(p)$

(c) *m-adjacency* (mixed adjacency) : two pixels  $p$  and  $q$  with values from  $V$  are *m-adjacent* if :

➤  $q \in N_4(p)$  or

➤  $q \in N_D(p)$  and the set  $N_4(p) \cap N_4(q)$  has no pixels whose values are from  $V$ .

Mixed adjacency is a modification of 8-adjacency. It is introduced to eliminate the ambiguities that often arise when 8-adjacency is used. Consider the example:

# Computer Vision

---

## Course 2

---

$$V = \{1\} - \text{binary image} \quad \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 & \dots & 1 \\ & \vdots & \ddots & \\ 0 & 1 & & 0 \\ & & \ddots & \\ 0 & 0 & & 1 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 & \dots & 1 \\ & \vdots & & \\ 0 & 1 & & 0 \\ & & \ddots & \\ 0 & 0 & & 1 \end{pmatrix}$$

The three pixels at the top (first line) in the above example show multiple (ambiguous) 8-adjacency, as indicated by the dashed lines. This ambiguity is removed by using  $m$ -adjacency.

## Computer Vision

---

### Course 2

---

A (*digital*) *path* (or *curve*) from pixel  $p$  with coordinates  $(x,y)$  to  $q$  with coordinates  $(s,t)$  is a sequence of distinct pixels with coordinates:

$$(x_0, y_0) = (x, y), (x_1, y_1), \dots, (x_n, y_n) = (s, t)$$
$$(x_{i-1}, y_{i-1}) \text{ and } (x_i, y_i) \text{ are adjacent, } i = 1, 2, \dots, n$$

The *length* of the path is  $n$ . If  $(x_0, y_0) = (x_n, y_n)$  the path is *closed*.

Depending on the type of adjacency considered the paths are: 4-, 8-, or  $m$ -paths.

Let  $S$  denote a subset of pixels in an image. Two pixels  $p$  and  $q$  are said to be *connected* in  $S$  if there exists a path between them consisting only of pixels from  $S$ .

$S$  is a *connected set* if there is a path in  $S$  between any 2 pixels in  $S$ .

Let  $R$  be a subset of pixels in an image.  $R$  is a *region* of the image if  $R$  is a connected set.

Two regions  $R_1$  and  $R_2$  are said to be *adjacent* if  $R_1 \cup R_2$  form a connected set. Regions that are not adjacent are said to be *disjoint*. When referring to regions only 4- and 8-adjacency are considered.

## Computer Vision

---

### Course 2

---

Suppose that an image contains  $K$  disjoint regions,  $\mathbf{R}_k$ ,  $k = 1, \dots, K$ , none of which touches the image border.

$$\mathbf{R}_u = \bigcup_{k=1}^K \mathbf{R}_k, \quad (\mathbf{R}_u)^c - \text{the complement of } \mathbf{R}_u$$

We call all the points in  $\mathbf{R}_u$  the *foreground* of the image and the points in  $(\mathbf{R}_u)^c$  the *background* of the image.

The *boundary* (*border* or *contour*) of a region  $\mathbf{R}$  is the set of points that are adjacent to points in the complement of  $\mathbf{R}$ ,  $(\mathbf{R})^c$ . The border of an image is the set of pixels in the region that have at least one background neighbor. This definition is referred to as the *inner border* to distinguish it from the notion of *outer border* which is the corresponding border in the background.

### Distance measures

For pixels  $p$ ,  $q$ , and  $z$ , with coordinates  $(x,y)$ ,  $(s,t)$  and  $(v,w)$  respectively,  $D$  is a *distance function* or *metric* if:

- (a)  $D(p, q) \geq 0$  ,  $D(p, q) = 0$  iff  $p=q$
- (b)  $D(p, q) = D(q, p)$
- (c)  $D(p, z) \leq D(p, q) + D(q, z)$

The *Euclidean distance* between  $p$  and  $q$  is defined as:

$$D_e(p,q) = \left[ (x-s)^2 + (y-t)^2 \right]^{\frac{1}{2}} = \sqrt{(x-s)^2 + (y-t)^2}$$

The pixels  $q$  for which  $D_e(p,q) \leq r$  are the points contained in a disk of radius  $r$  centered at  $(x, y)$ .

# Computer Vision

---

## Course 2

---

The  $D_4$  distance (also called *city-block distance*) between  $p$  and  $q$  is defined as:

$$D_4(p, q) = |x - s| + |y - t|$$

The pixels  $q$  for which  $D_4(p, q) \leq r$  form a diamond centered at  $(x, y)$ .

$$D_4 \leq 2 \rightarrow \begin{array}{ccccc} & & 2 & & \\ & & 2 & 1 & 2 \\ & 2 & 1 & 0 & 1 & 2 \\ & & 2 & 1 & 2 \\ & & & 2 & \end{array}$$

The pixels with  $D_4 = 1$  are the 4-neighbors of  $(x, y)$ .

## Computer Vision

---

### Course 2

---

The  $D_8$  distance (called the *chessboard distance*) between  $p$  and  $q$  is defined as:

$$D_8(p, q) = \max\{|x - s|, |y - t|\}$$

The pixels  $q$  for which  $D_8(p, q) \leq r$  form a square centered at  $(x, y)$ .

$$D_8 \leq 2 \rightarrow \begin{array}{ccccc} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{array}$$

The pixels with  $D_8 = 1$  are the 8-neighbors of  $(x, y)$ .

$D_4$  and  $D_8$  distances are independent of any paths that might exist between  $p$  and  $q$  because these distances involve only the coordinates of the point.

### Array versus Matrix Operations

An array operation involving one or more images is carried out on a *pixel-by-pixel* basis.

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

Array product:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} \\ a_{21}b_{21} & a_{22}b_{22} \end{bmatrix}$$

Matrix product:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix}$$

*We assume array operations unless stated otherwise!*

### Linear versus Nonlinear Operations

One of the most important classifications of image-processing methods is whether it is linear or nonlinear.

$$H[f(x, y)] = g(x, y)$$

$H$  is said to be a *linear operator* if:

$$H[a f_1(x, y) + b f_2(x, y)] = a H[f_1(x, y)] + b H[f_2(x, y)]$$

$\forall a, b \in \mathbb{R}, \forall f_1, f_2 - \text{images}$

Example of nonlinear operator:

$$H[f] = \max\{f(x, y)\} = \text{the maximum value of the pixels of image } f$$

$$f_1 = \begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix}, f_2 = \begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}, a = 1, b = -1$$

## Computer Vision

---

### Course 2

---

$$\max\{a f_1 + b f_2\} = \max\left\{1 \begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix} + (-1) \begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = \max\left\{\begin{bmatrix} -6 & -3 \\ -2 & -4 \end{bmatrix}\right\} = -2$$

$$1 \cdot \max\left\{\begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix}\right\} + (-1) \cdot \max\left\{\begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = 3 + (-1)7 = -4$$

### Linear versus Nonlinear Operations

One of the most important classifications of image-processing methods is whether it is linear or nonlinear.

$$H[f(x, y)] = g(x, y)$$

$H$  is said to be a *linear operator* if:

$$H[a f_1(x, y) + b f_2(x, y)] = a H[f_1(x, y)] + b H[f_2(x, y)]$$

$\forall a, b \in \mathbb{R}, \forall f_1, f_2 - \text{images}$

Example of nonlinear operator:

$$H[f] = \max\{f(x, y)\} = \text{the maximum value of the pixels of image } f$$

$$f_1 = \begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix}, f_2 = \begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}, a = 1, b = -1$$

## Computer Vision

---

### Course 2

---

$$\max\{a f_1 + b f_2\} = \max\left\{1 \begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix} + (-1) \begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = \max\left\{\begin{bmatrix} -6 & -3 \\ -2 & -4 \end{bmatrix}\right\} = -2$$

$$1 \cdot \max\left\{\begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix}\right\} + (-1) \cdot \max\left\{\begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = 3 + (-1)7 = -4$$