AIU.
ALAMEIN
INTERNATIONAL UNIVERSITY
جامعة العلمين الدولية
QUALITY ASSURANCE CENTER

Image Processing (CSE281)

Fall 2025/2026

Dr. Essam Abdellatef

Contact Number: 012 8 192 55 90

Email: essam.abdellatef@su.edu.eg

Introduction

"One picture is worth more than ten thousand words"

Introduction

Image processing is the technique of performing operations on digital images to enhance their quality, extract useful information, or prepare them for further analysis.

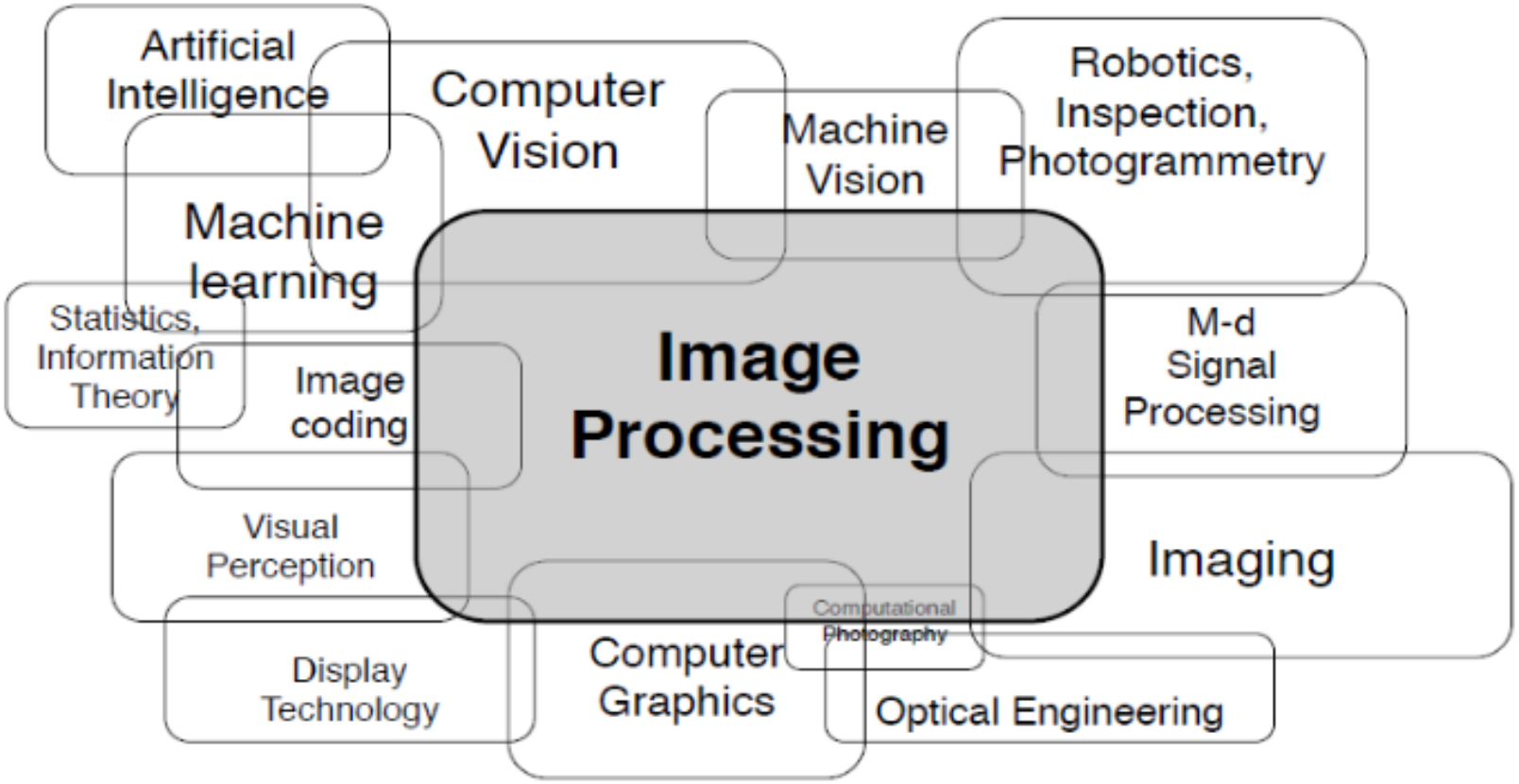
It involves algorithms and methods that manipulate pixel values to achieve tasks such as improvement, restoration, compression, segmentation, or feature extraction.

Introduction

What We See

What Computers See

Introduction



Digital Image ?

An image may be defined as a two-dimensional function f(x, y), where x, y: the ***spatial coordinate, f-the amplitude of any pair of coordinate x, y is*** called ***the intensity or gray level of the image at that point.***

Digital Image: x, y and f are all finite (discrete quantities).

Digital Image ?

Digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are called ***picture elements (PEL), image elements and pixels.***

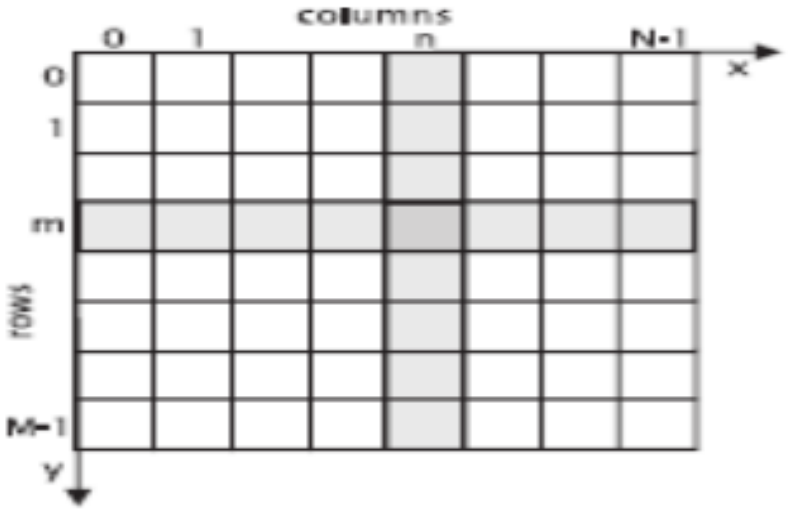
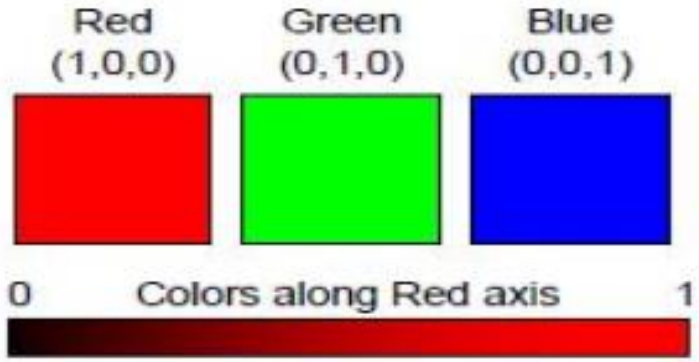


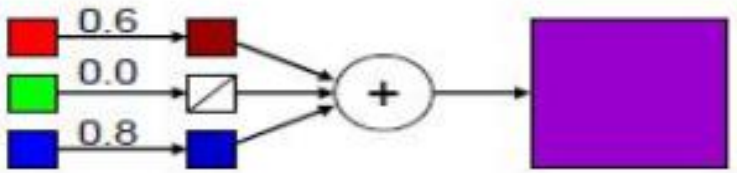
Image Representation

Discrete representation of images

- We'll carve up image into a rectangular grid of pixels *P[x,y]*
- Each pixel *p will store an intensity value in [0 1]*
- 0 → black; 1 → white; in-between → gray
- Image size mxn →(mn) pixels

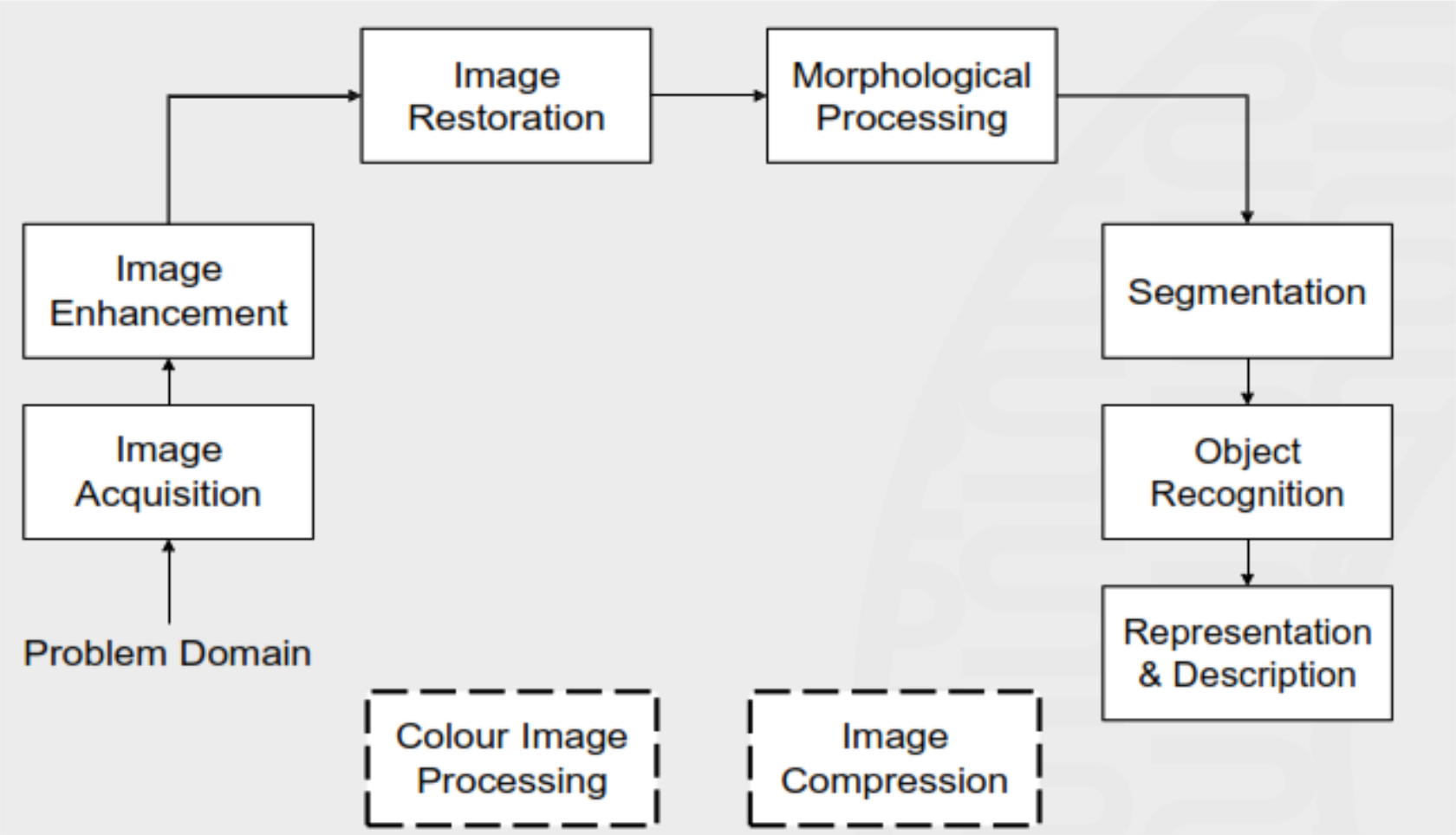
Color Image


Red
(1,0,0)
Green
(0,1,0)
Blue
(0,0,1)
0
Colors along Red axis
1


0.6
0.0
0.8
+


RGB
channels

Key Stages in Digital Image Processing



Key Stages in Digital Image Processing

Image Acquisition

- The process of capturing a real-world image and converting it into a digital form suitable for computer processing.

Image Enhancement

- Techniques used to improve the visual appearance of an image or to highlight certain features for better interpretation. Examples include contrast adjustment, histogram equalization, noise reduction, and sharpening.

Key Stages in Digital Image Processing

Image Restoration

- The process of reconstructing or recovering an image that has been degraded by known causes such as noise, blur, or motion.

Morphological Processing

- A set of image processing techniques that deal with the shape and structure of objects in an image.

Key Stages in Digital Image Processing

Image Segmentation

- The process of partitioning an image into meaningful regions or objects to simplify its analysis. Segmentation separates foreground from background or divides an image into regions based on properties like color, intensity, or texture.

Object Recognition

- The task of identifying and classifying objects within an image. It involves labeling objects (car, face, tree) based on their shape, color, texture, or learned features, often using machine learning or deep learning models.

Key Stages in Digital Image Processing

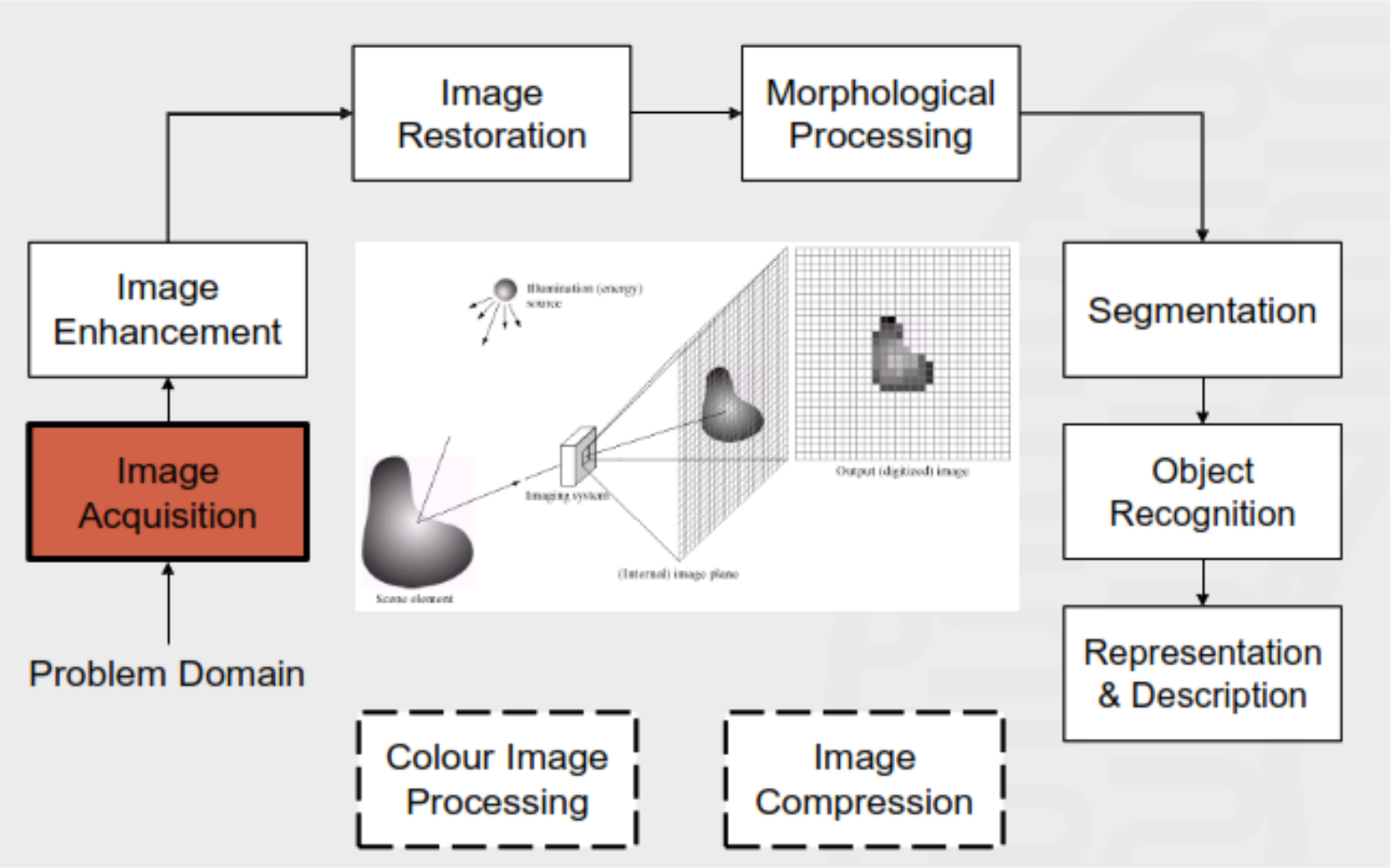
Image Representation and Description

- The step of converting processed image data into a form suitable for analysis.

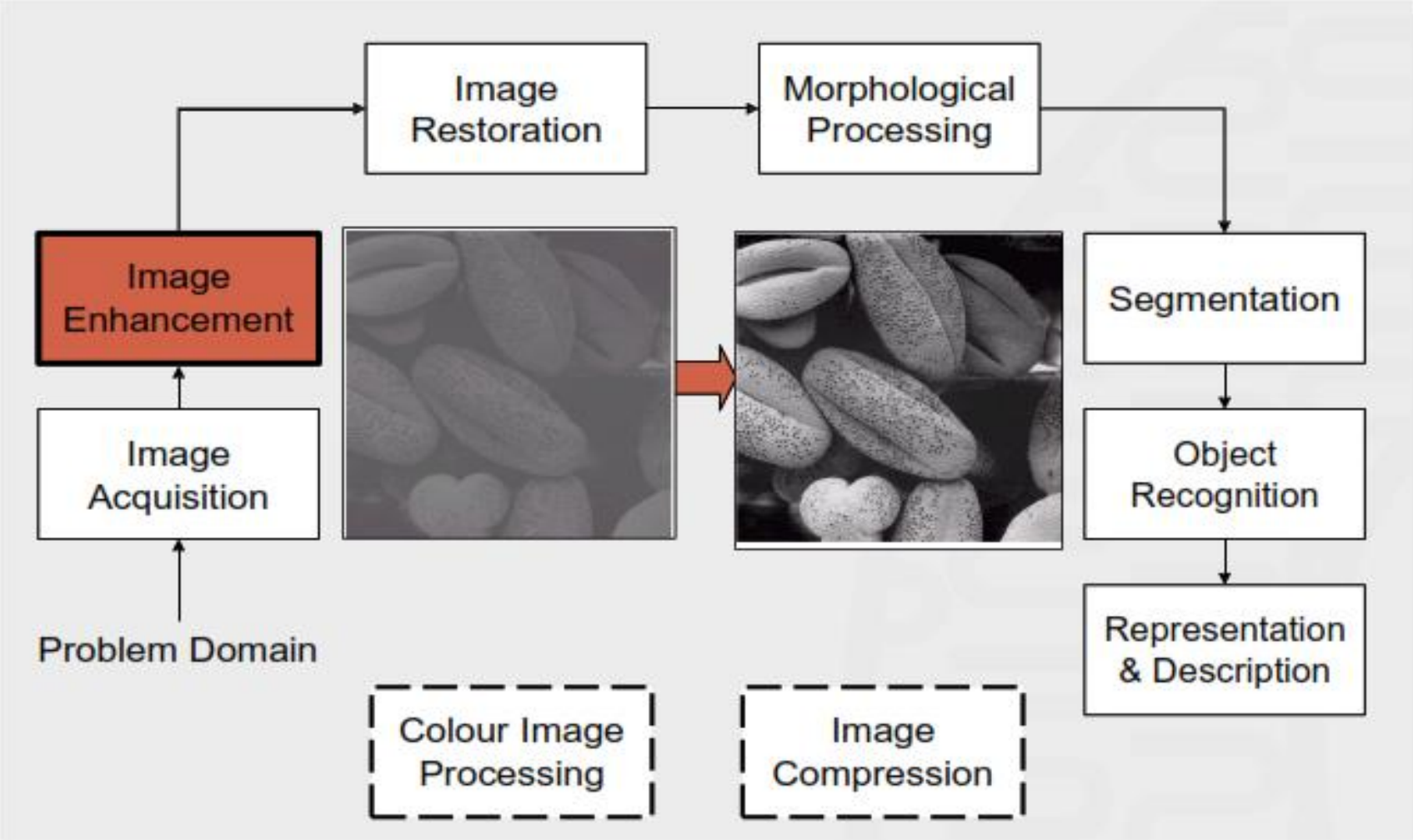
Image Compression

- Reducing the amount of data required to represent an image while preserving acceptable quality.

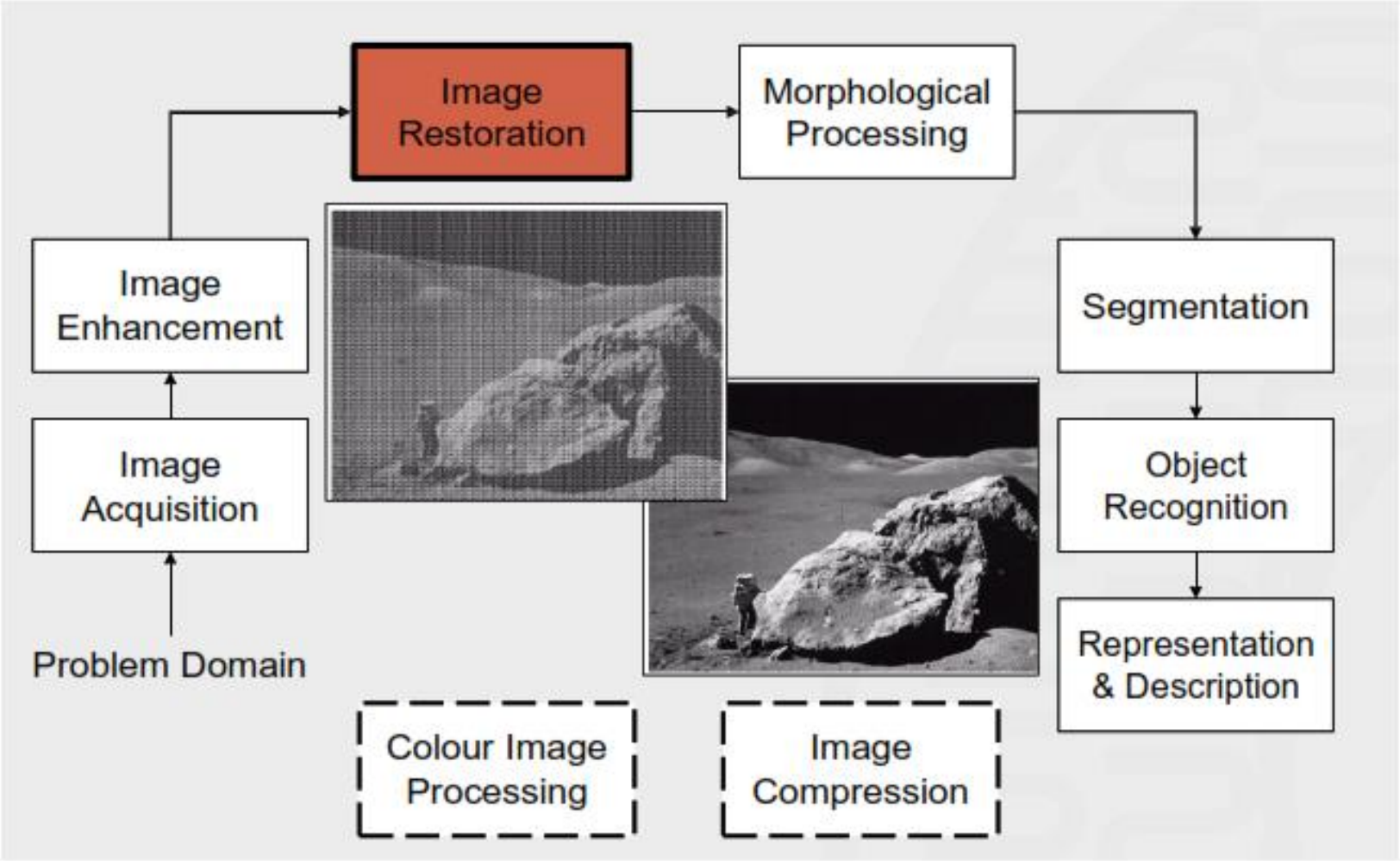
Key Stages in Digital Image Processing



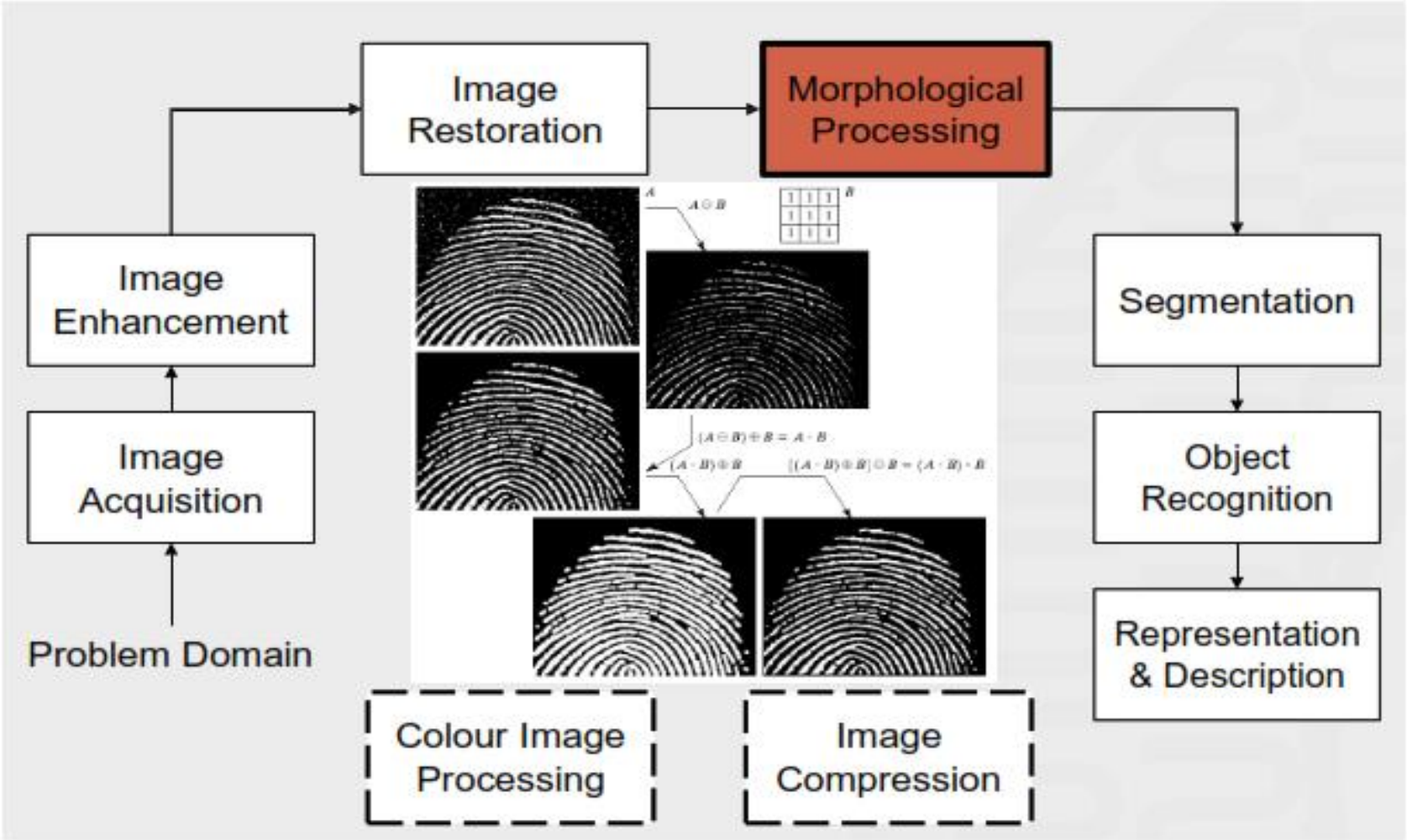
Key Stages in Digital Image Processing



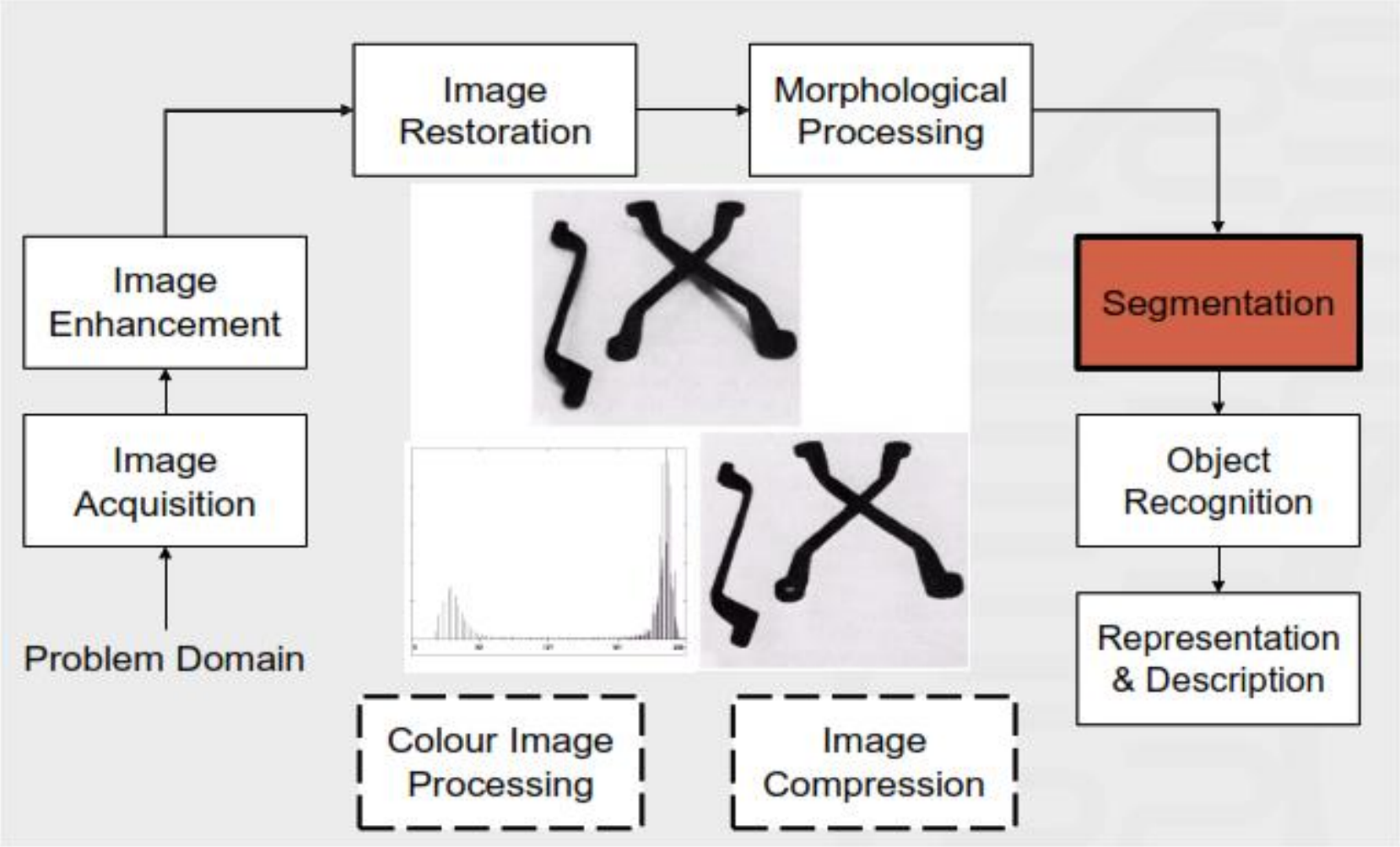
Key Stages in Digital Image Processing



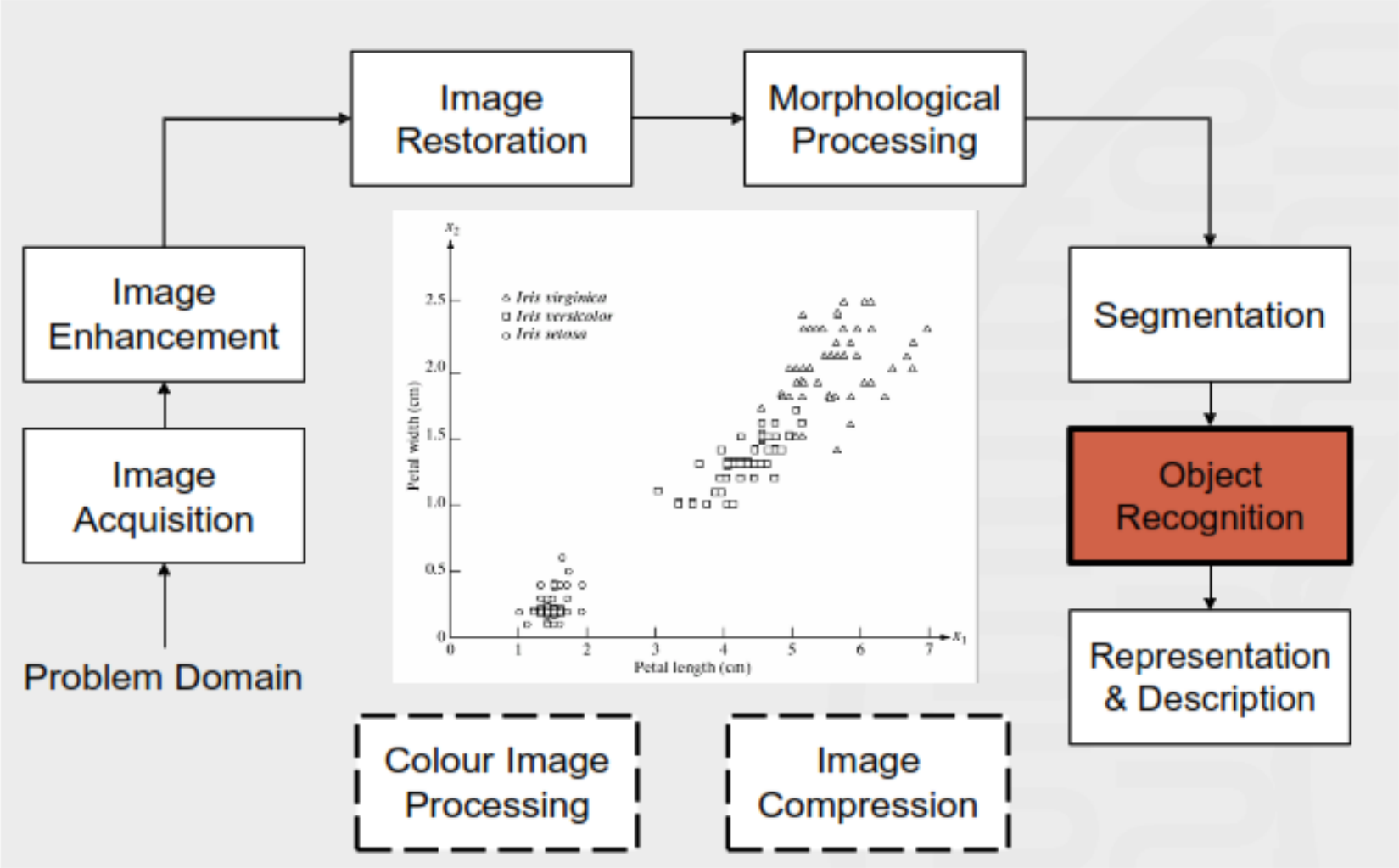
Key Stages in Digital Image Processing



Key Stages in Digital Image Processing



Key Stages in Digital Image Processing



Key Stages in Digital Image Processing

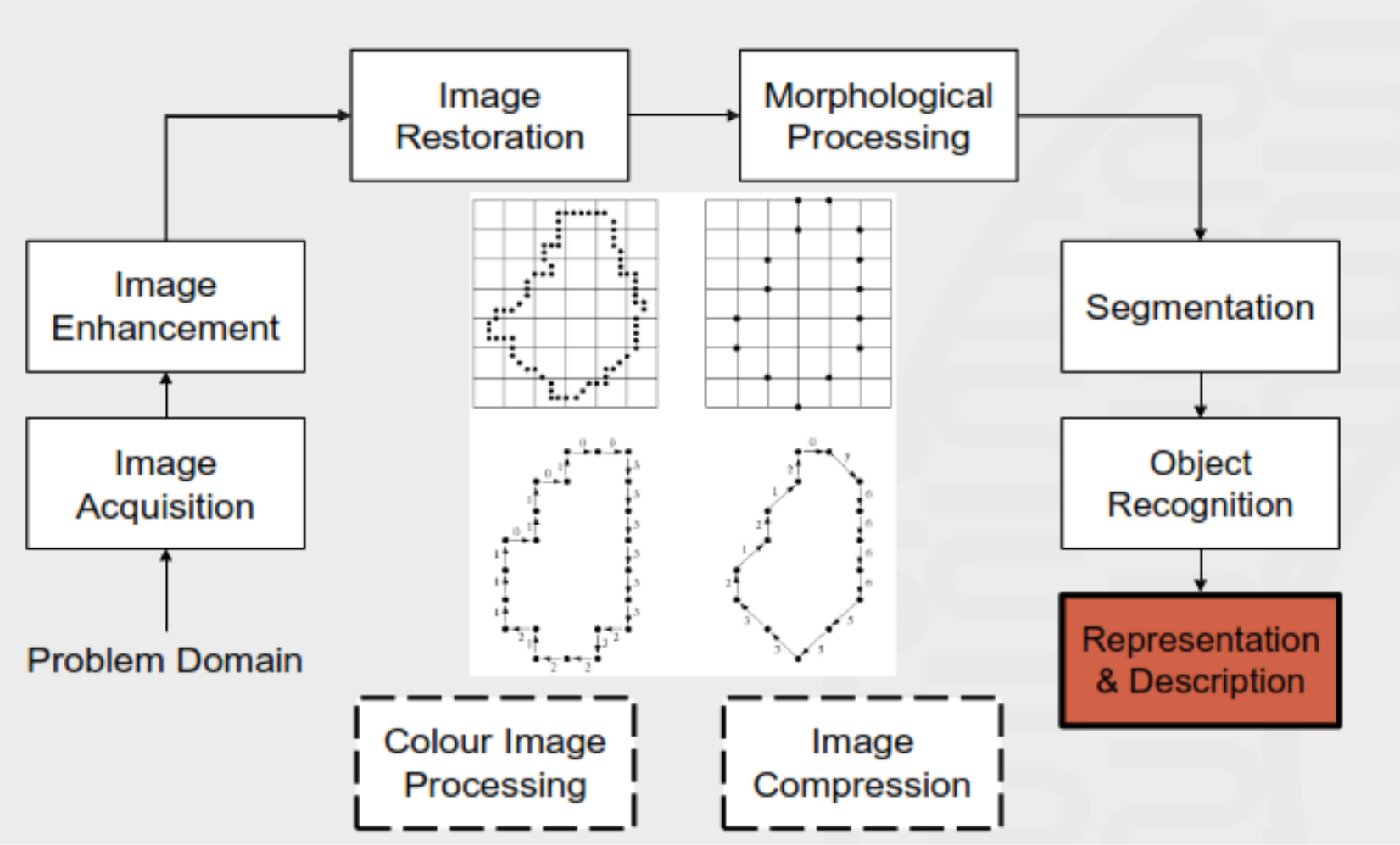


Image Sampling And Quantization

The output of most sensors is a continuous waveform so to create a digital image we need to do two processes:

- ***Image sampling:*** is to digitizing the image coordinates.
- ***Image quantization:*** is to digitizing the image amplitude (intensity level).

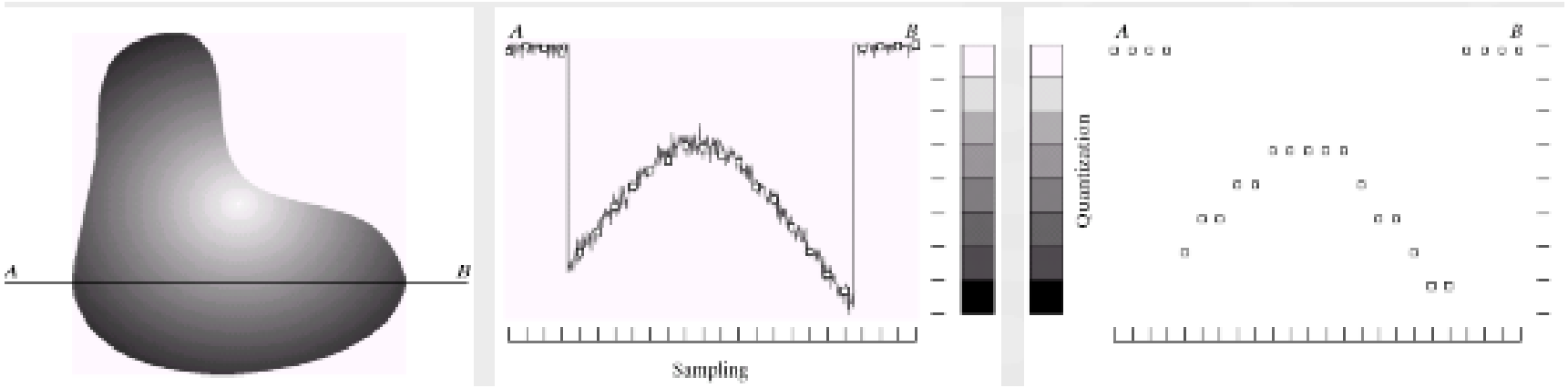
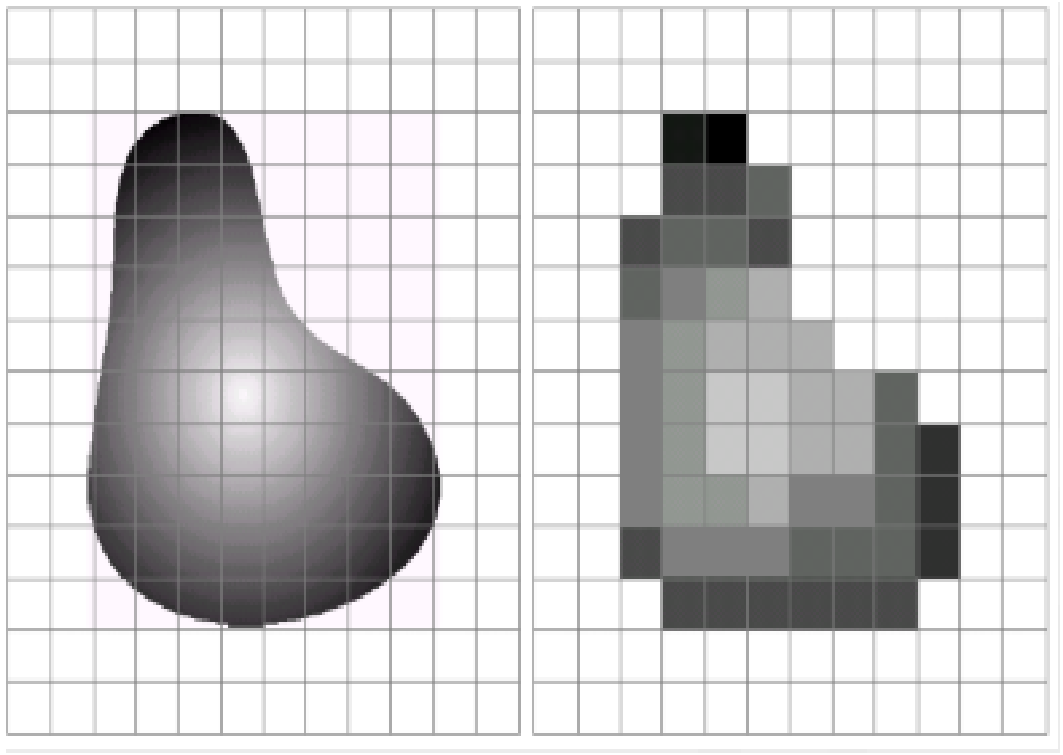


Image Sampling And Quantization

(a) Continuous image projected onto a sensor array.
(b) Result of sampling and quantization (digitized image).



Dynamic Range

It can be defined as the ratio between maximum measurable intensity level and minimum detectable intensity level in an image.

Image contrast: is the difference in intensity between the highest and lowest intensity level in image.

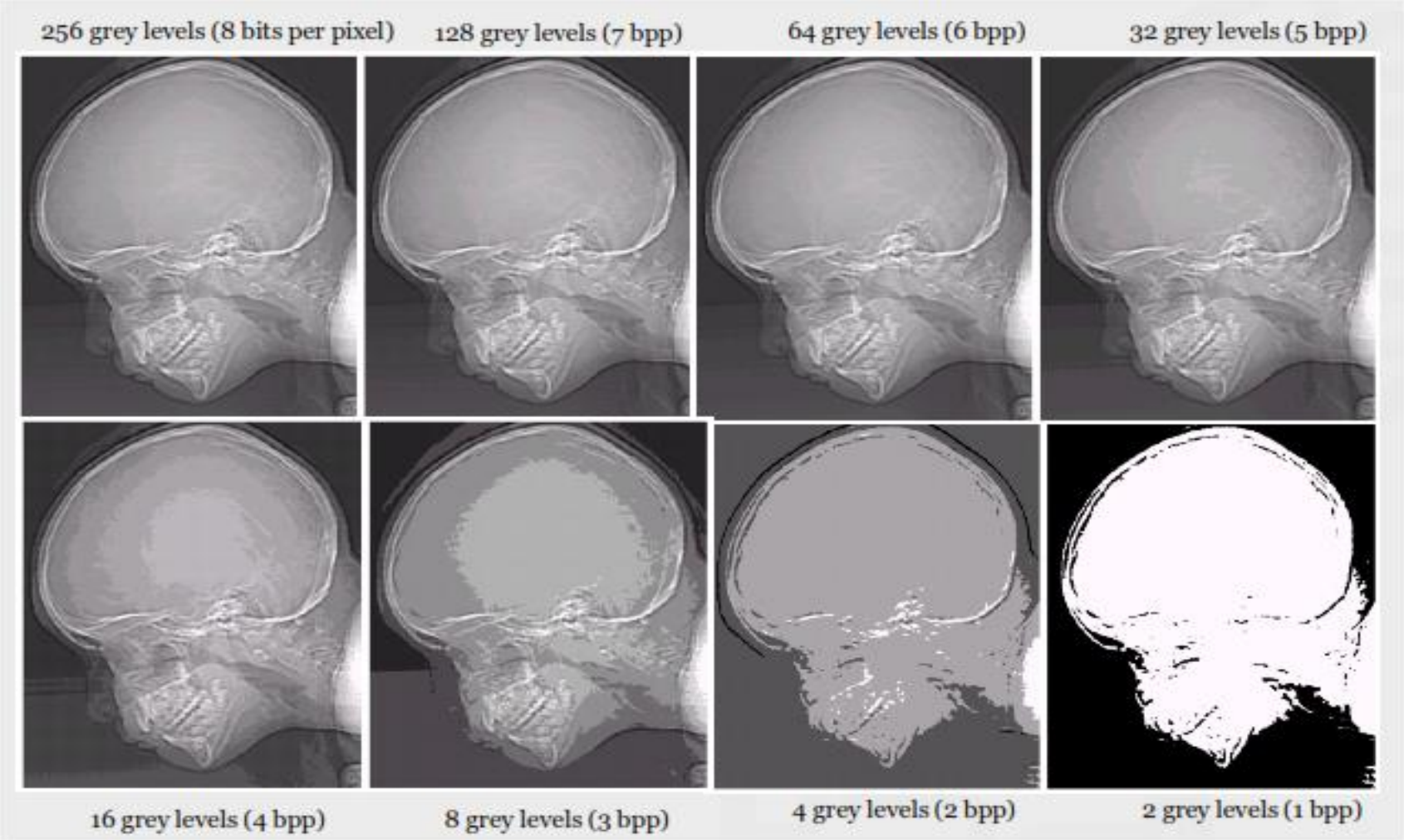
Intensity Resolution

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

Intensity Resolution

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	65536	1010101010101010

Intensity Resolution



Basic Relationships between Pixels

Neighbors of pixel

A pixel ‘p’ at coordinates (x, y) has Four horizontal and vertical neighbors called 4- neighbors N4(p): ***(x+1,y),(x-1,y),(x,y+1),(x,y-1)***

Four diagonal neighbors of ‘p’ called ND(p): ***(x+1,y+1),(x+1,y-1),(x-1,y+1),(x-1,y-1)*** N4(p) and Nd(p) together are constructing N8(p).

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) $D(p, q) \geq 0 \quad (D(p, q) = 0 \quad \text{iff} \quad p = q)$,

(b) $D(p, q) = D(q, p)$, and

(c) $D(p, z) \leq D(p, q) + D(q, z)$.

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

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

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

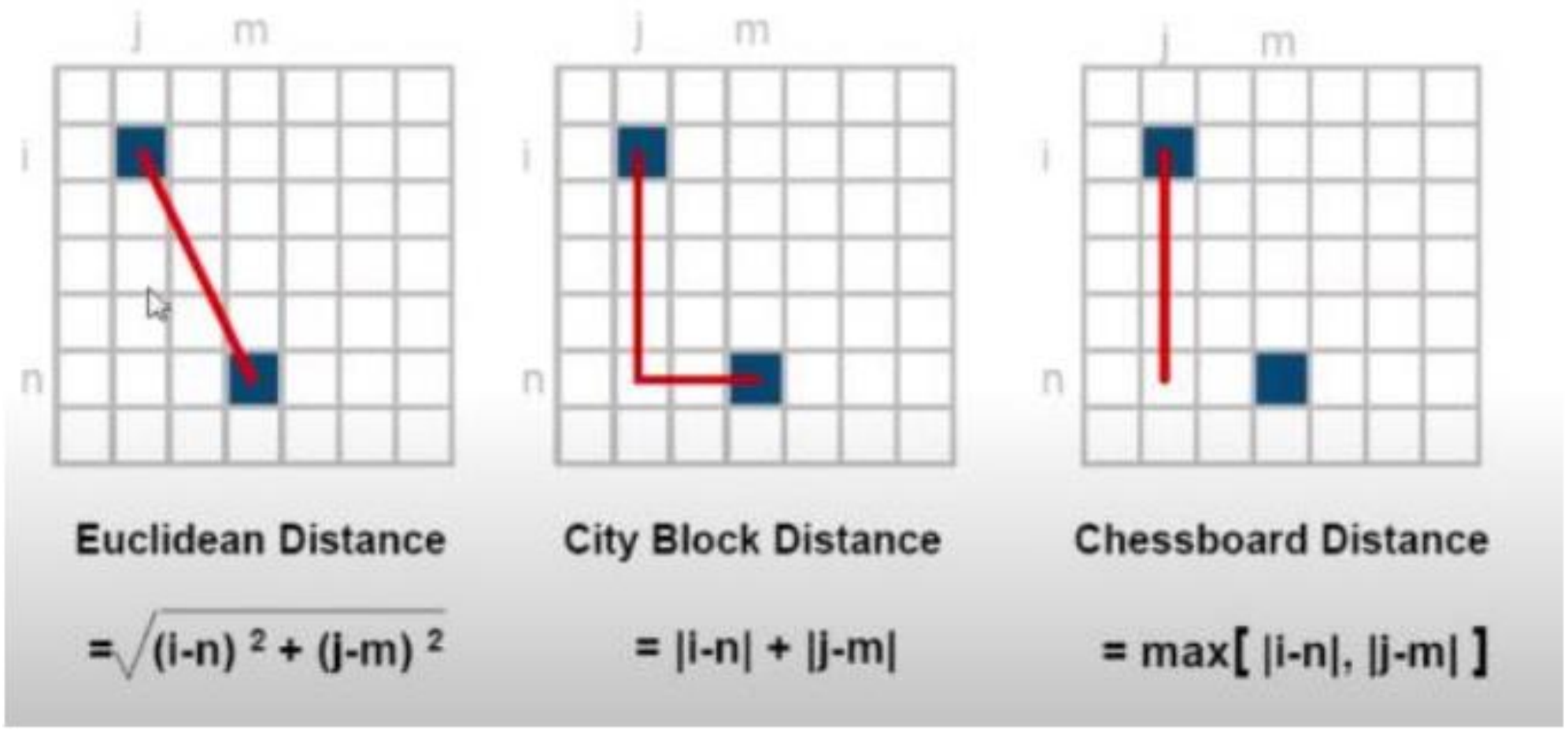
$$D_4(p, q) = |x - s| + |y - t|$$

Distance measures

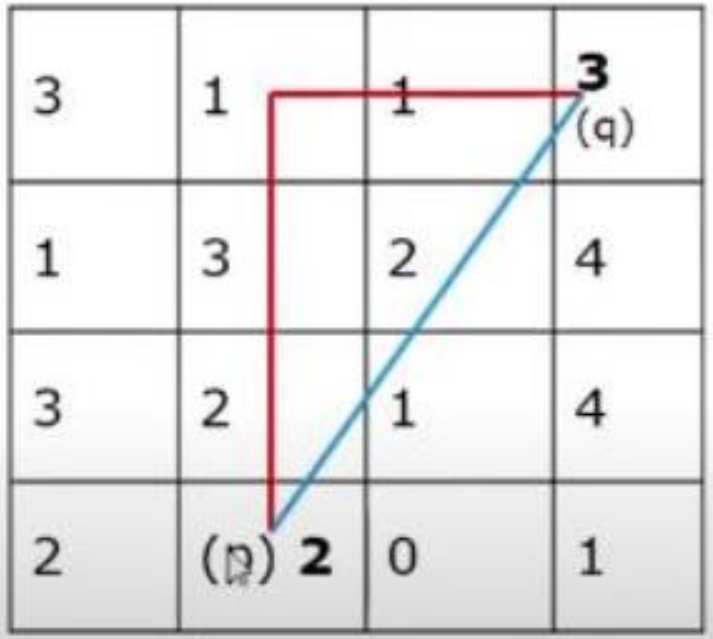
The D_8 *distance* (called the *chessboard distance*) between p and q is defined as

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

Distance measures



Distance measures



D4 = | 1-4| +|2-4|
D4 = 3 + 2 = 5

D8 = max(| 1-4|,|2-4|)
D8 = max(3, 2) = 3

$Euc = ((1-4)^2 + (2-4)^2)^{1/2}$
$Euc = (3^2 + 2^2)^{1/2}$
$Euc = (9 + 4)^{1/2}$
$= 3.6$

Image Interpolation

- ***Image Interpolation*** is the process of using known data to estimate values at unknown location.
- ***Image interpolation*** is used for zooming, rotating, geometric corrections.
- ***Zooming is image resizing task*** and come under ***image re-sampling methods.***

Image Interpolation

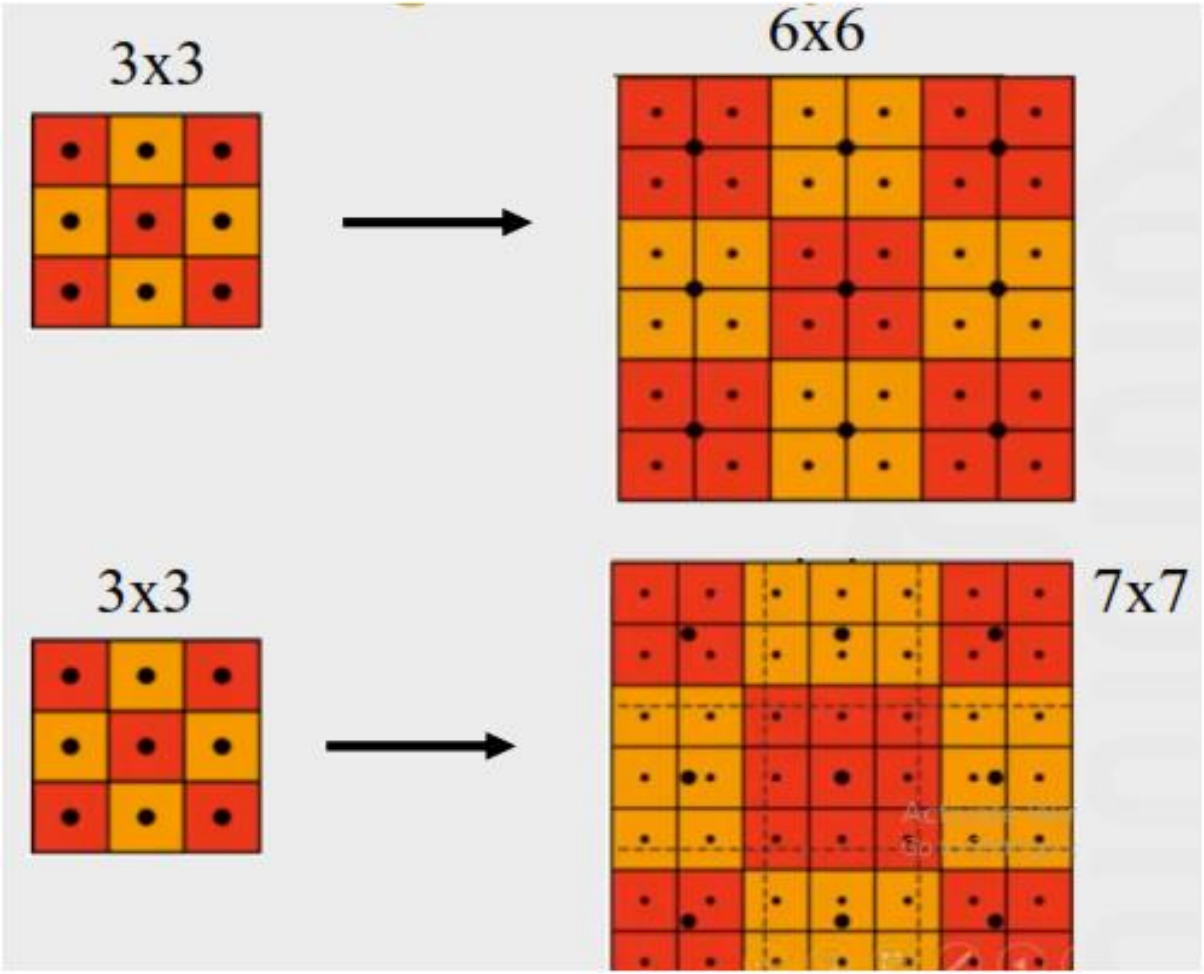
Zooming requires 2 steps:

- The creation of new pixel locations.
- The assignment of gray levels to these new locations.

Two techniques for zooming:

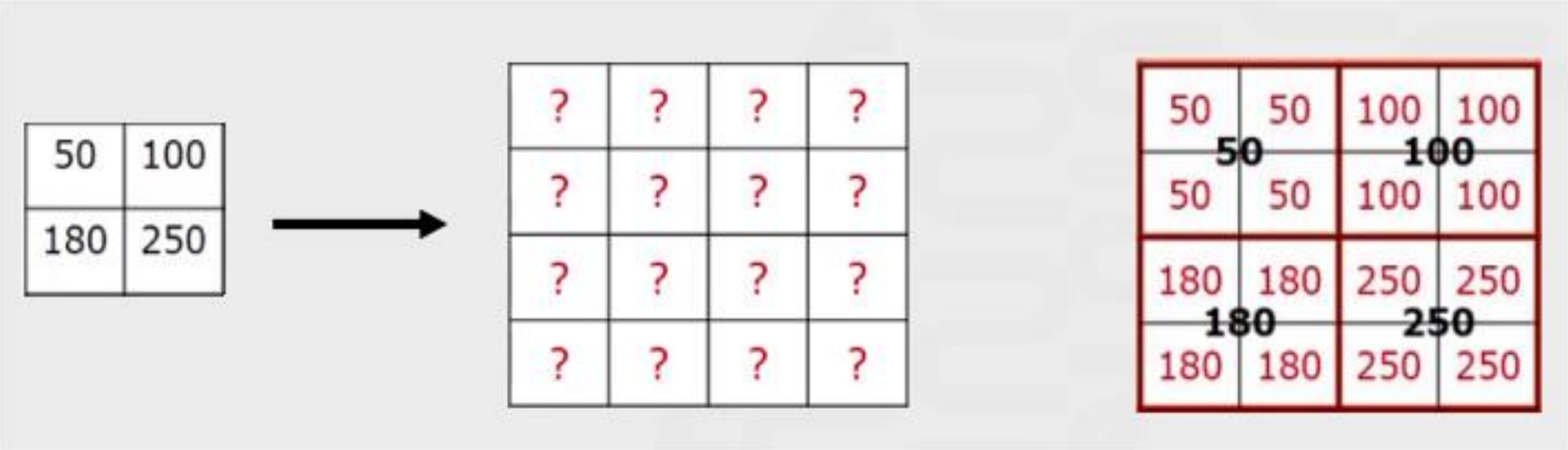
1. Nearest neighbor interpolation
2. Bilinear interpolation

Nearest neighbor interpolation



Nearest neighbor interpolation

Example: Suppose A 2x2 pixels Image will be enlarged 2 times by the nearest neighbor method:



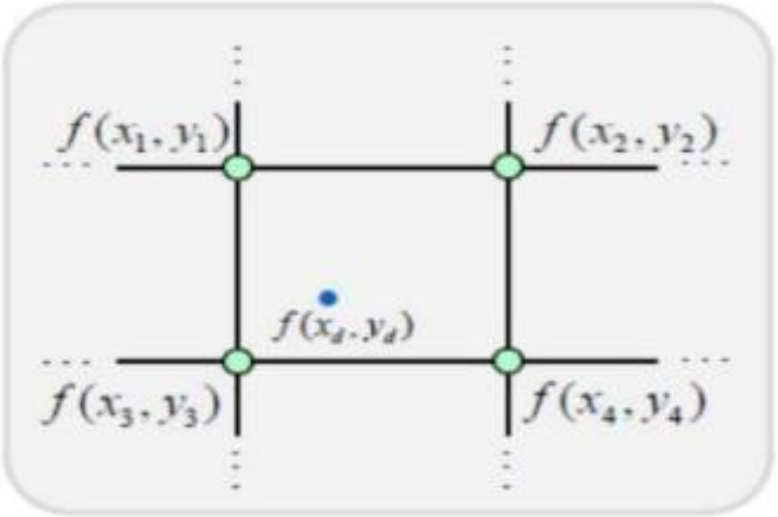
Bilinear interpolation

- Here we use the 4 nearest neighbors to estimate the intensity at a given location.
- Let *(x,y)* denote the coordinates of the location to which we want to assign an intensity value and let *v(x,y)* denote that value, then:

$$v(x, y) = ax + by + cxy + d$$

- Here the 4 coefficients are determined from the 4 equations in 4 unknowns using the 4 nearest neighbors of point *(x,y)*.

Bilinear interpolation



Model:

$$f(x, y) = ax + by + cxy + d$$

coefficients to be estimated:

a, b, c, d

Using the 4 neighbors:

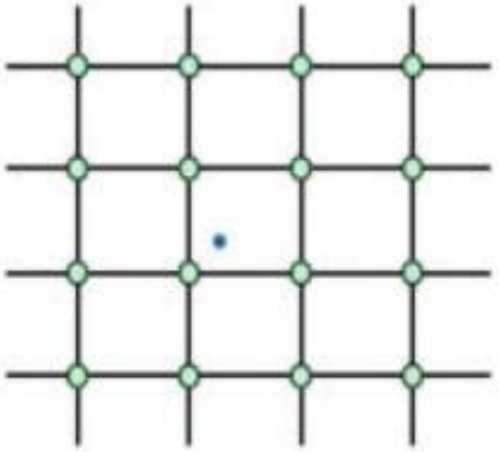
$$f(x_1, y_1) = ax_1 + by_1 + cx_1y_1 + d$$
$$f(x_2, y_2) = ax_2 + by_2 + cx_2y_2 + d$$
$$f(x_3, y_3) = ax_3 + by_3 + cx_3y_3 + d$$
$$f(x_4, y_4) = ax_4 + by_4 + cx_4y_4 + d$$

$$\begin{pmatrix} x_1 & y_1 & x_1y_1 & 1 \\ x_2 & y_2 & x_2y_2 & 1 \\ x_3 & y_3 & x_3y_3 & 1 \\ x_4 & y_4 & x_4y_4 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} f(x_1, y_1) \\ f(x_2, y_2) \\ f(x_3, y_3) \\ f(x_4, y_4) \end{pmatrix}$$

after estimation

$$f(x_d, y_d) = ax_d + by_d + cx_dy_d + d$$

Bicubic interpolation



$$f(x,y) = a_{00} + a_{01}y + a_{02}y^2 + a_{03}y^3 +$$
$$a_{10}x + a_{11}xy + a_{12}xy^2 + a_{13}xy^3 +$$
$$a_{20}x^2 + a_{21}x^2y + a_{22}x^2y^2 + a_{23}x^2y^3 +$$
$$a_{30}x^3 + a_{31}x^3y + a_{32}x^3y^2 + a_{33}x^3y^3$$

Use the 16 neighbors to estimate the 16 coefficients

Model:

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

coefficients to be estimated:

$a_{00}, a_{01}, \ldots a_{33}$

Reading Image

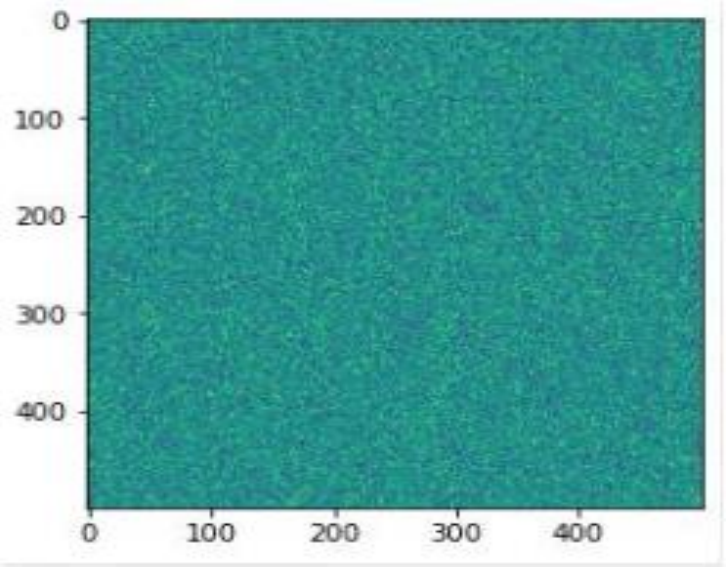
The io module is typically used for image input and output, such as reading and saving images.

```
from skimage import io
from matplotlib import pyplot as plt
my_image= io.imread("F:\\AfTer PHD\\Fares\\New folder\\58\\1.jpg")
plt.figure()
plt.imshow(my_image)
print(my_image)
```

```
[[[220 220 220]
  [220 220 220]
  [220 220 220]
  ...
  [204 204 204]
```

Reading Image

```
from skimage import io
import numpy as np
from matplotlib import pyplot as plt
random_image = np.random.random([500, 500])
plt.figure()
plt.imshow(random_image)
print(random_image)
```

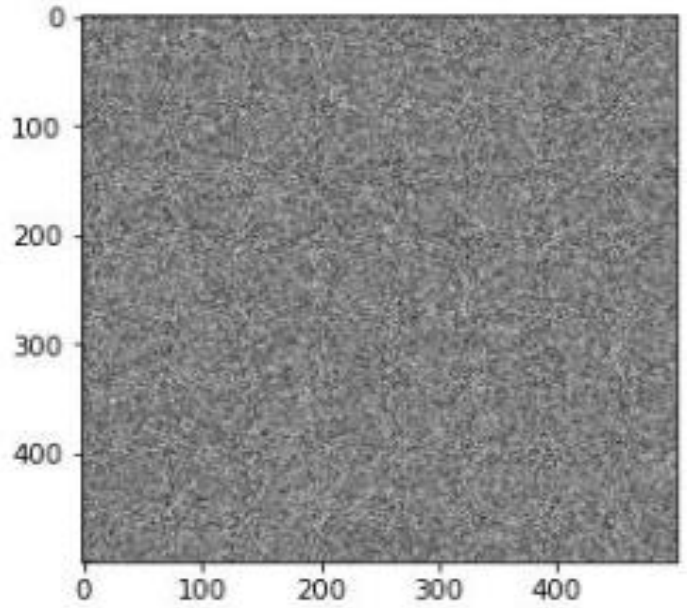



```
[[0.09348087 0.46405212 0.79277585 ... 0.57289124 0.3749102  0.76669397]
 [0.49754371 0.91491119 0.40190473 ... 0.59374764 0.28193718 0.95292175]
 [0.46321565 0.0103784  0.13611176 ... 0.3081001  0.5734807  0.86000102]
 ...
```

This line creates a 500x500 array of random values between 0.0 and 1.0 using NumPy.

Reading Image

```
from skimage import io
import numpy as np
from matplotlib import pyplot as plt
random_image = np.random.random([500, 500])
plt.figure()
plt.imshow(random_image,'gray')
print(random_image)
```




```
[[0.14993804 0.63492864 0.12913192 ... 0.43744068 0.12208103 0.05390321]
 [0.70377902 0.1857645  0.94281653 ... 0.59699908 0.62782933 0.03622855]
 [0.26969628 0.49568709 0.08286238 ... 0.67077313 0.72519898 0.94923826]
 ...
```

Reading Image

Data type	Range
uint8	0 to 255
uint16	0 to 65535
uint32	0 to 2^{32} - 1
float	-1 to 1 or 0 to 1
Int8	-128 to 127
int16	-32768 to 32767
int32	-2^{31} to 2^{31} - 1

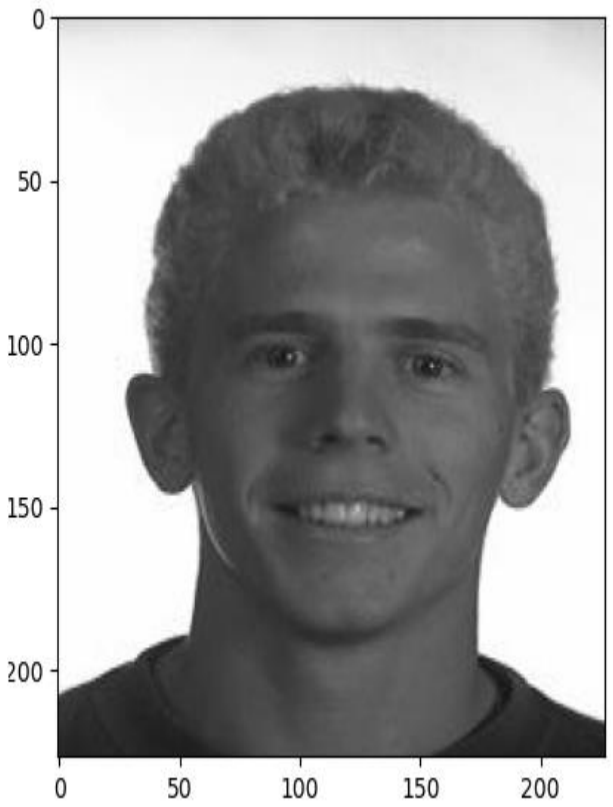
Reading Image

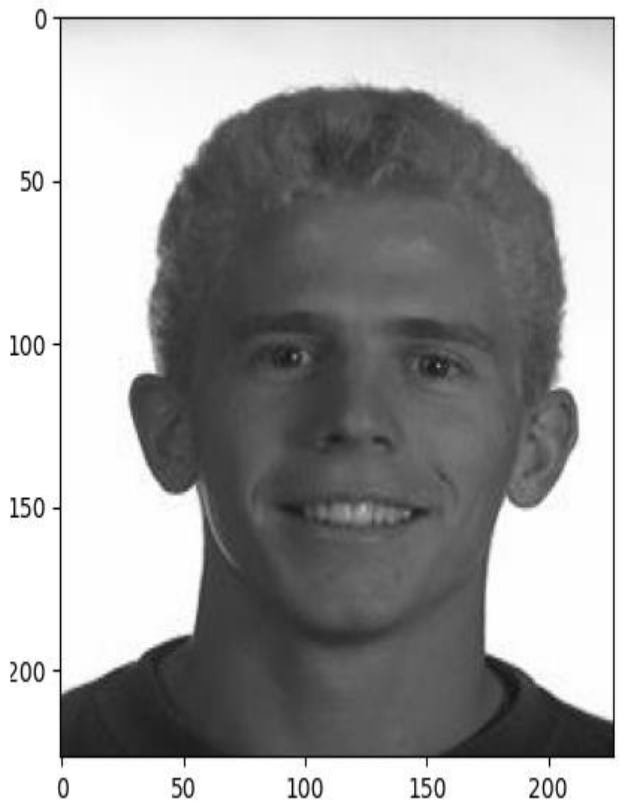
```
from skimage import io
from matplotlib import pyplot as plt
my_image= io.imread("F:\\AfTer PHD\\Fares\\New folder\\58\\1.jpg")
plt.figure()
plt.imshow(my_image)
from skimage import img_as_float, img_as_ubyte
my_float_image=img_as_float(my_image)
plt.figure()
plt.imshow(my_float_image)
img_ubyte=img_as_ubyte(my_float_image)
plt.figure()
plt.imshow(img_ubyte)
```

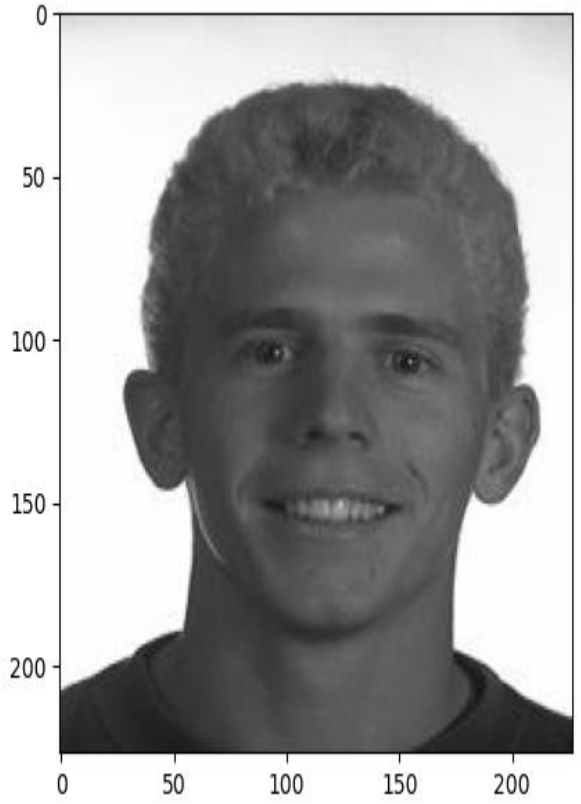
img_as_float: Converts an image to floating-point format with pixel values scaled between 0.0 and 1.0.

img_as_ubyte: Converts an image to unsigned 8-bit integer format (uint8), with pixel values between 0 and 255.

Reading Image



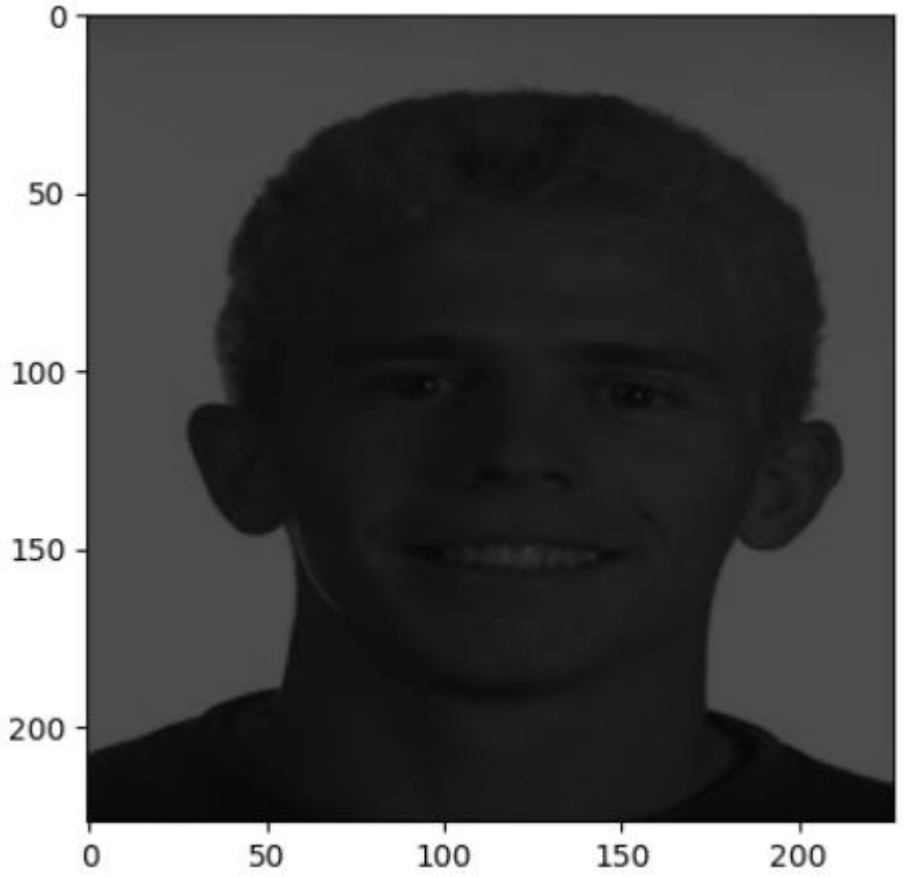




This is often useful for image processing tasks, like filtering or brightness adjustment, because floating-point math is more flexible.

Reading Image

```
dark_image=my_float_image*0.3
plt.figure()
plt.imshow(dark_image)
```



Reading Image

(height, width, channels)
height – number of rows, width – number of columns, channels – number of color channels

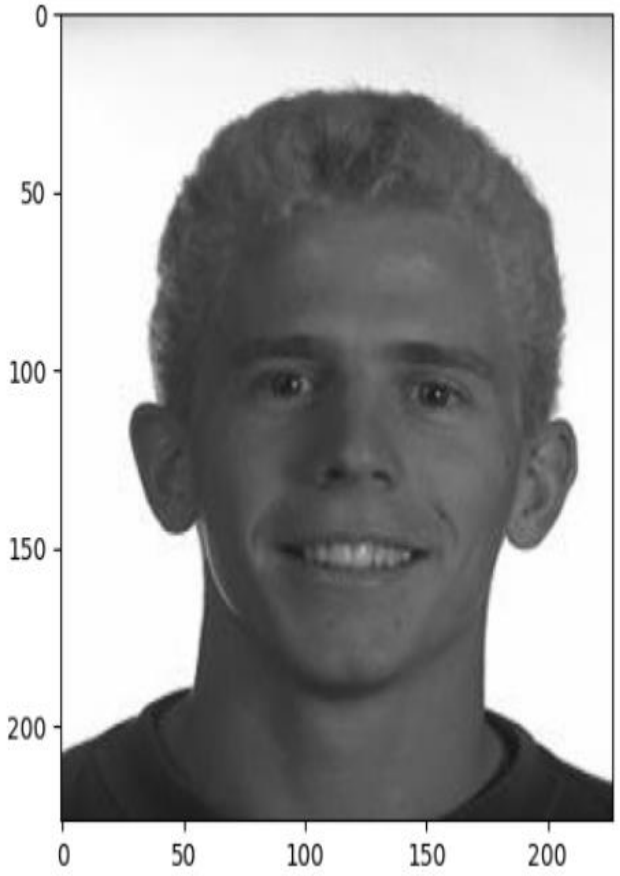
my_image [:50, :, :]
:50 → select the first 50 rows
: → select all columns
: → select all color channels

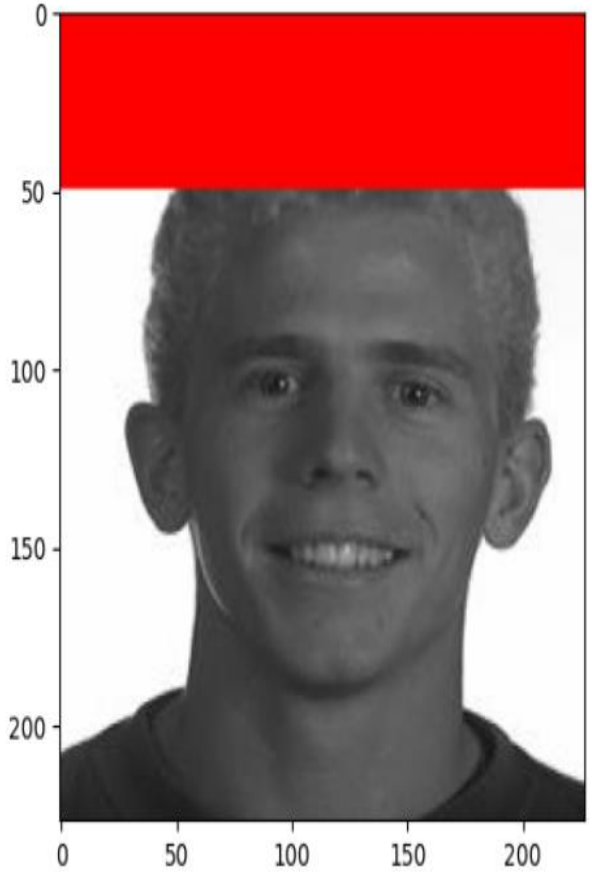
```
from skimage import io
from matplotlib import pyplot as plt
my_image= io.imread("F:\\AfTer PHD\\Fares\\New folder\\58\\1.jpg")
plt.figure()
plt.imshow(my_image)
my_image[ : 50, : , : ]=[255,0,0]
plt.figure()
plt.imshow(my_image)
my_image[my_image.shape[0]-50:,:,:]=[255,0,0]
plt.figure()
plt.imshow(my_image)
```

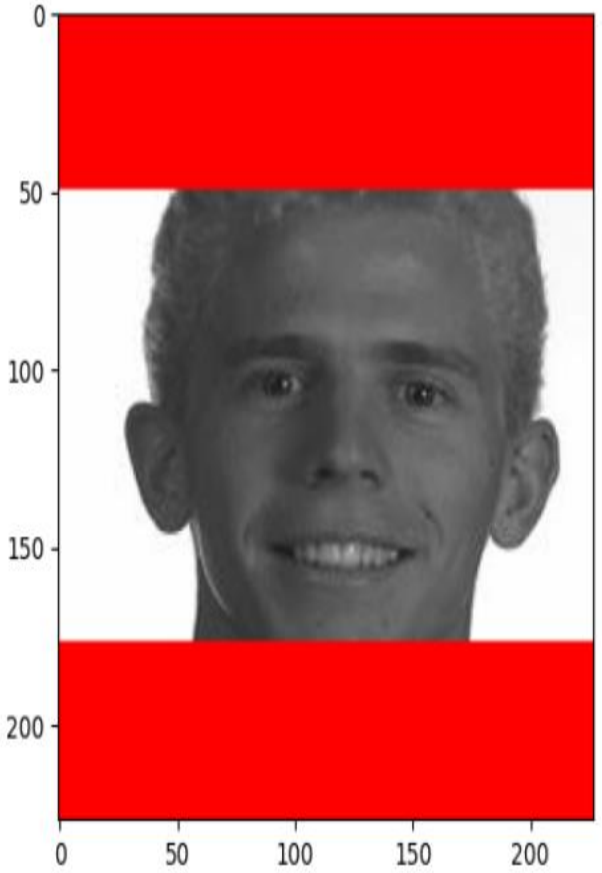
my_image [:50, :, :] = [255, 0, 0]
This modifies the top 50 rows of the image. It sets all pixel values in those rows to [255, 0, 0], which is pure red. So, the top strip of the image becomes red.

This modifies the bottom 50 rows of the image.
my_image.shape [0] - 50: targets the last 50 rows.
For example: my_image.shape [0] = 1000
Start from row 950 to the end

Reading Image







Reading Image

```
my_image[:,:50,:]=[0,0,255]
plt.figure()
plt.imshow(my_image)
my_image[:,my_image.shape[1]-50:,:]=[0,0,255]
plt.figure()
plt.imshow(my_image)
```

my_image[:, :50, :] = [0, 0, 255]
This modifies the left 50 columns of the image.
[0, 0, 255] → sets the pixel values to blue

my_image [:, my_image.shape [1]-50:, :] = [0, 0, 255]
Modifies the right 50 columns of the image.

Reading Image