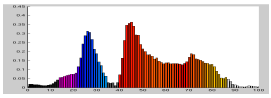


Dr. Qadri Hamarsheh

Digital Image Processing (750474)

Lecture 3

Outline of the Lecture

- **Image Processing Toolbox (IPT--Matlab).**
- **Reading Images.**
- **Displaying Images**

Image Processing Toolbox (IPT) (Matlab)

- **IPT** is a collection of functions that extend the capability of the Matlab numeric computing environment and support a wide range of image processing including:
  - Spatial image transformation.
  - Morphological operations.
  - Neighborhood and block operations
  - Linear filtering and filter design.
  - Transform.
  - Image enhancement and analysis.
  - Deblurring.
  - Region of interest operations.

Coordinate Conventions:

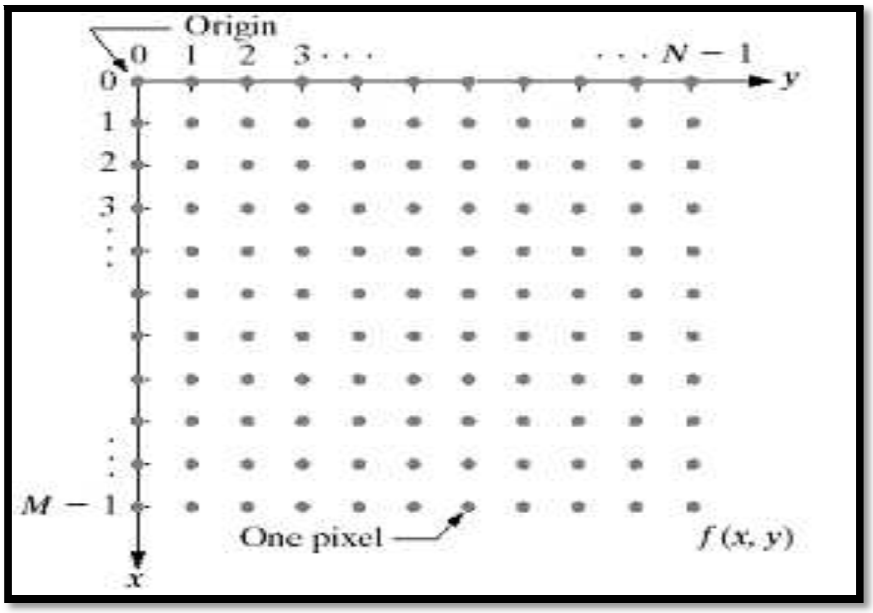
- The result of **sampling** and **quantization** is a matrix of real numbers; two ways are used to represent the digital image:
  - **First convention**: this convention is frequently used in **image processing books**, in which,
    - ✓ The image origin is defined to be at $(x, y) = (0,0)$, the next coordinate value along the first row of the image is $(x, y) = (0,1)$.
    - ✓ x ranges from 0 to $M - 1$ and y ranges from 0 to $N - 1$ in integer increment.
  - **Second convention**: this convention is used in **IPT Matlab toolbox**, the notation (r, c) to indicate rows and columns.
    - ✓ The origin of the coordinate system is at $(r, c) = (1,1)$.
    - ✓ r ranges from 1 to M and c ranges from 1 to N in integer increment.
- IPT documentation refers to the coordinates as pixel coordinates or spatial coordinates.

Matrix notation of the digital image:

- Representations of digital image functions:

1. Image processing books matrix representation.

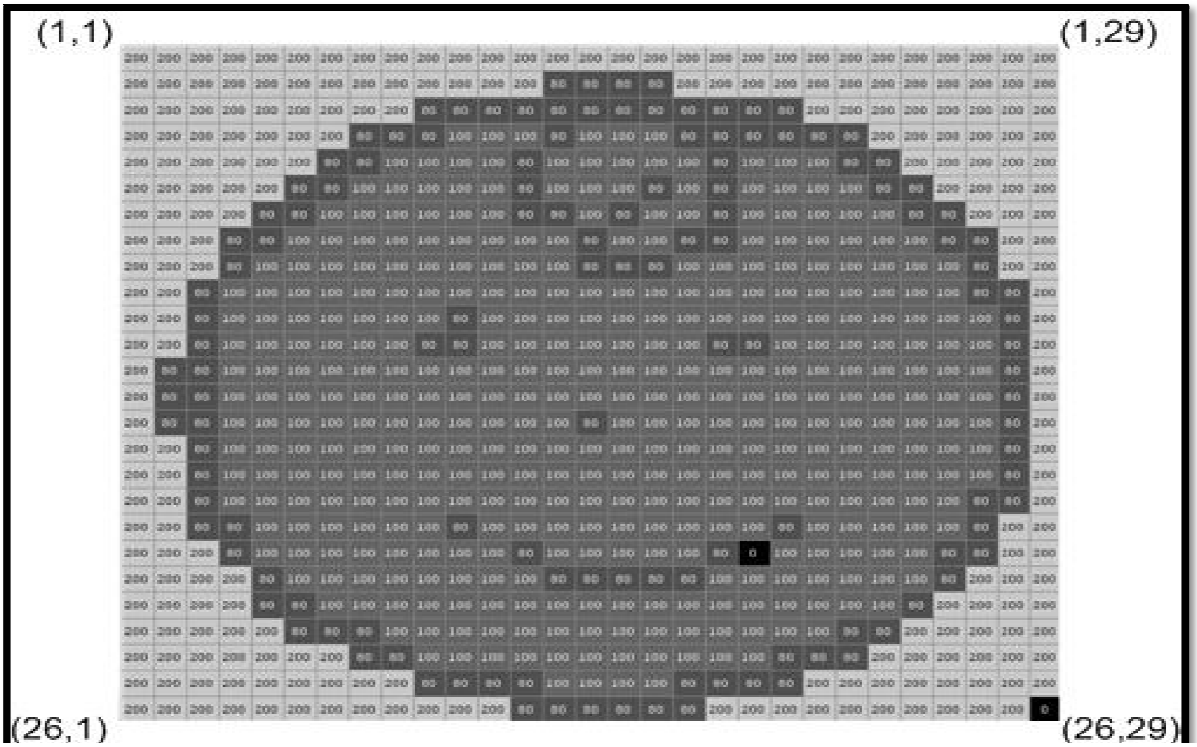
$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix}$$

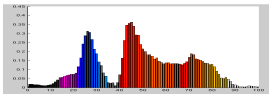


2. Matlab matrix representation

$$f(r,c) = \begin{bmatrix} f(1,1) & f(1,2) & \dots & f(1,N) \\ f(2,1) & f(2,2) & \dots & f(2,N) \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ f(M,1) & f(M,2) & \dots & f(M,N) \end{bmatrix}$$

- The two representations are **equal** except for the **shift** in origin.
- The notation $f(p,q)$ denotes the element located in row p and column q.
- $1 \times N$ matrix is called a **`row vector`**.
- $M \times 1$ matrix is called a **`column vector`**.
- 1×1 matrix is **`scalar`**.





Reading Images

- To read an image, use the **imread** command, whose syntax is:

I = imread ('filename')

- This command **reads** and **stores** the image in an **array** I.

'filename'- string containing the **complete name (path)** of the image file including extension.

Example:

```
>> f= imread('chestxray.jpg');
```

reads the jpg image "**chestxray**' into image array.

```
>> I= imread(pout.tif');
```

reads and stores the image in an array I.

whos command:

- whos command is used to get information about **variable** in the **workspace**.

```
>> whos
```

Name	Size	Bytes	Class	Attributes
I	291x240	69840	uint8	---------

size command:

- Function size gives the row and column **dimensions** of an image:

```
>> size(f);
        ans =
        1024 1024
>> [M , N] = size(f);
```

- This syntax returns the number of rows (M) and the number of columns (N) in the image.
- The following table summarizes some ways to get information about an image. These are not specific to the Image Processing Toolbox.

Command	Description
whos	To get information about size, type, and bytes, of all variables.
whos I	For information about an image stored in I.
size(I)	To get the size of the image stored in I.
class(I)	To get type of data stored in I

Displaying Images

- To display an image, the **imshow** function is used, which has the basic syntax:

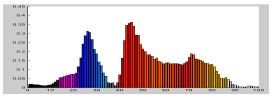
a) imshow (f,G)

f- an image, G- the **number of intensity levels** used to display the image, if **G** is omitted, it defaults to **256** levels.

b) imshow (f, [low high])

- ✓ Displays as **black** all values **less than or equal to low**.
- ✓ Displays as **white** all values **greater than or equal to high**.
- ✓ The values in between are displayed using the **default number of levels**.

c) imshow (f ,[ ])



- ✓ Sets variable **low** to the **minimum** value of array f and **high** to its **maximum** value. This form is useful for images with **low dynamic range**.

pixval command :

- Pixval is used to display the intensity values of individual pixel **interactively**: Moving the cursor, the coordinates of the cursor position and the corresponding intensity values are shown.
- Pixval displays **Red, Green, blue** components, when the image is **color**.
- If the left button on the mouse is **clicked** and then held pressed, pixval displays the **Euclidean distance** between the initial and current cursor location, the syntax is:

pixval

Example 1

```
>> f = imread ('rose_512.tif'); % read from disk an image
>> whos f % extract basic information about the image
>> imshow (f) % display the image.
```

- If another image, **g**, is displayed using imshow, Matlab replaces the image in the screen with the new image.

```
>> figure, imshow (g) % keep the first image and
                      %output a second image.
>> imshow (f), imshow (g) % display both images
```

Example 2

- Suppose that we read an image **h** and find that using imshow (h) produces the image that has a **low dynamic range**, to correct:

```
>> Imshow (h, [  ]) % improve the image h.
```

- There are a series of photos that come as part of the image processing toolkit. To get the list of images and credits, you can type:

```
>> help imdemos
```

- If you want to view any of these photos, you can use the **imshow**, which opens a separate window displaying the image. For instance:

```
>> imshow('football.jpg');
>> imshow('coins.png');
>> imshow('autumn.tif');
>> imshow('board.tif');
```