

## Session 1

# Introduction to Digital Image Processing and MATLAB

## OBJECTIVES:

The objective of this lab session is

- Introduction of MATLAB
  - MATLAB workspace, Command Window, Variable panel
  - Working with variables, vectors and function
  - Working with library functions.
- Image processing with MATLAB
  - How to read an image in Matlab.
  - How to show an image in Matlab.
  - How to access Image Pixels in Matlab.
  - How to write Image in Matlab.
  - Mirror Image generation.
  - Flipped Image generation.

## WHAT IS MATLAB?:

- MATLAB = Matrix Laboratory
- “MATLAB is a high-level language and interactive environment that enables you to perform computationally intensive tasks faster than with traditional programming languages such as C, C++ and Fortran.”(www.mathworks.com)
- MATLAB is an interactive, interpreted language that is designed for fast numerical matrix calculations

## THE MATLAB ENVIRONMENT:

The Figure 1.1 shows the MATLAB window components:

**Workspace** Displays all the defined variables

**Command Window** To execute commands in the MATLAB environment

**Command History** Displays record of the commands used

**File Editor Window** Define your functions

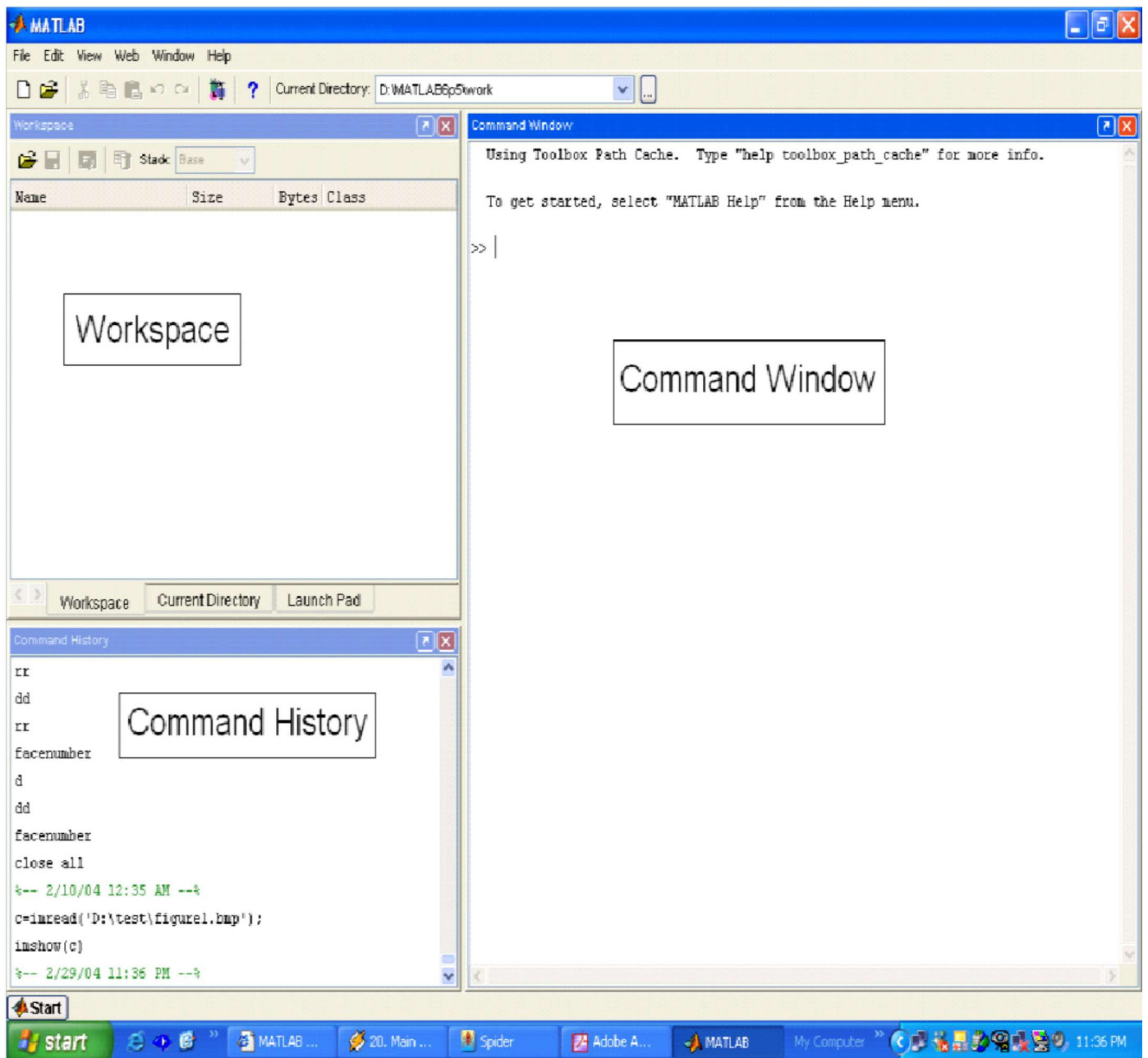
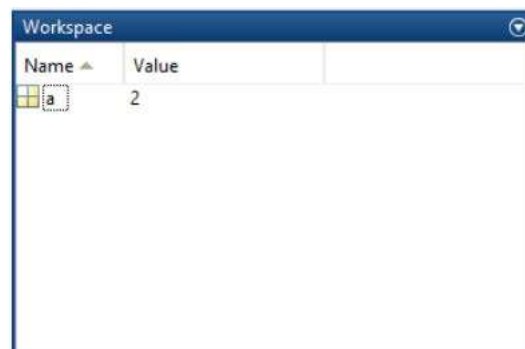
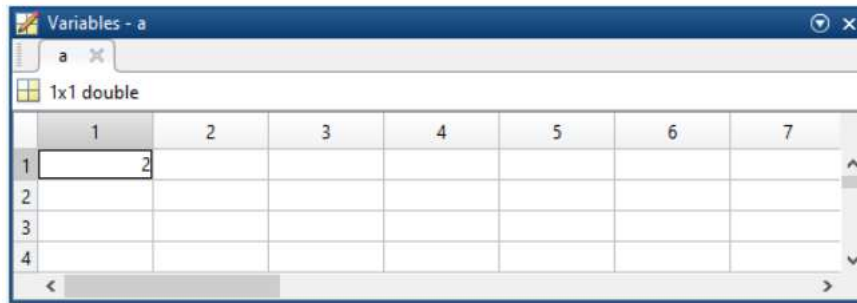


Figure 1.1: MATLAB Environment



(a)

Figure 1.2: Workspace



(b)

Figure 1.3: Variables

## WORKSPACE AND VARIABLES: :

Let's start with variable,  $a = 2$ . After entering, the next line will display the variable with the assigned value.

$$a = 2$$

And you can see that, the variable is appeared in the workplace (as in Figure 1.2). If you double click on that variable icon, a panel named 'Variables' will popup (as in Figure 1.3).

MATLAB stores everything as matrix. You can note the indication of  $1 \times 1$  double in the variable panel. It mean  $a$  is a 1 by 1 matrix (eventually a matrix with only one element), which is double type. By default any numerical data is double type. There are other data types as well such as `uint8`(unsigned integer 8 bit), `char` (character), `logical` (boolean).

## WORKING WITH VECTOR AND MATRIX:

In MATLAB, the matrix element indices start from the top left corner. As we traverse right, we go through each column. And as we traverse down, we go through each row.

### How to build a matrix?

$$A = [1 \ 2 \ 3; \ 4 \ 5 \ 6; \ 7 \ 8 \ 9];$$

Creates matrix  $A$  of size  $3 \times 3$

**Special matrices** `zeros(n,m)`, `ones(n,m)`, `rand()`

**Basic Operations on Matrices** All operators in MATLAB are defined on matrices: `+`, `-`, `*`, `/`, `^`, `sqrt`, `sin`, `cos`, etc.

Element-wise operators are defined with a preceding dot: `.*`, `./`, `.^`

**Matrix Functions** `size(A)` – size vector, `sum(A)` – columns sums vector, `sum(sum(A))` – sum of all the elements.

**Accessing elements of a matrix** To access we use the following format of the command: `Matrix(which_row, which_column)`.

### Assigning elements of a matrix

$$M(2,2) = 99$$

$$M =$$

1	2	3	4
5	99	7	8
9	1	2	3

**Question 1.** Now, can you generate a matrix with random numbers greater than or equal to 1 using one single line of command?

You can apply element wise operations. To perform that, we need to put a dot(.) in front of the operator. For example, the following command will do element wise multiplication, that means 1st element of A will be multiplied with the 1st element of B, 2nd of A will be multiplied with 2nd element of B, and so on.

$$C = A .* B$$

$$C =$$

9	16	21
24	25	24
21	16	9

**Question 2.** Now, can you square every element of the matrix A using one single line of command?

**Question 3.** Also, can you square each elements of both A and B, add the squared elements of A with the squared elements B and store them in the matrix C? (Again, using one single line of command)

## HOW TO DISPLAY AN IMAGE? :

To display image, use the `imshow` function.

**Syntax** `imshow(A)`

**Description** `imshow(A)` displays the image stored in array A.

## HOW TO WRITE AN IMAGE DATA?:

Use `imwrite` to write image to graphics file

**Syntax** `imwrite(A, filename, fmt)`

**Example** Program-code 1.1 shows how to write an image file. Figure 1.4 shows the output.

```
1 a=imread('pout.tif');
2 imwrite(a,gray(256),'b.bmp');
3 imshow('b.bmp')% imshow is used to display image
```

Program 1.1: Writing image to disk

## ACCESSING THE PIXEL DATA:

There is a one-to-one correspondence between pixel coordinates and the coordinates MATLAB® uses for matrix subscripting. This correspondence makes the relationship between an image's data matrix and the way the image is displayed easy to understand. For example, the data for the pixel in the fifth row, second column is stored in the matrix element (5,2). You use normal MATLAB matrix subscripting to access values of individual pixels. For example, the MATLAB code

$$A(2,15)$$

returns the value of the pixel at row 2, column 15 of the image A.

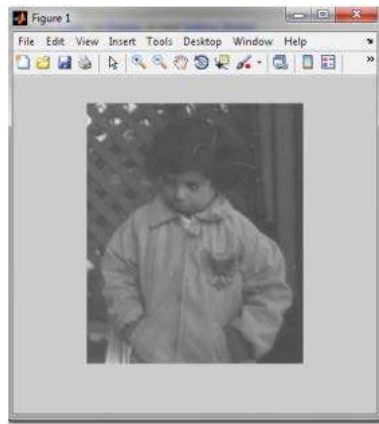


Figure 1.4: Writing image to disk file

## MIRROR IMAGE GENERATION:

```

1 % this program produces mirror image of the image passed to it n also
2 % displays both the original and mirror image
3 a=imread('pout.tif');
4 [r,c]=size(a);
5 for i=1:1:r
6     k=1;
7     for j=c:-1:1
8         temp=a(i,k);
9         result(i,k)=a(i,j);
10        result(i,j)=temp;
11        k=k+1;
12    end
13 end
14 subplot(1,2,1),imshow(a)
15 subplot(1,2,2),imshow(result)

```

Program 1.2: Mirror Image Generation



Figure 1.5: Source Image and Mirrored Output

**Task 1.** Write a MATLAB code that reads a gray scale image and generates the flipped image of original image. Your output should be like the one given in Figure 1.6.

**Task 2.** Write a MATLAB code that will do the following:



Figure 1.6: Source Image with Desired Output

1. Read any gray scale image.
2. Display that image.
3. Again display the image such that the pixels having intensity values below than 50 will display as black and pixels having intensity values above than 150 will display as white. And the pixels between these will display as it is.

## REFERENCES:

- [https://www.mathworks.com/help/matlab/learn\\_matlab/desktop.html](https://www.mathworks.com/help/matlab/learn_matlab/desktop.html)
- <https://www.mathworks.com/help/matlab/elementary-math.html>
- [https://www.mathworks.com/help/matlab/learn\\_matlab/matrices-and-arrays.html](https://www.mathworks.com/help/matlab/learn_matlab/matrices-and-arrays.html)
- [https://www.mathworks.com/help/matlab/functionlist.html?s\\_cid=doc\\_ftr](https://www.mathworks.com/help/matlab/functionlist.html?s_cid=doc_ftr)
- <https://www.mathworks.com/help/matlab/2-and-3d-plots.html>