Introduction to MATLAB

Introduction to MATLAB

- MATLAB is a high-performance technical computing environment developed by MathWorks Inc. in 1984.
- Capabilities: numerical calculations, matrix manipulations (MATLAB=MATrix LABoratory), data analysis, data visualization, simulations, analytical calculations,…

Why MATLAB ?

- Quick to learn, and good documentation
- Many existing code, toolboxes, built-in functions available.
- Excellent display capabilities
- Easy to do very rapid prototyping
- Widely used for teaching and research in universities and industry
- Weakness: MATLAB is an interpreted language, slower than C/C++.

Why MATLAB for Image Processing

- A digital image is a matrix!
- Well supported by toolboxes.
- A good choice for image processing development

Some Facts on MATLAB

- Everything in MATLAB is a matrix! (A single number is really a 1x1 matrix)
- MATLAB is an interpreted language, no compilation needed
- MATLAB does not need any variable declarations, no dimension statements, has no packaging, no storage allocation, no pointers
- Programs can be run step by step, with full access to all variables, functions, etc.
- Rows and columns are always numbered starting at 1

MATLAB Environment

- Command Window
  - Type commands
- Workspace
  - View program variables
  - Double click on a variable to see it in the Array Editor
- Command History
  - View past commands
  - Save a whole session using *diary*

Three Places to Write Code

- Prompt: write and press enter
- Script file
- Function file

Everything is a Matrix

```
a1=[1:0.5:3]

a2=[1 2 3; 4 5 6; 7 8 9]

for i = 1:2
        for j=1:4
                a3(i,j)=i*j;
        end
end
a3'  %transpose a matrix

a4=zeros(3,3)

a5=ones (2,3)

a6=[a4; a5]

a7=[a4 a5]
```

Access a Matrix Element(s)

a1(3) % index starts with 1

a2(1,3) % matrix(row number, column number)

a2(2,:) % access a matrix row (2nd row)

a2(:,3) % access a matrix column (3rd column)

a2(2:3, 2:3)

Manipulate Matrices

```
a=magic(3)
i=find(a>3)
[i,j]=find(A>3)
[i,j,v]=find(A>3)

A=[1 2 3; 4 5 6; 7 8 9]
B=zeros(3); B(2:3,2:3)=1
A*B
A.*B
A=[1 2; 3 4]; B=[1 1; 1 -1]
A/B   % A*inv(B)
A./B

sum(A.*B)
```

MATLAB Code for Image Manipulation

```
% Read in an image
im1= imread('image  file name');

% image is of class UINT8
class(im1)

% Display the image
imshow(im1)

% Get the size of an image
imSize=size(im1);

% Convert the rgb image to a grayscale image
grIm1=rgb2gray(im1);

% See the histogram of the image
imhist(grIm1)
```

MATLAB Code - Threshold an Image

```
% Threshold an image
Threshold =100;
tIm1=grIm1>100; imshow(tIm1)

%note, the arithmetic operations are not defined over uint8 variables so first convert them into double
tIm2=double(grIm1>100).*double(grIm1);

% this does not work, why?
imshow(tIm2)

% the imshow() assumes the doubles in an image to be between 0 and 1. If it is not, it chops the intensities outside this range. So, cast the result to uint8 before displaying
tIm2=uint8(tIm2);
imshow(tIm2)

% Let's save the thresholded image in a file
imwrite(tIm2, 'thresholdedImg,jpg');
```

Script Files (.m)

- Sequences of MATLAB commands can be written to m-files.
- Enter the name of the file (without extension) will automatically execute all statements
- Use the Editor/Debugger to edit, run

Function Files

- Functions are like any other m-file, but they take input arguments and/or return output arguments. Used for replacing repetitive portions of the codes.
- It is always recommended to name function file the same as the function name
- See myfunction.m
- Call it with f=myfunction(x,y)

Operators

Arithmetic operators:

+, -, *, /, ^, .*, ./, .^

Logical operators:

==, ~=, <, <=, >, >=, &, |, ~

Getting Help

- Use the Help Browser
- Type
  - help
  - help function
  - lookfor keyword (if you don't know the exact name of the function)
- Run demos
  - Type demos
  - Type help demos