IMAGE PROCESSING I

IntuoN Lertrusdachakul

Tutorial I: Introduction to MATLAB

Outline

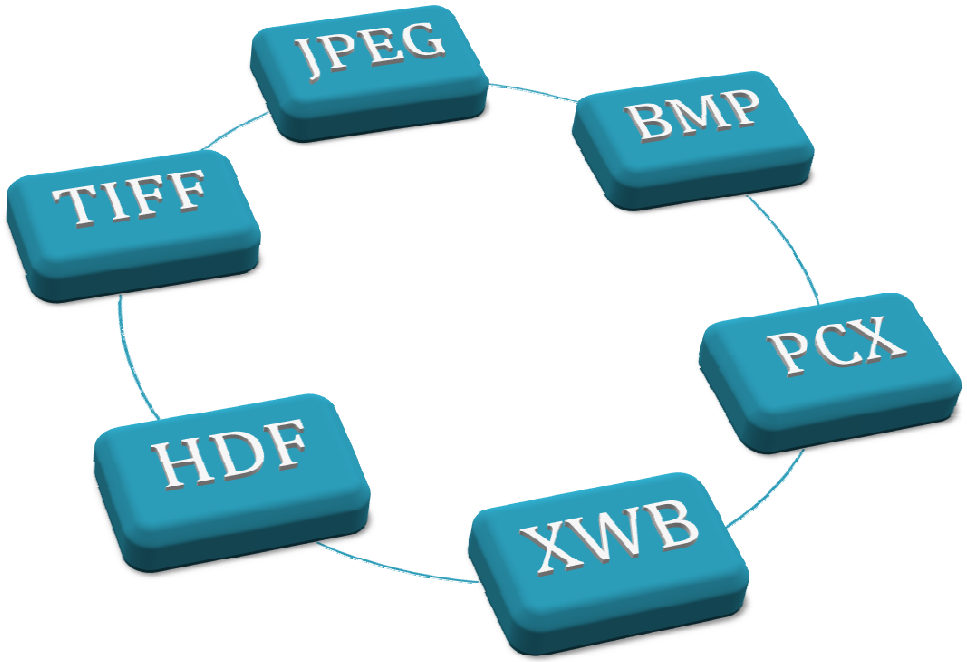
- Intro
- Getting Started
- Basic Operation in DIP
- Graphic Application
- Learning MATLAB by Example
- Assignments
- Further Information

Intro…

- MATLAB is derived from MATrix LABoratory
- MATLAB is an interactive, matrix-based system for scientific and engineering numerical computation and visualization.
- Matlab also features a family of application-specific solution called toolboxes (including Image Processing Toolbox)
- Powerful, fast, and reliable

Getting Started

- Image file formats



Getting Started

- Image Representation

Binary Image	• represented by an MxN logical matrix where pixel values are either 0(black) or 1 (white)
Grayscale Image	•or intensity image represented as a matrix of double data type of size MxN. Element values denote the pixel grayscale intensities as a number with decimals between 0 and 1 to each pixel.
RGB Image	•or truecolor image represented as a three-dimensional MxNx3 double matrix. Each pixel has red, green, blue components along the third dimension values in [0,1]
Indexed Image	•represented with an index matrix of size MxN and a colormap matrix of size Kx3. The colormap holds all colors used in the image and the index matrix represents the pixels by referring to colors in the colormap.
8-bit Images	•or uint8 uses less memory and some operations compute faster than with double type

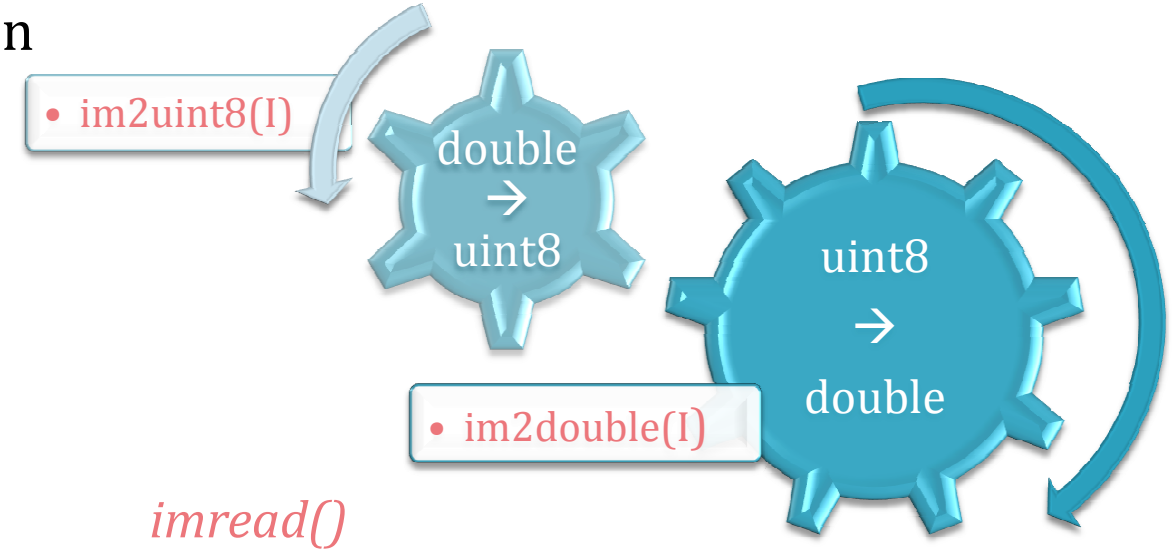
Getting Started (II)

- Image Format Conversion

RGB format → Intensity format	*rgb2gray()*
RGB format → Indexed format	*rgb2ind()*
Indexed format → RGB format	*ind2rgb()*
Indexed format → Intensity format	*ind2gray()*
Intensity format → Indexed format	*gray2ind()*
Regular Matrix → intensity format by scaling	*mat2gray()*
Intensity/indexed/RGB format → Binary format	*dither()*

Getting Started (III)

- Image Type Conversion



- Image I/O
  - Read an Image *imread()*
  - Write an Image to a file *imwrite(,)*
- Loading and Saving Variables
  - Load variable X *load x*
  - Save variable X *save x*

Getting Started (IV)

- Image Display
  - Set the default colormap *colormap*
  - Or for 8-bit grayscale images *gray(256)*
  - Display image *imshow*
  - Force window to make screen pixel the same as one image pixel *truesize*
  - Display several images by creating multiple figures *figure*
  - Or putting multiple images in the same figure *subplot*

Basic Operations

- Statistics

```
uMax = max(u(:));          % Compute the maximum value
uMin = min(u(:));          % Compute the minimum value
uAvg = mean(u(:));         % Compute the average value
uVar = var(u(:));          % Compute the variance value
uMed = median(u(:));       % Compute the median value
hist(u(:);                 % Plot histogram
```

Basic Operation (II)

- Basic manipulations

```
uClip = min(max(u,0),1);                % Clip elements to [0,1]
uPad = u([1,1:end,end],[1,1:end,end]);  % Pad image with one-pixel margin
uPad = padarray(u,[k,k],'replicate');   % Pad image with k-pixel margin
uCrop = u(RowStart:RowEnd,ColStart:ColEnd); % Crop image
uFlip = flipud(u);                      % Flip in the up/down direction
uFlip = fliplr(u);                      % Flip left/right
```

Basic Operation (III)

- Interpolate image
  uResize = imresize(u,ScaleFactor);

- Rotate by k*90 degrees with integer k
  uRot = rot90(u,k);

- Rotate by Angle degrees
  uRot = imrotate(u,Angle);

- Stretch contrast to [0,1]
  uc = (u - min(u(:))/(max(u(:)) - min(u(:)));

- Quantize to K graylevels {0,1/K,2/K,...,1}
  uq = round(u*(K-1))/(K-1);

Basic Operation (IV)

- Simulating noise
  - Add white Gaussian noise of standard deviation sigma

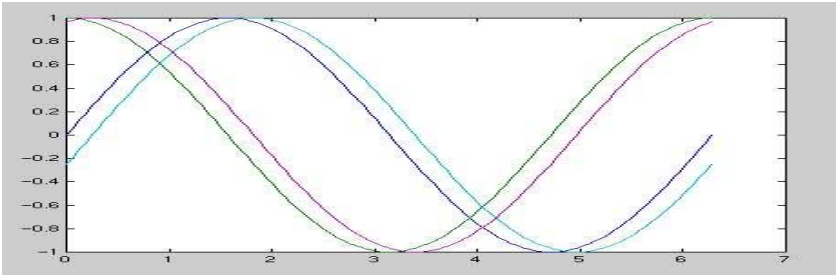
```
uNoisy = u + randn(size(u))*sigma;
```

  - Salt and pepper noise

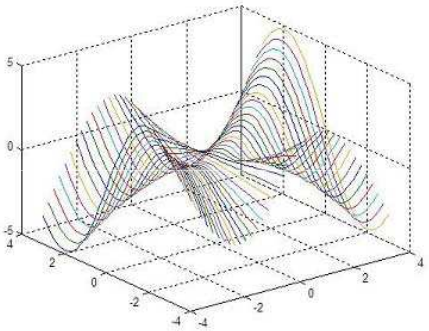
```
uNoisy = u;
uNoisy(rand(size(u)) < p) = round(rand(size(u)));
```

Graphic Applications

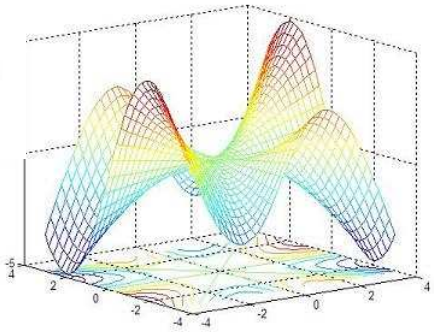
- Planar curves



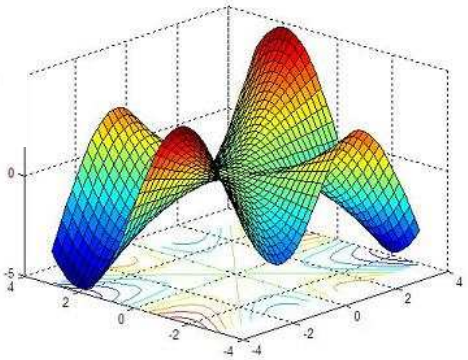
- 3-D Plots



- 3-D mesh plots

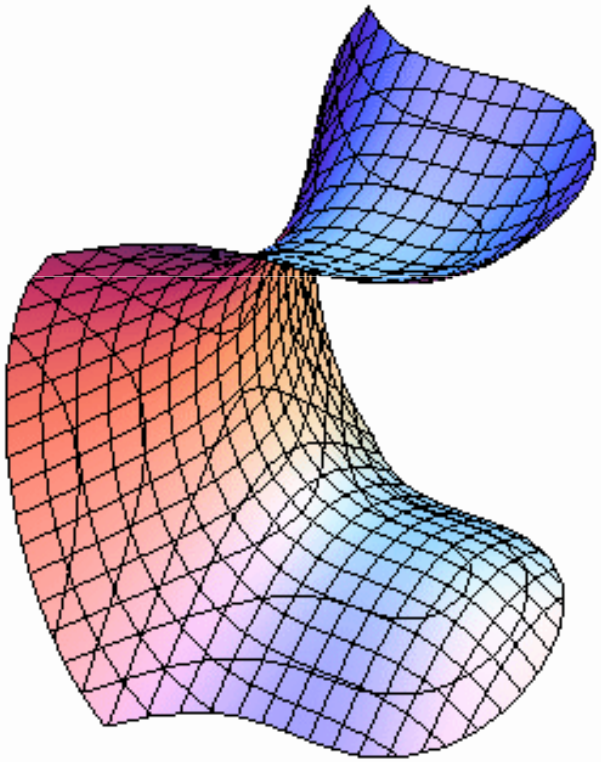


- 3-D surface plots



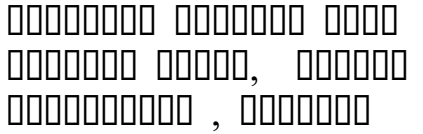
Graphic Applications (II)

$$x^3 + y^2 - z^2 = -2.$$



Learning MATLAB by Example


```
%-------------------------------------------------------------------------
% [1] Observe the results from the following example
%-------------------------------------------------------------------------
% Create Variables (scalars, vectors, and matrices are possible)
N = 5                                % a scalar
v = [1 0 0]                          % a row vector
v = [1;2;3]                          % a column vector
v = v'                               % transpose a vector
v = [1:.5:3]                         % a vector in a specified range:
v = pi*[-4:4]/4                      % [start: stepsize: end]
v = []                               % empty vector
m = [1 2 3; 4 5 6]                   % a matrix: 1st parameter is ROWS, and
                                     % 2nd parameter is COLS
m = zeros(2,3)                       % a matrix of zeros
v = ones(1,3)                        % a matrix of ones
m = eye(3)                           % identity matrix
v = rand(3,1)                        % random matrix (see also randn)
v = [1 2 3];                         % access a vector element
v(3)                                 % vector(number)
m = [1 2 3; 4 5 6]
m(1,3)                               % access a matrix element
matrix(rownumber, columnnumber)
m(2,:)                               % access a matrix row (2nd row)
m(:,1)                               % access a matrix column (1st row)
size(m)                              % size of a matrix
size(m,1)                            % number of rows
size(m,2)                            % number of columns
m1 = zeros(size(m))                  % create new zeros matrix with size of m
who                                  % list of variables
whos                                 % list/size/type of variables
```



Learning MATLAB by Example (II)


```
%-------------------------------------------------------------------------
% [2] Simple operations on vectors and matrices
%-------------------------------------------------------------------------
% (A) Pointwise (element by element) Operations:
% addition of vectors/matrices and multiplication by a scalar are done
% "element by element"

a= [1 2 3 4];                          % vector
2 * a                                  % scalar multiplication
a / 4                                  % scalar division
b = [5 6 7 8];                         % vector
a + b                                  % pointwise vector addition
a - b                                  % pointwise vector addition
a .^ 2                                 % pointise vector squaring (note .)
a .* b                                 % pointwise vector multiply (note .)
a ./ b                                 % pointwise vector division (note .)
log( [1 2 3 4] )                       % pointwise arithmetic operation
round( [1.5 2; 2.2 3.1] )              % pointwise arithmetic operation
```

Learning MATLAB by Example (III)


```
% (B) Vector Operations (no 'for loops' needed)
% Built-in MATLAB functions operate on vectors, if a matrix is given,
% then the function operates on each column of the matrix

a = [1 4 6 3]                    % vector
sum(a)                           % sum of vector elements
mean(a)                          % mean of vector elements
var(a)                           % variance
std(a)                           % standard deviation
max(a)                           % maximum
a = [1 2 3; 4 5 6]               % matrix
mean(a)                          % mean of each column
max(a)                           % max of each column
max(max(a))                      % to obtain max of matrix
max(a(:))                        % or...
```

Learning MATLAB by Example (IV)


```
% (C) Matrix Operations:

[1 2 3] * [4 5 6]'                    % row vector 1x3 times column vector
% 3x1 results in single number, also known as dot product or inner product
[1 2 3]' * [4 5 6] % column vector 3x1 times row vector 1x3 results in 3x3
% matrix, also known as outer product
a = rand(3,2)                         % 3x2 matrix
b = rand(2,4)                         % 2x4 matrix
c = a * b                             % 3x4 matrix
a = [1 2; 3 4; 5 6]                   % 3x2 matrix
b = [5 6 7];                          % 3x1 vector
b * a                                 % matrix multiply
a' * b'                               % matrix multiply
```

Learning MATLAB by Example (V)


```
%-------------------------------------------------------------------------
%(3) Saving
%-------------------------------------------------------------------------
save mysession                          % creates session.mat with all variables
save mysession a b                      % save only variables a and b
clear all                               % clear all variables
clear a b                               % clear variables a and b
load mysession                          % load session
a
b
```

Learning MATLAB by Example (VI)


```
%-------------------------------------------------------------------------
%(4) Relations and control statements
% Example: given a vector v, create a new vector with values equal to v
% if they are greater than 0, and equal to 0 if they less than or equal to 0.
%-------------------------------------------------------------------------
v = [3 5 -2 5 -1 0]                    % 1: FOR LOOPS
u = zeros( size(v) );                  % initialize
for i = 1:size(v,2)
if( v(i) > 0 )
u(i) = v(i);
end
end
u
v = [3 5 -2 5 -1 0]                    % 2: NO FOR LOOPS
u2 = zeros( size(v) );                 % initialize
ind = find( v>0 )                      % index into >0 elements
u2(ind) = v( ind )
```

Learning MATLAB by Example (VII)


```
%-------------------------------------------------------------------------
%(5) Creating functions using m-files:
% Functions in MATLAB are written in m-files. Create a file called 'thres.m'
% In this file put the following:
%-------------------------------------------------------------------------
function res = thres( v )
u = zeros( size(v) );                % initialize
ind = find( v>0 )                    % index into >0 elements
u(ind) = v( ind )
v = [3 5 -2 5 -1 0]
thres( v )                           % call from command line
```

Learning MATLAB by Example (VIII)


```
%-------------------------------------------------------------------------
%(6) Plotting
%-------------------------------------------------------------------------
x = [0 1 2 3 4];                    % basic plotting
plot( x );
plot( x, 2*x );
axis( [0 8 0 8] );
x = pi*[-24:24]/24;
plot( x, sin(x) );
xlabel( 'radians' );
ylabel( 'sin value' );
title( 'dummy' );
gtext( 'put cursor where you want text and press mouse' );
Tutorial 1: Introduction to MATLAB Page 8 of 12 10/07/2003
figure;                             % multiple functions in separate graphs
subplot( 1,2,1 );
plot( x, sin(x) );
axis square;
subplot( 1,2,2 );
plot( x, 2.*cos(x) );
axis square;
figure;                             % multiple functions in single graph
plot( x,sin(x) );
hold on;
plot (x, 2.*cos(x), '--' );
legend( 'sin', 'cos' );
hold off;
figure;                             % matrices as images
m = rand(64,64);
imagesc(m)
colormap gray;
axis image
axis off;
```

Learning MATLAB by Example (IX)


```
%-------------------------------------------------------------------------
%(7) Working with Images
%-------------------------------------------------------------------------
[I,map]=imread('trees.tif');          % read a TIFF image
figure, imshow(I,map)                 % display it as indexed image
I2=ind2gray(I,map);                   % convert it to grayscale
figure
imagesc(I2,[0 1])                     % scale data to use full colormap
                                      %for values between 0 and 1
colormap('gray')                      % use gray colormap
axis('image')                         % make displayed aspect ratio
                                      % proportional to image dimensions
I=imread('photo.jpg');                % read a JPEG image into 3D %array
figure
imshow(I)
rect=getrect;                         % select rectangle
I2=imcrop(I,rect);                    % crop
I2=rgb2gray(I2);                      % convert cropped image to grayscale
imagesc(I2)                           % scale data to use full colormap
                                      % between min and max values in I2
colormap('gray')
colorbar                              % turn on color bar
pixval                                % display pixel values interactively
truesize                              % display at resolution of one screen pixel
                                      % per image pixel
truesize(2*size(I2))                  % display at resolution of two screen pixels
                                      % per image pixel
I3=imresize(I2,0.5,'bil');            % resize by 50% using bilinear interpolation
I3=imrotate(I2,45,'bil','same');      % rotate 45 degrees and crop to original size
I3=double(I2);                        % convert from uint8 to double, to allow
                                      % math operations
imagesc(I3.^2)                        % display squared image (pixel-wise)
imagesc(log(I3))                      % display log of image
```

Exercise

- Download an image from internet and save file as img_tut1.jpg
- Load the image and Store it as the Variable I
- Check the size and class of all stored variables
- Save the Variable I
- List the files in your directory

Assignment

- Obtain a digital image of your face
- Change the spatial resolution and comment on the results.
- Change the number of gray levels and comment on the results.
- Obtain a digital image that shows aliasing error and comment on it.

Further Information

- The MathWorks Web site
  - http://www.mathworks.com
- Matlab Databook by Tim Love
  - http://www-h.eng.cam.ac.uk/help/tpl/programs/Matlab/matlabDatabook/
- MATLAB Primer by Kermit Sigmon
  - http://web.mit.edu/6.777/www/downloads/primer.pdf