

MATLAB for Image Processing

CS638-1 TA: Tuo Wang
tuowang@cs.wisc.edu
Feb 12th, 2010

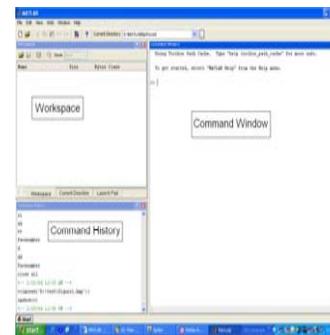
Outline

- **Introduction to MATLAB**
 - Basics & Examples
- **Image Processing with MATLAB**
 - Basics & Examples

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



- 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

MATLAB Help



- MATLAB Help is an extremely powerful assistance to learning MATLAB
- Help not only contains the theoretical background, but also shows demos for implementation
- MATLAB Help can be opened by using the HELP pull-down menu

MATLAB Help (cont.)



- Any command description can be found by typing the command in the search field
- As shown above, the command to take square root (`sqrt`) is searched
- We can also utilize MATLAB Help from the command window as shown

More about the Workspace

- `who`, `whos` – current variables in the workspace
- `save` – save workspace variables to *.mat file
- `load` – load variables from *.mat file
- `clear` – clear workspace variables

- CODE

Matrices in MATLAB

- Matrix is the main MATLAB data type
- How to build a matrix?
 - `A=[1 2 3; 4 5 6; 7 8 9];`
 - Creates matrix A of size 3 x 3
- Special matrices:
 - `zeros(n,m)`, `ones(n,m)`, `eye(n,m)`, `rand()`, `randn()`

Basic Operations on Matrices

- All operators in MATLAB are defined on matrices: `+`, `-`, `*`, `/`, `^`, `sqrt`, `sin`, `cos`, etc.
 - Element-wise operators defined with a preceding dot: `.*`, `./`, `.^`
 - `size(A)` – size vector
 - `sum(A)` – columns sums vector
 - `sum(sum(A))` – sum of all the elements
- [CODE](#)

Variable Name in Matlab

- Variable naming rules
 - must be unique in the first 63 characters
 - must begin with a letter
 - may not contain blank spaces or other types of punctuation
 - may contain any combination of letters, digits, and underscores
 - are case-sensitive
 - should not use Matlab keyword
- Pre-defined variable names
 - `pi`

Logical Operators

- `==`, `<`, `>`, (not equal) `~=`, (not) `~`
- `find('condition')` – Returns indexes of A's elements that satisfy the condition

Logical Operators (cont.)

- Example:

```
>>A=[7 3 5; 6 2 1], Idx=find(A<4)
A=
    7    3    5
    6    2    1
Idx=
    3
    4
    6
```

Flow Control

- MATLAB has five flow control constructs:
 - if statement
 - switch statement
 - for loop
 - while loop
 - break statement

if

- IF statement condition
 - The general form of the IF statement is

```
IF expression
    statements
ELSEIF expression
    statements
ELSE
    statements
END
```

- [CODE](#)

switch

- SWITCH – Switch among several cases based on expression
- The general form of SWITCH statement is:

```
SWITCH switch_expr
    CASE case_expr,
        statement, ..., statement
    CASE {case_expr1, case_expr2, case_expr3, ...}
        statement, ..., statement
    ...
    OTHERWISE
        statement, ..., statement
END
```

switch (cont.)

- Note:
 - Only the statements between the matching CASE and the next CASE, OTHERWISE, or END are executed
 - Unlike C, the SWITCH statement does not fall through (so BREAKS are unnecessary)

- [CODE](#)

for

- FOR repeats statements a specific number of times
- The general form of a FOR statement is:

```
FOR variable=expr  
    statements  
END
```
- [CODE](#)

while

- WHILE repeats statements an indefinite number of times
- The general form of a WHILE statement is:

```
WHILE expression  
    statements  
END
```
- [CODE](#)

Scripts and Functions

- There are two kinds of M-files:
 - Scripts, which do not accept input arguments or return output arguments. They operate on data in the workspace
 - Functions, which can accept input arguments and return output arguments. Internal variables are local to the function

Functions in MATLAB (cont.)

- Example:
 - A file called STAT.M:

```
function [mean, stdev]=stat(x)  
%STAT Interesting statistics.  
n=length(x);  
mean=sum(x)/n;  
stdev=sqrt(sum((x-mean).^2)/n);
```
 - Defines a new function called STAT that calculates the mean and standard deviation of a vector. Function name and file name should be the SAME!
 - [CODE](#)

Visualization and Graphics

- `plot(x,y), plot(x,sin(x))` – plot 1D function
- `figure, figure(k)` – open a new figure
- `hold on, hold off` – refreshing
- `axis([xmin xmax ymin ymax])` – change axes
- `title('figure titile')` – add title to figure
- `mesh(x_ax,y_ax,z_mat)` – view surface
- `contour(z_mat)` – view z as topo map
- `subplot(3,1,2)` – locate several plots in figure
- [CODE](#) and Debug CODE

Saving your Work

- `save mysession`
 % creates mysession.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

Outline

- **Introduction to MATLAB**
 – Basics & Examples
- **Image Processing with MATLAB**
 – Basics & Examples

What is the Image Processing Toolbox?

- The Image Processing Toolbox is a collection of functions that extend the capabilities of the MATLAB's numeric computing environment. The toolbox supports a wide range of image processing operations, including:
 - Geometric operations
 - Neighborhood and block operations
 - Linear filtering and filter design
 - Transforms
 - Image analysis and enhancement
 - Binary image operations
 - Region of interest operations

Images in MATLAB

- MATLAB can import/export several image formats:
 - BMP (Microsoft Windows Bitmap)
 - GIF (Graphics Interchange Files)
 - HDF (Hierarchical Data Format)
 - JPEG (Joint Photographic Experts Group)
 - PCX (Paintbrush)
 - PNG (Portable Network Graphics)
 - TIFF (Tagged Image File Format)
 - XWD (X Window Dump)
 - raw-data and other types of image data
- Data types in MATLAB
 - Double (64-bit double-precision floating point)
 - Single (32-bit single-precision floating point)
 - Int32 (32-bit signed integer)
 - Int16 (16-bit signed integer)
 - Int8 (8-bit signed integer)
 - Uint32 (32-bit unsigned integer)
 - Uint16 (16-bit unsigned integer)
 - Uint8 (8-bit unsigned integer)

Images in MATLAB

- Binary images : {0,1}
- Intensity images : [0,1] or uint8, double etc.
- RGB images : $m \times n \times 3$
- Multidimensional images: $m \times n \times p$ (p is the number of layers)

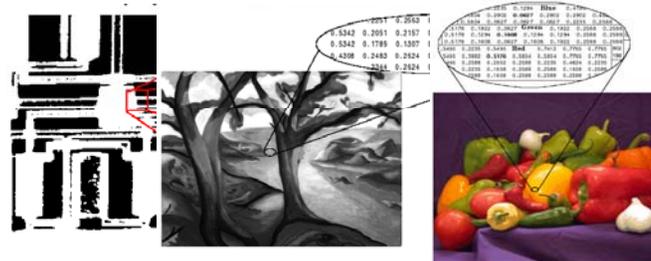


Image Import and Export

- Read and write images in Matlab

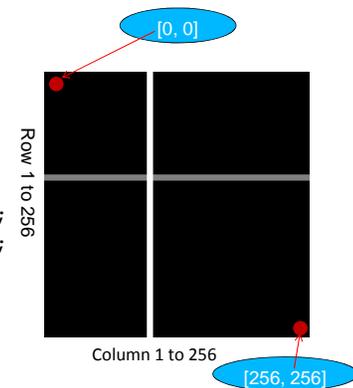

```
img = imread('apple.jpg');
dim = size(img);
figure;
imshow(img);
imwrite(img, 'output.bmp', 'bmp');
```
- Alternatives to imshow


```
imagesc(I)
imtool(I)
image(I)
```

Images and Matrices

**How to build a matrix
(or image)?**
Intensity Image:

```
row = 256;
col = 256;
img = zeros(row, col);
img(100:105, :) = 0.5;
img(:, 100:105) = 1;
figure;
imshow(img);
```



Images and Matrices

Binary Image:

```
row = 256;  
col = 256;  
img = rand(row,  
col);  
img = round(img);  
figure;  
imshow(img);
```

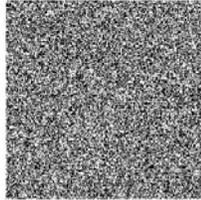


Image Display

- `image` - create and display image object
- `imagesc` - scale and display as image
- `imshow` - display image
- `colorbar` - display colorbar
- `getimage` - get image data from axes
- `trueimage` - adjust display size of image
- `zoom` - zoom in and zoom out of 2D plot

Image Conversion

- `gray2ind` - intensity image to index image
- `im2bw` - image to binary
- `im2double` - image to double precision
- `im2uint8` - image to 8-bit unsigned integers
- `im2uint16` - image to 16-bit unsigned integers
- `ind2gray` - indexed image to intensity image
- `mat2gray` - matrix to intensity image
- `rgb2gray` - RGB image to grayscale
- `rgb2ind` - RGB image to indexed image

Image Operations

- RGB image to gray image
- Image resize
- Image crop
- Image rotate
- Image histogram
- Image histogram equalization
- Image DCT/IDCT
- Convolution

[- CODE](#)

Outline

- Introduction to MATLAB
 - Basics & Examples
- Image Processing with MATLAB
 - Basics & **Examples**

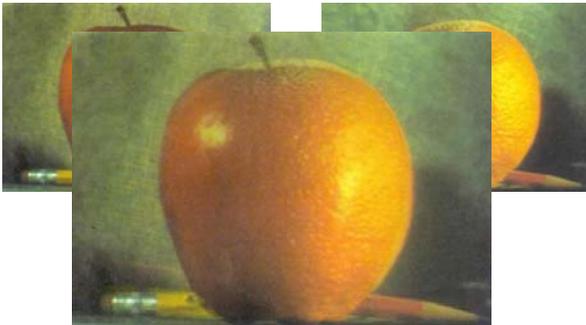
Examples working with Images (1/3)

[Create AVI movie with a series images](#)

[Video](#)

Examples working with Images (2/3)

[Blending two images](#)



Examples working with Images (3/3)

[Sobel descriptor to detect object edge](#)



Performance Issues

- The idea: MATLAB is
 - very fast on vector and matrix operations
 - Correspondingly slow with loops
- Try to avoid loops
- Try to vectorize your code
<http://www.mathworks.com/support/tech-notes/1100/1109.html>

Vectorize Loops

- Example
 - Given image matrices, A and B, of the same size (540*380), blend these two images
`apple = imread('apple.jpg');`
`orange = imread('orange.jpg');`
- Poor Style

```
% measure performance using stopwatch timer
tic
for i = 1 : size(apple, 1)
    for j = 1 : size(apple, 2)
        for k = 1 : size(apple, 3)
            output(i, j, k) = (apple(i, j, k) + orange(i, j, k))/2;
        end
    end
end
toc
```
- Elapsed time is [0.138116](#) seconds

Vectorize Loops (cont.)

- Example
 - Given image matrices, A and B, of the same size (600*400), blend these two images
`apple = imread('apple.jpg');`
`orange = imread('orange.jpg');`
- Better Style

```
tic % measure performance using stopwatch timer
Output = (apple + orange)/2;
toc
```

 - Elapsed time is [0.099802](#) seconds
- Computation is faster!

THE END

- Thanks for your attention! 😊
- Questions?