

Introduction to Matlab

CSC420 Spring 2017

Introduction to Image Understanding

Presented by: Hang Chu

Outline

1. Introduction

1. Overview

2. Variables

3. Matrix

4. Misc.

2. Image Processing with Matlab

3. References

What & Why

- **Matrix Laboratory**
 - **Dynamically** typed language
 - Variables require no declaration
 - Creation by initialization (x=10;)
 - All variables are treated as **matrices**
 - Scalar: 1×1 matrix; Vector: $N \times 1$ or $1 \times N$ matrix
 - Calculations are much faster
- **Advantages**
 - **Fast implementation and debugging**
 - Natural matrix operation
 - Powerful image processing toolbox



Matlab Main Screen

Command Window

- type commands

Current Directory

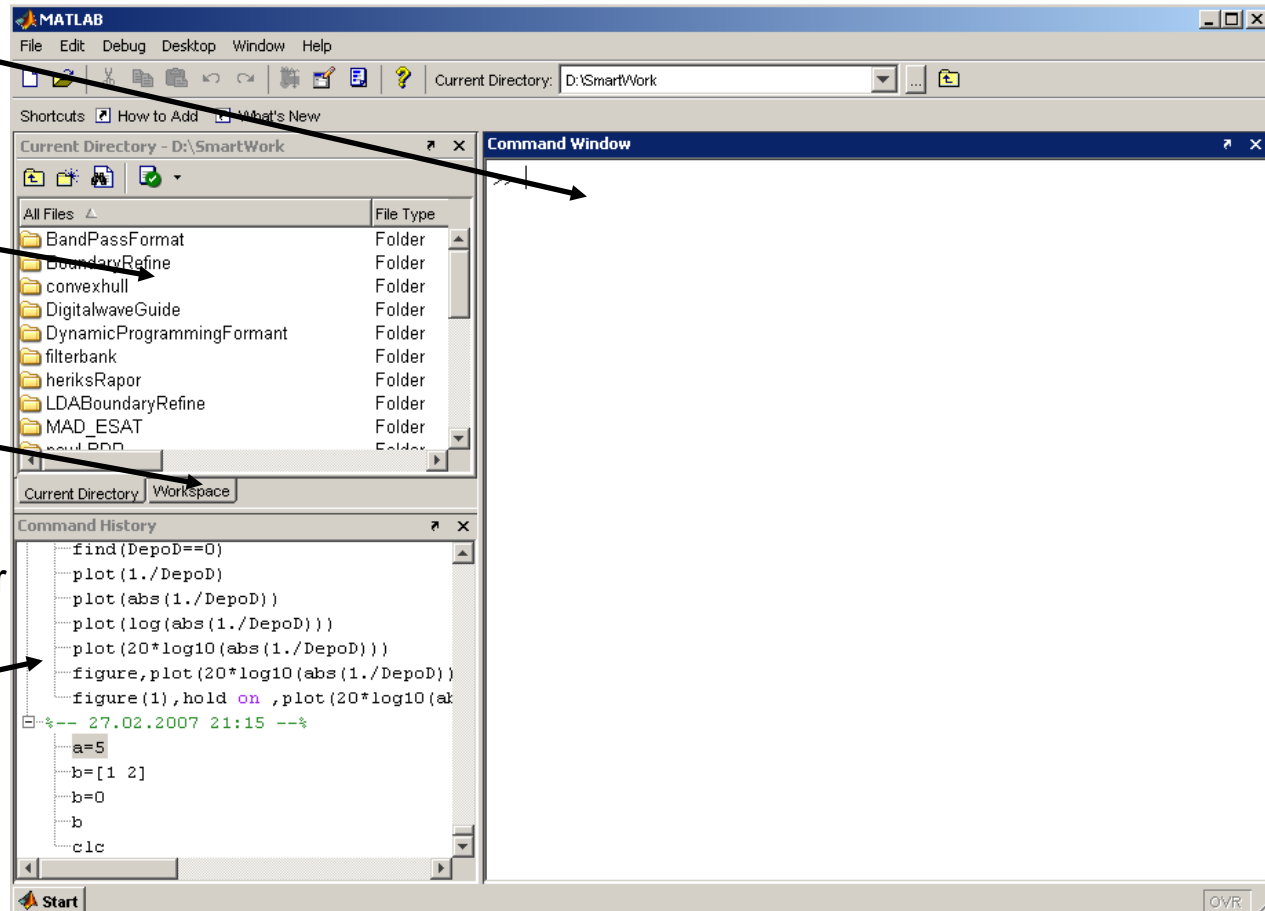
- View folders and m-files

Workspace

- View variables
- Double click on a variable to see it in the Array Editor

Command History

- view past commands
- save a whole session using diary



Outline

1. Introduction

1. Overview

2. Variables

3. Matrix

4. Misc.

2. Image Processing with Matlab

3. References

Variables

Defining variables

```
int a;  
a=1;  
double b;  
b=2+4;
```

C/C++

```
>>a=1;  
>>b=2+4;
```

Matlab

```
>> a=1  
a =  
    1  
>> b=2+4  
b =  
    6
```

Variables are created when they are used

All variables are created as matrices with “some” type (unless specified)

Variables

`a = 1;`

a <1x1 double>		
	1	2
1	1	
2		

```
>> whos a
```

Name	Size	Bytes	Class	Attributes
a	1x1	8	double	

`b = false;`

b <1x1 logical>		
	1	2
1	0	
2		

Variables

$A = [1, 2, 3]$

```
>> A=[1 2 3]
```

```
A =
```

```
    1    2    3
```

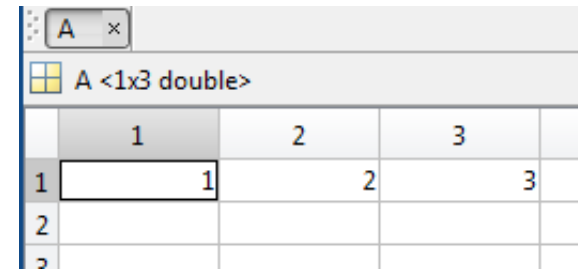
$B = [1,2,3;4,5,6]$

```
>> B=[1,2,3;4,5,6]
```

```
B =
```

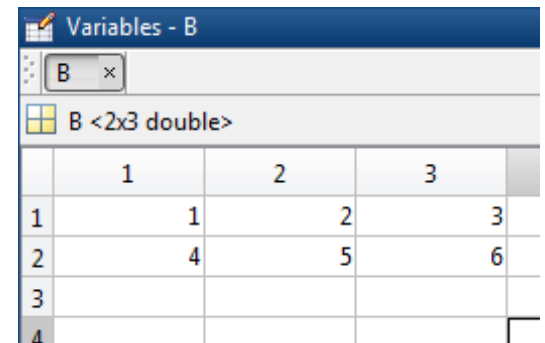
```
    1    2    3
    4    5    6
```

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



A screenshot of the MATLAB variable viewer window titled 'A'. It shows a 1x3 double array with the following values:

	1	2	3
1	1	2	3
2			
3			



A screenshot of the MATLAB variable viewer window titled 'Variables - B'. It shows a 2x3 double array with the following values:

	1	2	3
1	1	2	3
2	4	5	6
3			
4			

```
>> C= [1 2 3 ; 4 5 6; 7 8 9]
```

```
C =
```

```
    1    2    3
    4    5    6
    7    8    9
```


Variables

$D = [1 ; 2 ; 3]$

```
>> D = [1 ;2 ;3]
D =
     1
     2
     3
```

D <3x1 double>		
	1	2
1	1	
2	2	
3	3	
4		

$E = [1 \ 2 \ 3]'$

```
>> E = [1 2 3]'
E =
     1
     2
     3
```

Variables

```
>> A=1:10
```

```
A =
```

```
    1    2    3    4    5    6    7    8    9   10
```

```
>> B= 0:2:10
```

```
B =
```

```
    0    2    4    6    8   10
```

```
>> 1:0.5:5
```

```
ans =
```

```
    1.0000    1.5000    2.0000    2.5000    3.0000    3.5000    4.0000    4.5000    5.0000
```

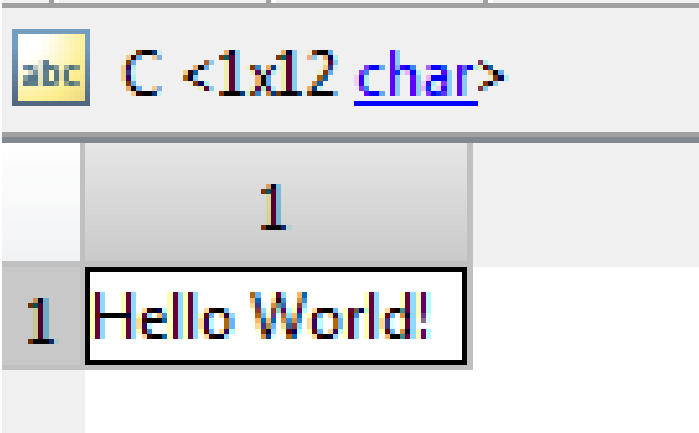
Variables

```
C = 'Hello World!';
```

```
>> C='Hello World!'
```

```
C =
```

```
Hello World!
```



The image shows a MATLAB variable editor window. The title bar reads 'abc C <1x12 char>'. The main area contains a table with one row and one column. The row index is '1' and the value is 'Hello World!'. The cell containing 'Hello World!' is highlighted with a black border.

	C <1x12 char>
1	Hello World!

Variables

```
A = zeros(3);
```

```
B = ones(5);
```

```
C = rand(100,2);
```

```
D = eye(20);
```

```
E = sprintf('%d\n',9);
```

Outline

1. Introduction

1. Overview
2. Variables
- 3. Matrix**
4. Misc.

2. Image Processing with Matlab

3. References

Matrix Index

Matrix indices begin from 1 (not 0!!!)

Matrix indices must be positive integers

A =

```
1   2   3
4   5   6
7   8   9
```

```
>> A(1,3)
```

```
ans =
     3
```

```
>> A(1)
```

```
ans =
     1
```

```
>> A(2)
```

```
ans =
     4
```

Column-Major Order

```
>> A(0)
```

Subscript indices must either be real positive integers or logicals.

```
>> A(1,4)
```

Index exceeds matrix dimensions.

```
^^
```

Matrix Index

A =

```
1     2     3
4     5     6
7     8     9
```

```
>> A(2,2:3)
```

```
ans =
```

```
5     6
```

```
>> A(2,1:end)
```

```
ans =
```

```
4     5     6
```

```
>> A(2,:)
```

```
ans =
```

```
4     5     6
```

```
>> A(2,1:2:3)
```

```
ans =
```

```
4     6
```

```
>> A(2,[1 3])
```

```
ans =
```

```
4     6
```

```
>> A(:)
```

```
ans =
```

```
1
4
7
2
5
8
3
6
9
```

Matrix Index

Accessing Elements

```
A = rand(4);
```

```
A(2,3)
```

```
A(:,2)
```

```
A(end,:)
```

```
A([1,2],[1,3])
```

```
A(1:2,3:end)
```

<http://www.mathworks.com/company/newsletters/articles/matrix-indexing-in-matlab.html>

Matrix Operations

+ addition

- subtraction

* multiplication

\wedge power

' complex conjugate transpose

Matrix Operations

Given A and B:

```
>> A = [1 2 3;4 5 6;7 8 9]

A =

     1     2     3
     4     5     6
     7     8     9
```

```
>> B = [3 5 2; 5 2 8; 3 6 9]

B =

     3     5     2
     5     2     8
     3     6     9
```

Addition

```
>> X = A + B

X =

     4     7     5
     9     7    14
    10    14    18
```

Subtraction

```
>> Y = A - B

Y =

    -2    -3     1
    -1     3    -2
     4     2     0
```

Product

```
>> Z = A * B

Z =

    22    27    45
    55    66   102
    88   105   159
```

Transpose

```
>> T = A'

T =

     1     4     7
     2     5     8
     3     6     9
```

Matrix Operations

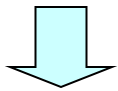
- .^{*} element-wise multiplication
- ./ element-wise division
- .[^] element-wise power

Matrix Operations

```
A = [1 2 3; 5 1 4; 3 2 1]
```

```
A =
```

```
    1    2    3  
    5    1    4  
    3    2   -1
```



```
x = A(1,:)
```

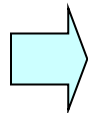
```
y = A(3 ,:)
```

```
x =
```

```
    1    2    3
```

```
y =
```

```
    3    4   -1
```



```
b = x .* y
```

```
c = x ./ y
```

```
d = x .^y
```

```
b =
```

```
    3    8   -3
```

```
c =
```

```
    0.33    0.5   -3
```

```
d =
```

```
    1    16    0.33
```

Matrix Operations

A/B Solve linear equation $xA=B$ for x

$A\backslash B$ Solve linear equation $Ax=B$ for x

Matrix Concatenation

$$X=[1 \ 2], Y=[3 \ 4]$$

```
>> A=[X Y]
```

```
A =
```

```
    1    2    3    4
```

```
>> A=[X ; Y]
```

```
A =
```

```
    1    2  
    3    4
```

Outline

1. Introduction

1. Overview
2. Variables
3. Matrix
4. **Misc.**

2. Image Processing with Matlab

3. References

Strings

```
A = 'vision and geometry'  
strfind(A, 'geometry')  
strcmp(A, 'computer vision')  
B = strcat(A, ' 12345')  
c = [A, ' 12345']  
D = sprintf('I am %02d years old.\n', 9)  
int2str, str2num, str2double
```

<http://www.mathworks.com/help/matlab/ref/strings.html>

Cell and Structure

- Cells

- `a = {};` `a = cell(1)`
- `b = {1,2,3}`
- `c = {{1,2},2,{3}}`
- `D = {'cat','dog','sheep','cow'}`
- `E = {'cat',4}`

```
>> D{2}
ans =
dog
```

```
>> E{1}
ans =
cat
>> E{2}
ans =
    4
```

- Structures

- `A = struct('name','1.jpg','height',640,'width',480);`
- `b.name = '1.jpg'`

http://www.mathworks.com/help/matlab/matlab_prog/cell-vs-struct-arrays.html

Operators

== Equal to

~= Not equal to

< Strictly smaller

> Strictly greater

<= Smaller than or equal to

>= Greater than equal to

& And operator

| Or operator

Flow Control

- if, for, while

```
if (a<3)
    Some Matlab Commands;
elseif (b~=5)
    Some Matlab Commands;
end
```

```
while ((a>3) & (b==5))
    Some Matlab Commands;
end
```

```
for ii=1:100
    Some Matlab Commands;
end
```

```
for j=1:3:200
    Some Matlab Commands;
end
```

```
for k=[0.1 0.3 -13 12 7 -9.3]
    Some Matlab Commands;
end
```

Vectorization

Optimize your code for Matrix operations

Examples

In other languages:

```
tic; i = 0;  
for t = 0:.001:1000  
    i = i + 1;  
    y(i) = sin(t);  
end; toc;
```

Elapsed time is 0.509381 seconds.

In MATLAB:

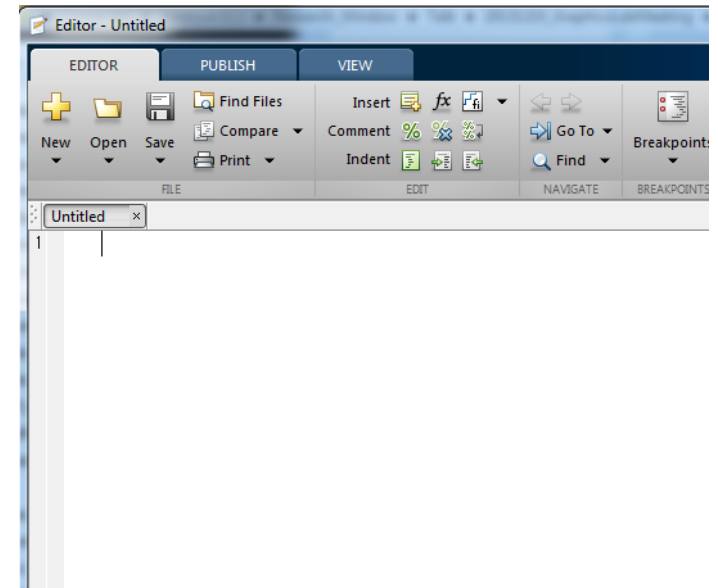
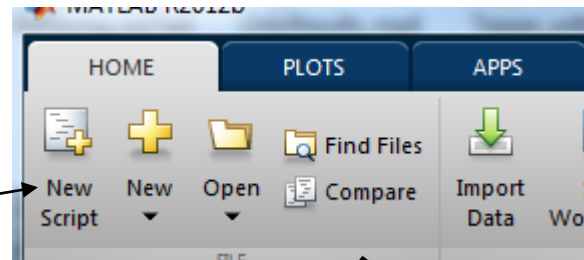
```
tic; t = 0:.001:1000;  
y = sin(t); toc;
```

Elapsed time is 0.011212 seconds.

http://www.mathworks.com/help/matlab/matlab_prog/vectorization.html

M-File

Click to create
a new M-File



- A text file containing script or function
- Extension “.m”

Functions

For example,

Implement your own function Add3()

B = Add3(A)

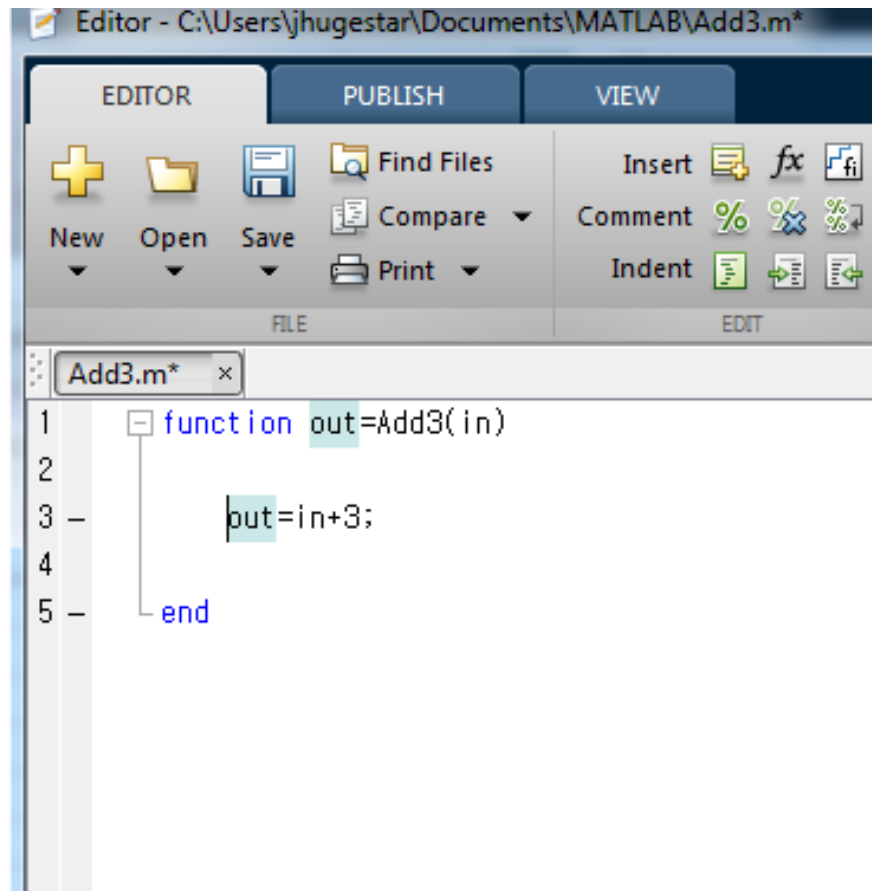
How?

Create a M-file with the function name

Use the function definition at the beginning

```
function out1=functionname(in1)
function out1=functionname(in1,in2,in3)
function [out1,out2]=functionname(in1,in2)
```

Functions



The screenshot shows the MATLAB Editor interface with the file 'Add3.m' open. The editor window displays the following code:

```
1 function out=Add3(in)
2
3     out=in+3;
4
5 end
```

```
>> a =magic(3)
```

```
a =
```

```
     8     1     6
     3     5     7
     4     9     2
```

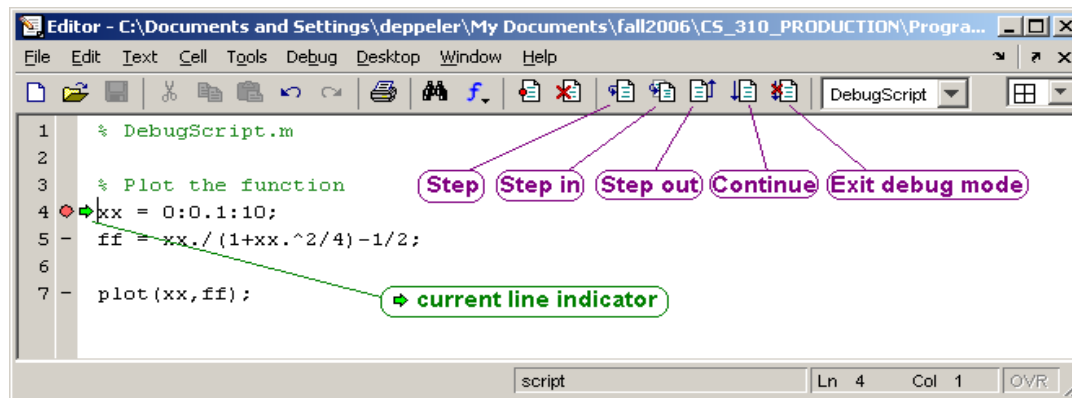
```
>> b =Add3(a)
```

```
b =
```

```
    11     4     9
     6     8    10
     7    12     5
```

Debugging

Breakpoints

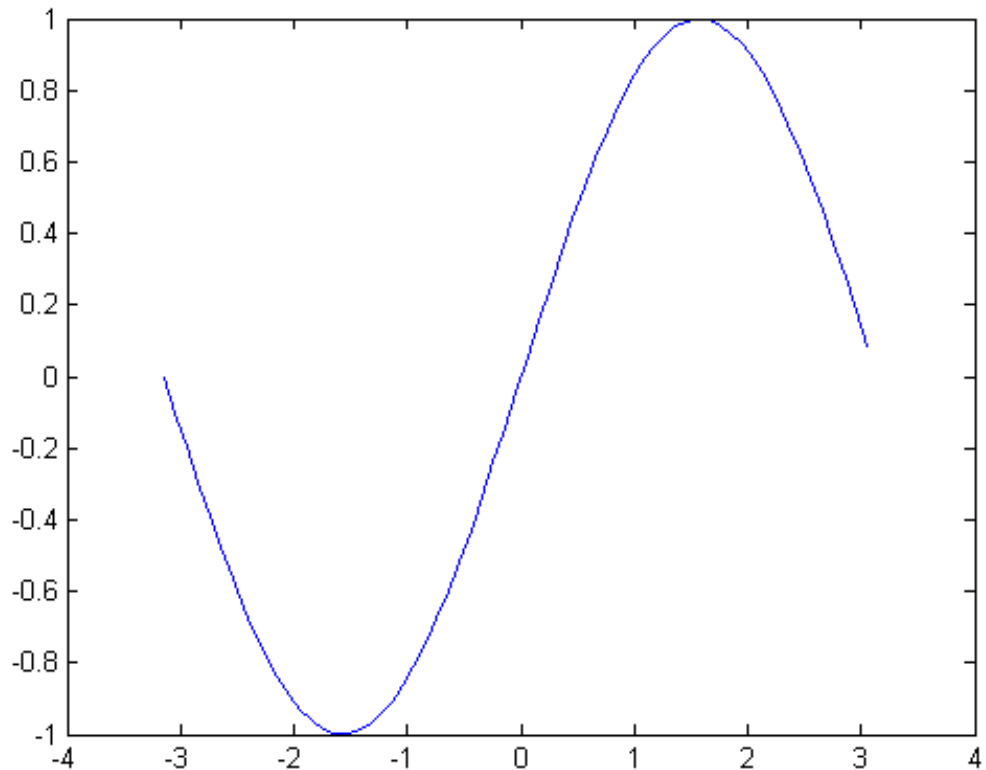


Plotting

Plotting functions

plot, plot3d, bar, area, hist, contour, mesh

```
x = -pi:1:pi;  
y = sin(x);  
plot(x,y)
```



Help & Doc

help functionName

doc functionName

Outline

1. Introduction

1. Overview
2. Variables
3. Matrix
4. Misc.

2. Image Processing with Matlab

3. References

Image Data Structure

- Image as **matrices**

- Gray image: $m \times n$
- RGB image: $m \times n \times 3$

- Format:

- $[0, 255]$ uint8
- $[0, 1]$ double

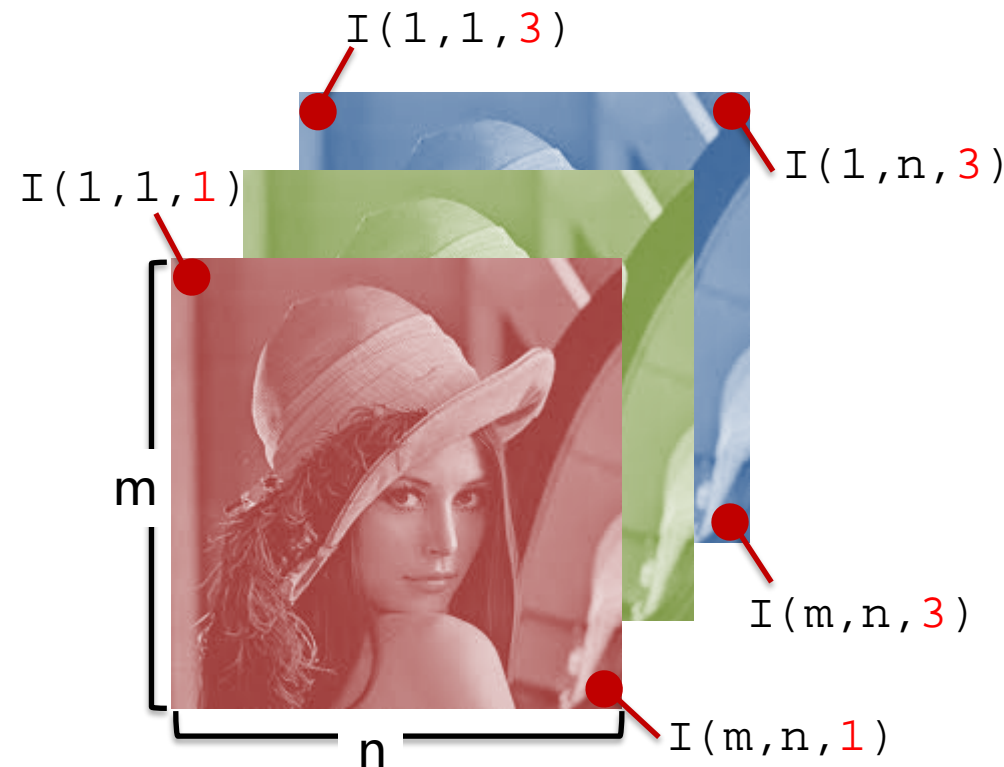


Image I/O/Display

```
% Read image (support bmp, jpg, png, ppm, etc)
```

```
I = imread('lena.jpg');
```

```
% Save image
```

```
imwrite(I, 'lena_out.jpg');
```

```
% Display image
```

```
imshow(I);
```

```
% Alternatives to imshow
```

```
imagesc(I);
```

```
imtool(I);
```

```
image(I);
```

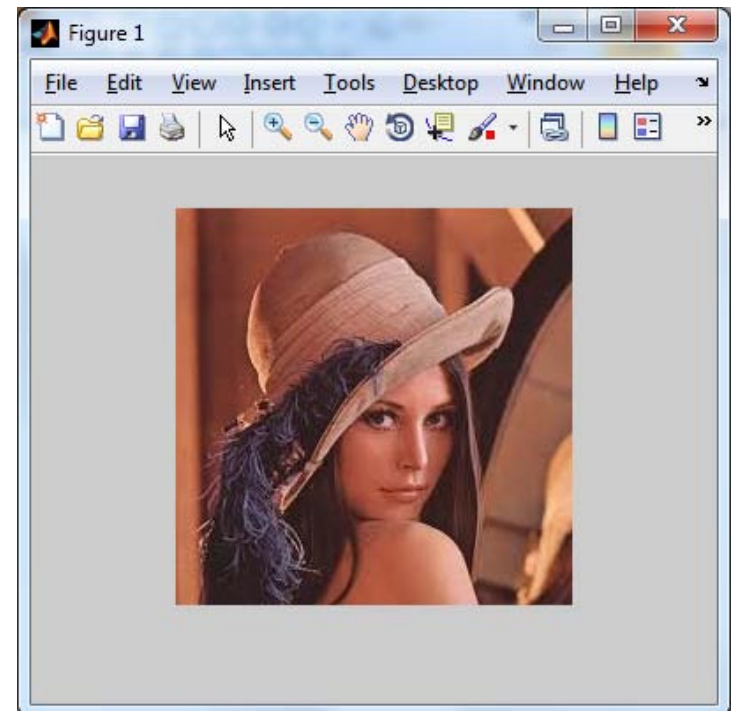


Image Conversions

```
% Type conversion
```

```
I1 = im2double(I);
```

```
I2 = im2uint8(I);
```

```
% Convert from RGB to grayscale
```

```
I3 = rgb2gray(I);
```

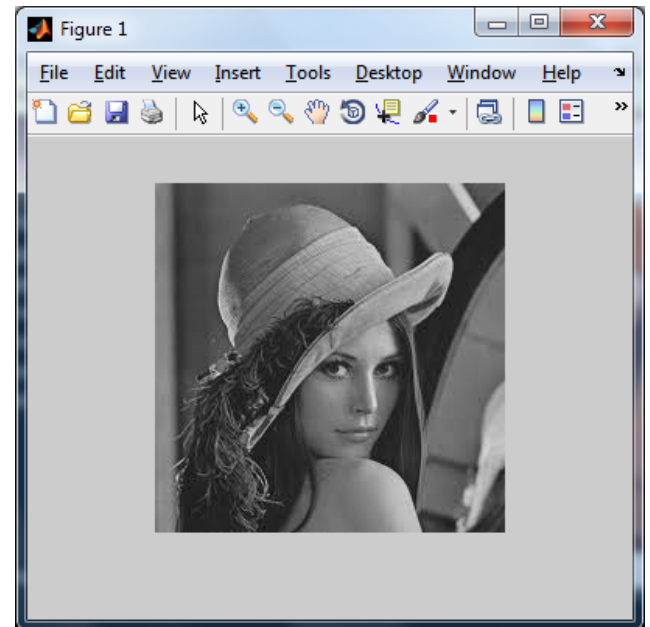


Image Operations

```
% Resize image as 60% smaller
```

```
Ires = imresize(I, 0.6);
```

```
% Crop image from user's input
```

```
imshow(I);
```

```
Rect = getrect;
```

```
Icrp = imcrop(I, Rect);
```

```
% Rotate image by 45 degrees
```

```
Irot = imrotate(I, 45);
```

```
% Affine transformation
```

```
A = [1 0 0; .5 1 0; 0 0 1];
```

```
tform = maketform('affine', A);
```

```
Itran = imtransform(I, tform);
```

$$\mathbf{p}_{warped}^i = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \mathbf{p}_{source}^i + \mathbf{t}$$

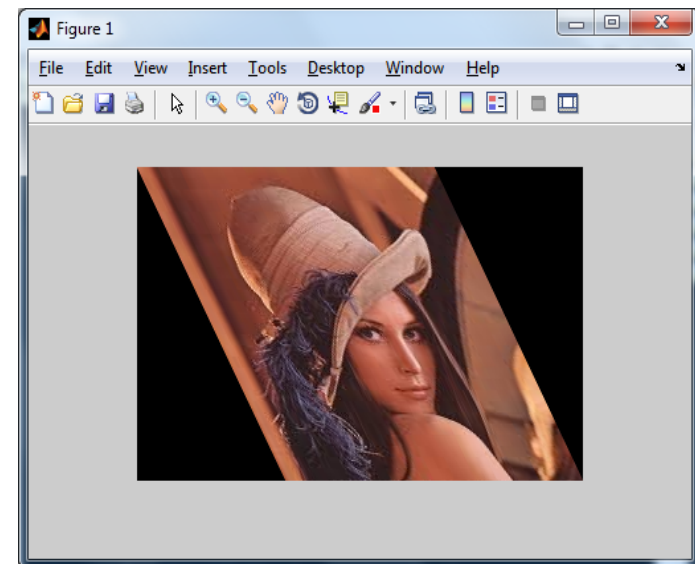
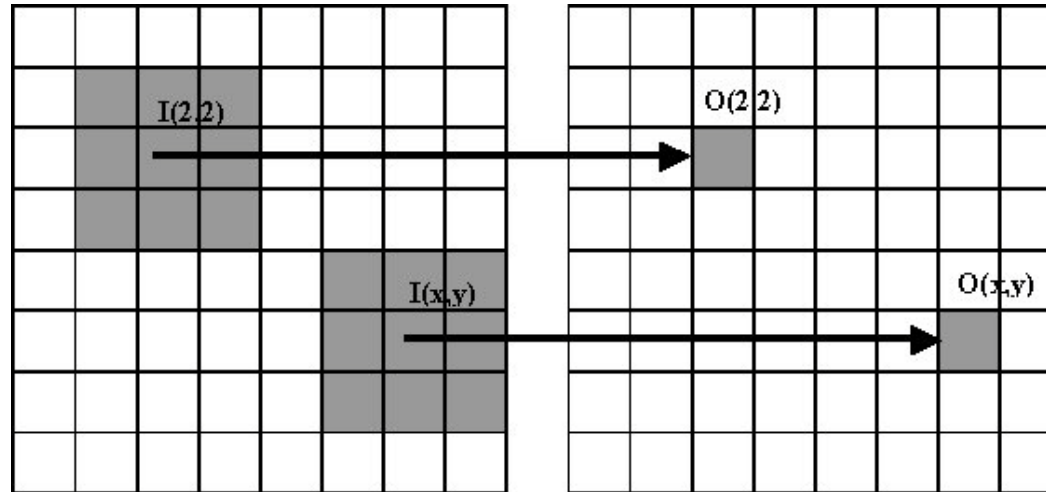


Image Filtering / Convolution

- A **filter** (or called mask, kernel, neighborhood) is $N \times N$ matrix.



- Filters help us perform different kinds of operations:



Outline

1. Introduction

1. Overview
2. Variables
3. Matrix
4. Misc.

2. Image Processing with Matlab

3. References

References

More tutorials

- Matlab course @ ETHZ (<http://goo.gl/W2jmZJ>)
- Introductory Digital Processing @ IIT (<http://goo.gl/U0osD2>)

Open source CV algorithms with Matlab interface

- VLFeat (<http://www.vlfeat.org/>)
- Piotr Dollar's toolbox (<http://vision.ucsd.edu/~pdollar/toolbox/>)
- Mexopencv (<http://www.cs.stonybrook.edu/~kyamagu/mexopencv/>)

References

– Matlab Documentation

- <http://www.mathworks.com/help/matlab/>

– Cheat Sheets

- <http://web.mit.edu/18.06/www/Spring09/matlab-cheatsheet.pdf>
- http://www.geog.ucsb.edu/~pingel/210b/general/matlab_refcard.pdf

Thank you!