CSE185
Introduction to Computer Vision
Lab 01: Image Processing in MATLAB

Instructor: Daniel Leung

TA: Mohammadkazem Ebrahimpour

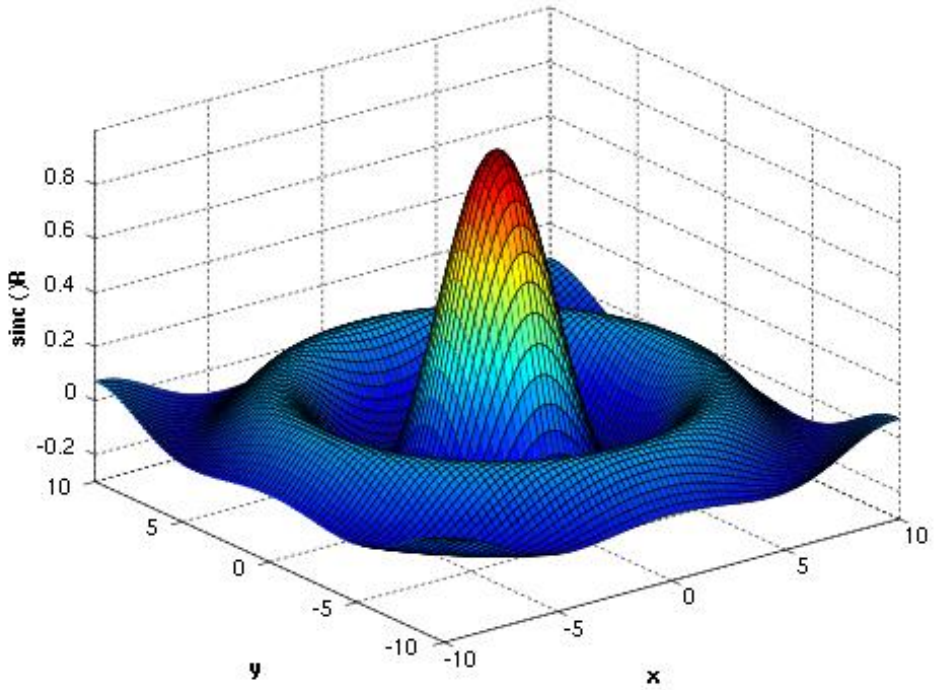
Xueqing Deng

Lab Rules

- Attendances are mandatory
- Each lab assignment is closed at 11:59pm of the 7th day after it is assigned. You cannot submit any work after it is closed.
- You must demo your lab submission within 14 days after it is assigned in order for it to be graded.
  - ALL SUBMITTED LABS WITHOUT DEMO **WILL NOT BE GRADED**.
  - If you demo it within 7 days, you are allowed to make corrections and re-submit it before the assignment is closed.
  - You will have time to demo your submissions to your TA during lab time of the following week (during last hour or two).
- If you expect to submit your work late, you must request for approval from Daniel only **BEFORE** the due date.
  - All late submission requests after the due date will not be considered unless accompanied with proper documentations of excuses.

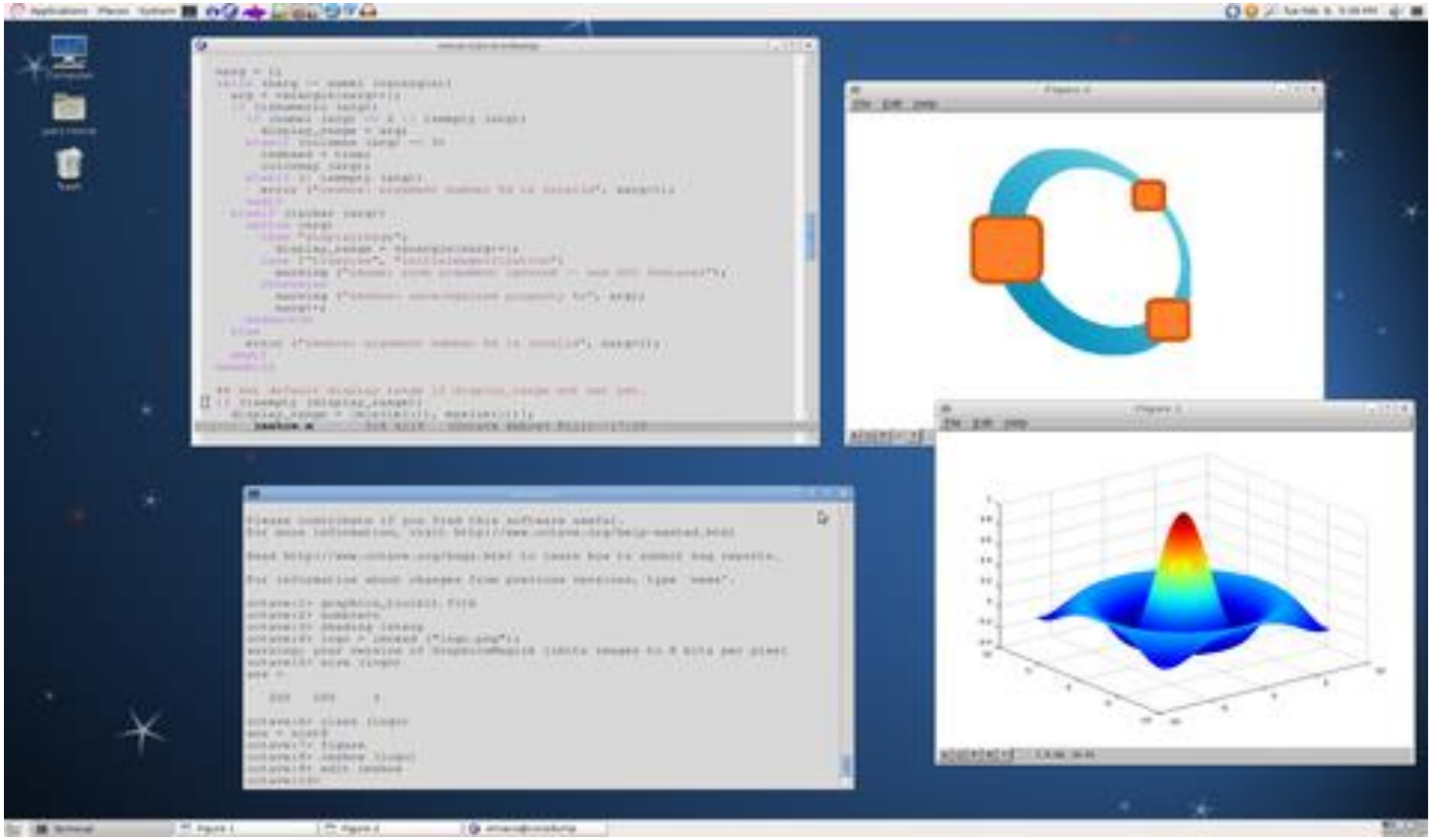
Introduction to MATLAB

- MATLAB is a numerical computing environment
- Allow easy operation on matrix, image, N-D data
- Easily plot and visualize data
- Simple GUI
- Interface with other languages (e.g. C/C++, Python)



Introduction to MATLAB

- MATLAB is NOT free
  - need license, but free for UCM students
- GNU Octave is free and compatible with MATLAB



Variable

- Variable: no declaration, implicit type conversion

```
>> x = 10
x =
    10
>> x = 'test'
x =
test
>> z = 10; y = z + 10
y =
    20
>> y = z^2
y =
    100
>> y = mod(z, 3)
y =
    1
```

MATLAB use single quote for string

Vector

- Vector: use `[ ]` or `init:step:end`

```
>> vec = [1, 100]
vec =
    1    100
>> vec = 1:2:10
vec =
    1    3    5    7    9
```

- Use ( ) to access elements (index starts from 1):

```
>> vec(3)
ans = 5
```

- Access part of vector:

```
>> vec(2:4)
ans =
    3    5    7
```

Matrix

- Matrix: use semicolon to separate each row

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

- Access elements:

```
>> A(2, 3)
ans =
     6
```

- Access sub-matrix (Useful and Important!):

```
>> A(1:2, 2:3)
ans =
     2     3
     5     6
```

Multiplication

- Matrix-vector multiplication:

```
>> A = [1 2 3; 4 5 6; 7 8 9];
>> x = [1; 1; 1];
>> A * x
ans =
     6
    15
    24
```

$$A * x = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} * \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 6 \\ 15 \\ 24 \end{pmatrix}$$

- Element-wise multiplication:

```
>> A = [1 2 3; 4 5 6; 7 8 9];
>> B = [1 1 1; 2 2 2; 3 3 3];
>> A .* B
ans =
     1    2    3
     8   10   12
    21   24   27
```

$$A .* B = \begin{pmatrix} 1 \cdot 1 & 2 \cdot 1 & 3 \cdot 1 \\ 4 \cdot 2 & 5 \cdot 2 & 6 \cdot 2 \\ 7 \cdot 3 & 8 \cdot 3 & 9 \cdot 3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 8 & 10 & 12 \\ 21 & 24 & 27 \end{pmatrix}$$

If statement

- If statement

```
if EXPRESSION
    ...
end
```

- If-else statement

```
if EXPRESSION
    ...
else
    ...
end
```

```
if EXPRESSION
    ...
elseif EXPRESSION
    ...
else
    ...
end
```

Loop

- For loop

```
for i = 1:10
    ...
end
```

- While loop

```
while EXPRESSION
    ...
end
```

M file

- MATLAB executable file/script, function: *.m file
- Write commands in the script, and type script file name in command window to run the script

Editor - D:\Dropbox\phoenix104104\CSE185\Lab01\lab01.m

lab01.m

```
1
2       % load image
3 -     img = imread('03.jpg');
4 -     fprintf('Load 03.jpg...\n');
5
6       % show image
7 -     figure, imshow(img);
8
9       % save image
10 -    imwrite(img, 'output.jpg');
11 -    fprintf('Save output.jpg...\n');
12
```

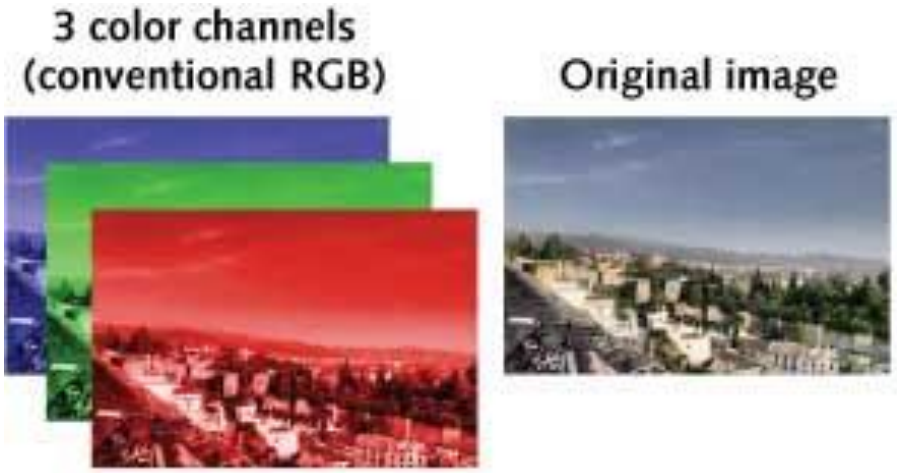
Command Window

New to MATLAB? See resources for Getting Started.

```
>> lab01
Load 03.jpg...
Save output.jpg...
>>
```

Color Image

- Color image is a 3-D matrix in MATLAB: $Height \times Width \times 3$

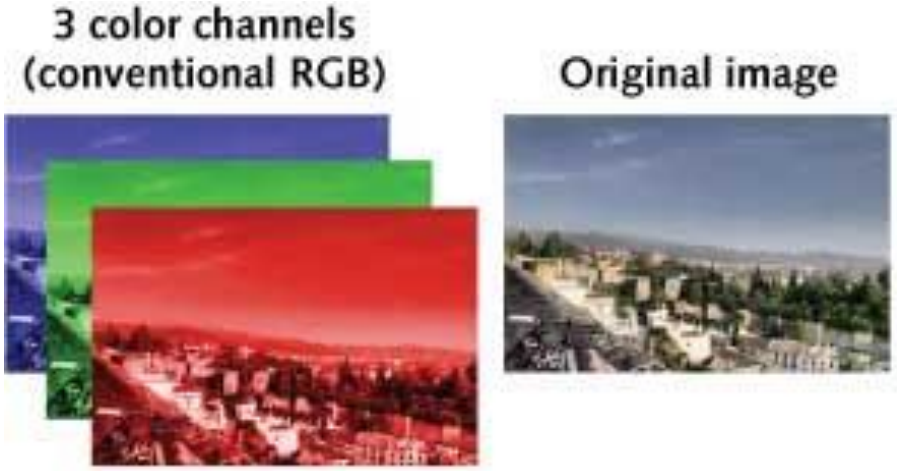


- Read image: `I = imread(filename);`
- Show image: `figure, imshow(I);`
- Save image: `imwrite(I, filename);`

Hint: type `help imread` in command window, or press F1 on function name to see the usage of the function

Color Image

- Color image is a 3-D matrix in MATLAB: $Height \times Width \times 3$



- `I(:, :, 1)` is red channel
- `I(:, :, 2)` is green channel
- `I(:, :, 3)` is blue channel

: means select all elements in this dimension

Gray-scale Image

- Gray-scale image is a 2-D matrix in MATLAB: $Height \times Width$ (Only one intensity layer)

Gray-scale image

Color Image

- Use `size()` to check matrix dimension

```
>> I = imread('01_gray.jpg');
>> size(I)
ans =
    300    400
```

Pixel Range and Type

- When loading image to MATLAB:
  - Default data type is uint8
  - Each pixel/element has a value between 0 and 255 (8 bits)

- Use `im2double()` to convert data type to double:
  - pixel range is between 0 and 1

```
>> I = imread('01.jpg');
>> I(1, 1)
ans =
    34
>> I = im2double(I);
>> I(1, 1)
ans =
    0.1333
```

The same as `I = double(I) / 255.0;`

Image Processing in MATLAB

- Set the value of green channel to zero

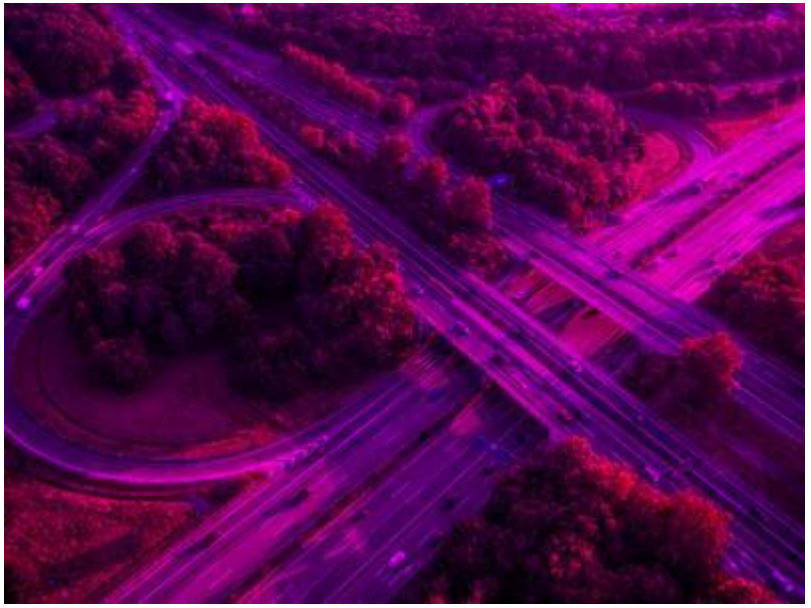


Image Processing in MATLAB

- Convert RGB to Y (gray-scale)

$$Y = 0.299 \times R + 0.587 \times G + 0.114 \times B$$

- Do NOT use `rgb2gray()` function in MATLAB

Reference: RGB to YUV https://en.wikipedia.org/wiki/YUV

Image Processing in MATLAB

- Rotate image 90 degree: use `imrotate()`

Image Processing in MATLAB

- Crop image boundary: extract sub-matrix

Image Processing in MATLAB

- Crop image boundary: extract sub-matrix

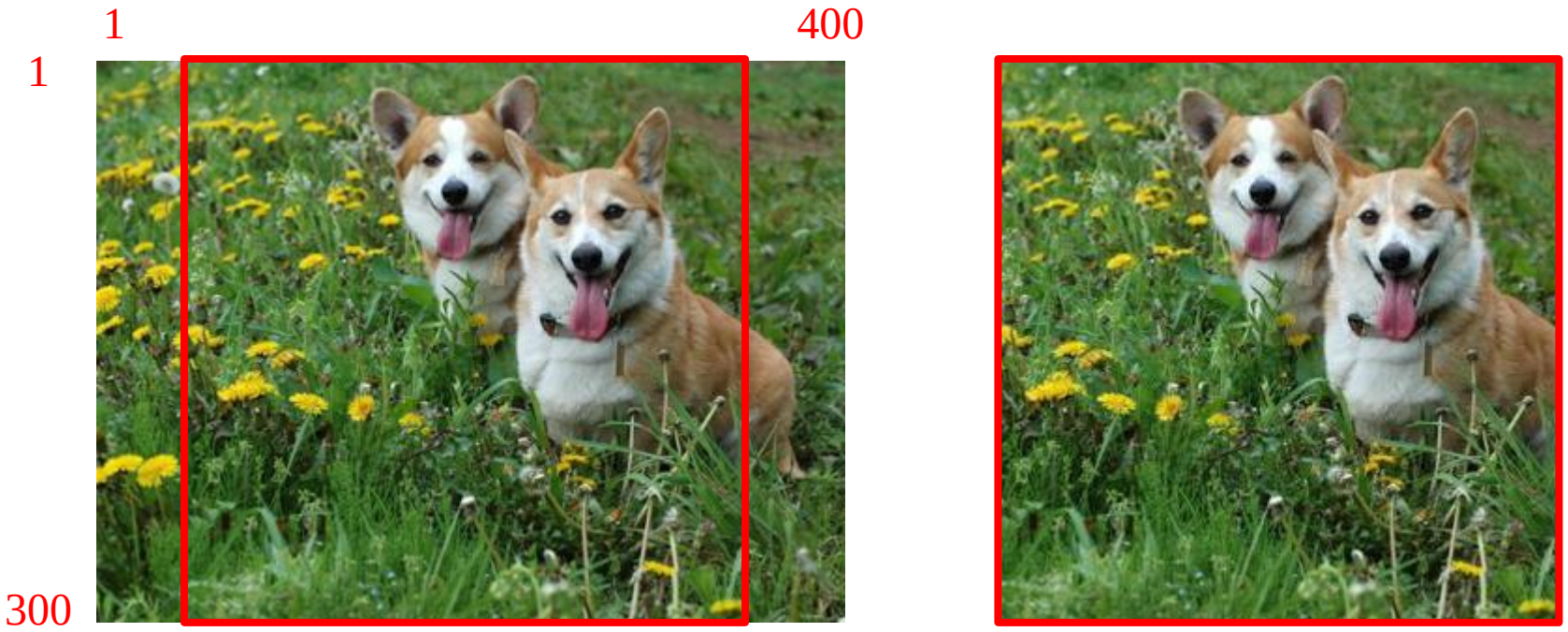


Image Processing in MATLAB

- Horizontally flip image: use `flip()`

Image Processing in MATLAB

- Combine 4 images into one big image with 2 x 2 grid

Image Processing in MATLAB

- Combine 4 images into one image with 2 x 2 grid
- Hint: use `zeros(Height, Width, 3, 'uint8')` to create a canvas/matrix first, and consider each image as a sub-matrix of the canvas

- The size of our testing image is $300 \times 400 \times 3$, use 10 pixels for separations:

```
I1 = imread('01.jpg');
canvas = zeros(300 * 2 + 10, 400 * 2 + 10, 3, 'uint8');
canvas(1:300, 1:400, :) = I1;
```

Image Processing in MATLAB

- Use `(:)` to convert image/matrix to vector
  - matrix in MATLAB is column-major (not like in C)

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \qquad A(:) = \begin{pmatrix} 1 \\ 4 \\ 7 \\ 2 \\ \vdots \\ 3 \\ \vdots \\ 9 \end{pmatrix}$$

- Use `reshape()` to convert vector to image/matrix

```
>> I = imread('01.jpg');
>> I = I(:);
>> size(I)
ans =
    360000         1
>> I = reshape(I, 300, 400, 3);
>> size(I)
ans =
    300   400     3
```

The size of image is $300 \times 400 \times 3$

The size of vector is 360000×1

Image Processing in MATLAB

- Average two image vectors, and convert the vector back to image

TODO

1. Set red channel of 01.jpg to zero, and save as red.jpg (2pt)
2. Convert 02.jpg from RGB to gray scale, and save as gray.jpg (2pt)
3. Rotate 03.jpg by 90 degree, and save as rotate.jpg (2pt)
4. Crop 04.jpg 30 pixels from all sides, and save as crop.jpg (4pt)
5. Vertically flip 04.jpg, and save as flip.jpg (2pt)
6. Combine 4 images (01.jpg ~ 04.jpg) into one matrix with 2 x 2 grid and 15 pixels separations, and save as combine.jpg (4pt)
7. Convert 04.jpg and flip.jpg to vectors, average them, convert vector back to image, and save as average.jpg (4pt)
8. Save all the code in lab01.m and upload all output images and your lab01.m (in a zip file)

Tips

- Without the ending semicolon, MATLAB will print the value of this function/variable
- Use command `close all;` to close all figures at once
- Use command `clear all;` to delete all variables in workspace
- Use command `clc;` to clean/flush command window
- When using submatrix indexing (`index1:index2`), both the indexes are included
  - `x(1:300)` counts from 1 to 300, totally 300 elements
  - `x(300:600)` counts from 300 to 600, totally 301 elements

Reference

- MATLAB: http://www.mathworks.com/products/matlab/
- Octave: https://www.gnu.org/software/octave/
- Introduction to MATLAB with Image Processing http://www.slideshare.net/Sutanshu_Raj/introduction-to-matlab-with-image-processing-5495912
- MATLAB tutorials: https://www.tutorialspoint.com/matlab/