Introduction of Digital Image Processing in Matlab

Author: Chul Min Yeum

Email: cmyeum@uwaterloo.ca

Last updated: 2020-12-15

Table of Contents

Introduction

Description

This tutorial is designed to provide a general understanding of digital image processing as well as how to use Image Processing Toolbox in MATLAB. Students can run and configure each section of code and make an effor to understand (1) objective of each code section, (2) meaning of outcome, and (3) underlying working principle or logic of functions in Image Processing Toolbox. Students should be familar wtih all these funtions used in this tutorial to complete task assignments in the course. In this tutorial, the detailed explanation of each code section is omitted. Students try to search for the meaning and encourage to ask relevant questions on our course website if they are not clear.

Reference

- Introduction to digital image processing with MATLAB (Chapters 1-4)
- Fundamentals of digital image processing (Chapters 1 and 2)
- Matlab Primer Part 1
- Matlab Primer Part 2

- Image Processing Toolbox™ User's Guide

Before you start

- Make sure that this code is running on the directory where this file present. It is very common mistake.
- Copy sample images into your folder.
- The initial MATLAB theme may not be comfortable for your eye. If you have a better color scheme (however, it is a matter of personal taste), please run this "*ChangeColor.m*" .

Tutorials

Image size

```
sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'car1.jpg'); % image filename
img = imread(filenameImg); % read an image
imshow(img); % show the image
```

```
Warning: Image is too big to fit on screen; displaying at 33%
```

```
[w,h,c] = size(img); % w: height, h: width, c: channel
fprintf('height: %d, width: %d, and channel(depth): %d', w, h, c)
```

```
height: 2336, width: 3504, and channel(depth): 3
```

```
info = imfinfo(filenameImg);

if strcmp(info.ColorType, 'truecolor');  c = 3;  else;  c = 1; end
fprintf('width: %d, height: %d, and channel(depth): %d', info.Width, info.Height, c);
```

```
width: 3504, height: 2336, and channel(depth): 3
```

```
% alternative
if ndims(img)==3; c = 3; else; c = 1; end
fprintf('width: %d, height: %d, and channel(depth): %d', info.Width, info.Height, c);
```

```
width: 3504, height: 2336, and channel(depth): 3
```

Q.001: Print "info" and identify what kind of information (metadata) is stored in the image?

Q.002: What is the resolution (size) of image? Please run this code with 'cameraman.tif' in the 'img' folder. What's the resolution of 'cameraman.tif'? Can you see the difference between color and grayscale images?

Image coordinate systems

```
sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'pout.tif'); % image filename
img = imread(filenameImg); % read an image

figure;
subplot(121);
imshow(img); hold on;
plot(50, 150,'ro'); % x and y indicate the width and height of the image !! (Don't be confused)
plot(150, 50,'bo'); hold off

subplot(122);
img2 = insertShape(img, 'circle', [50 150 5], 'lineWidth', 1, 'Color', 'Red');
img2 = insertShape(img2, 'circle', [150 50 5], 'lineWidth', 1, 'Color', 'Blue');
imshow(img2);
```

```
[w,h,c] = size(img); % w: width, h: height, c: channel
fprintf('img: width: %d, height: %d, and channel(depth): %d', w, h, c)
```

```
img: width: 291, height: 240, and channel(depth): 1
```

```
clear w h c

[w,h,c] = size(img2); % w: width, h: height, c: channel
fprintf('img2: width: %d, height: %d, and channel(depth): %d', w, h, c);
```

```
img2: width: 291, height: 240, and channel(depth): 3
```

Q.003: Please explain why c is changed.

Q.004: Please explain the difference between above two different methods for inserting color circles.

```
clear w h c

sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'autumn.tif'); % image filename
img = imread(filenameImg); % read an image

% When you say the location of the pixel, the order is width and height.
pxVal1 = img(50,150,:);
```

```
fprintf('pixel value at (150, 50): R: %d, G: %d, B: %d', pxVal1(1), pxVal1(2), pxVal1(3))
```

```
pixel value at (150, 50): R: 236, G: 226, B: 209
```

```
% Opencv has the BGR color format, not RGB
% https://www.learnopencv.com/why-does-opencv-use-bgr-color-format/

pxVal2 = impixel(img,150, 50); % the order is different from the above.
fprintf('pixel value at (150, 50): R: %d, G: %d, B: %d', pxVal2(1), pxVal2(2), pxVal2(3))
```

```
pixel value at (150, 50): R: 236, G: 226, B: 209
```

Grayscale image

```
clear; close all; clc;

sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'pout.tif'); % image filename
img = imread(filenameImg); % read an image

whosImg = whos('img');
fprintf('numeric type of img: %s', whosImg.class)
```

```
numeric type of img: uint8
```

```
% 0 is black and 255 is white
grey0 = ones(300,300,'uint8') * 0;
grey100 = ones(300,300,'uint8') * 100;
grey200 = ones(300,300,'uint8') * 200;

figure(1);
subplot(131);imshow(grey0);
subplot(132);imshow(grey100);
subplot(133);imshow(grey200);
```

```
grey300 = ones(300,300,'uint8') * 300;
if isequal(grey100(10,10), 100); disp('True'); else; disp('False'); end
```

```
True
```

```
if isequal(grey200(10,10), 200); disp('True'); else; disp('False'); end
```

```
True
```

```
if isequal(grey300(10,10), 300); disp('True'); else; disp('False'); end
```

```
False
```

Q.005: Please compare these results. Why does the third one produce "False"?

```
% check the size of the image
fprintf('bytes of the image: %d', whosImg.bytes)
```

```
bytes of the image: 69840
```

```
% 1 byte = 8 bits (8 bits can represet 256 values)
fprintf('bytes of the image: %d', whosImg.size(1)* whosImg.size(2)* 1);
```

```
bytes of the image: 69840
```

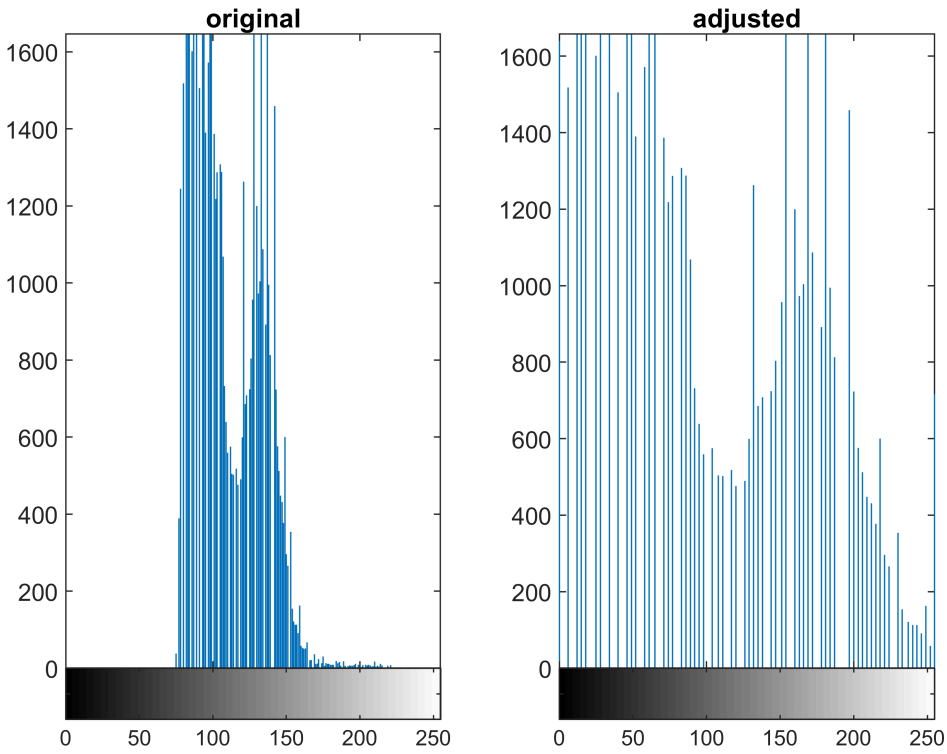
```
fprintf('bytes of the image: %d', prod(whosImg.size)* 1); % use of prod
```

```
bytes of the image: 69840
```

Q.006: What is byte and bit?

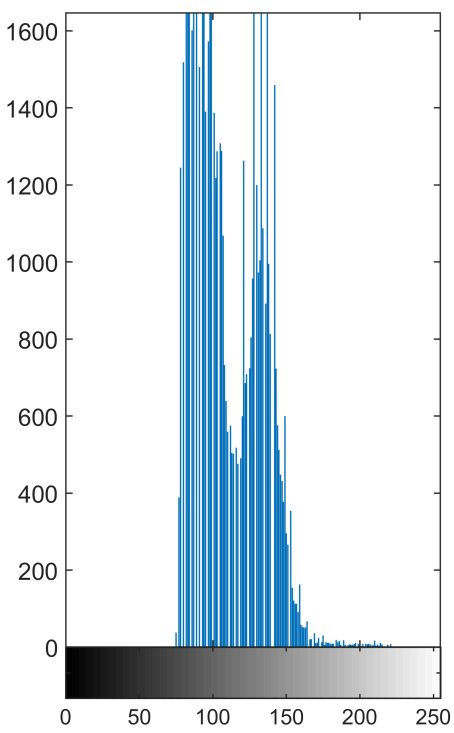
```
% image historgram
% See 1-8 in Image Processing Toolbox™ User's Guide
% https://www.mathworks.com/help/pdf_doc/images/images_tb.pdf
% We are going to adjust contrast of the image, making them bright!
figure;
subplot(121); imshow(img); title('original')
imgj = imadjust(img);
subplot(122); imshow(imgj); title('adjusted')
```

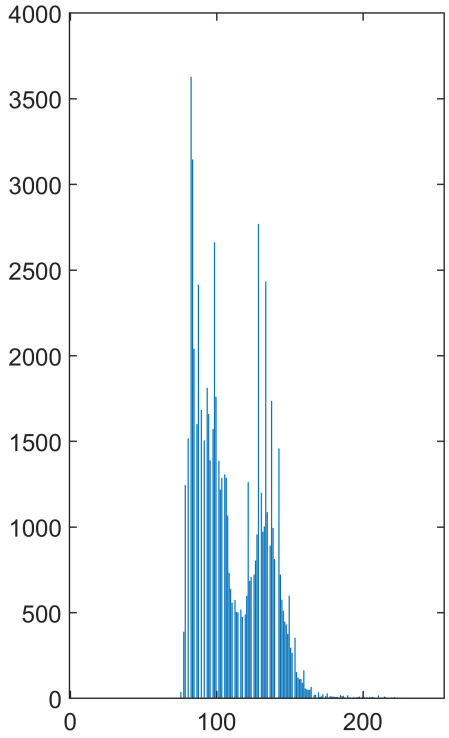

```
figure;
subplot(121); imhist(img); title('original')
imgj = imadjust(img);
subplot(122); imhist(imgj); title('adjusted')
```



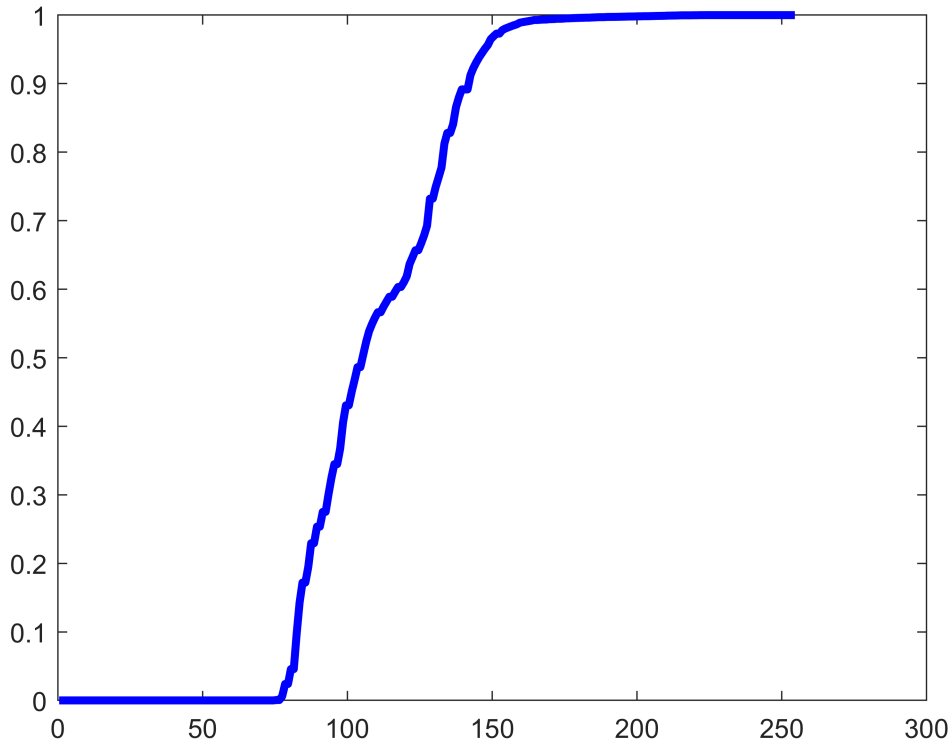

```
% How to adjust contrast without using imadjust
hist_edges = 0:1:255;
N = histcounts(img(:), hist_edges);
centers = (hist_edges(1:end-1)+hist_edges(2:end))/2;

figure;
subplot(121); imhist(img(:));
subplot(122); bar(centers, N); % they are same
```






```
% contrast adjustment
B = cumsum(N)./sum(N); % cumulative distribution
figure; plot(centers, B, 'linewidth', 3, 'color', [0 0 1]);
```




```
% please refer imadjust
indMin = find(B<0.01, 1, 'last'); % low_in
indMax = find(B>=0.99, 1, 'first'); % high_in

fprintf('new range from %d to %d', indMin, indMax);
```

```
new range from 78 to 162
```

```
% Here is how we adjust pixel values
% You are going to change each of pixel values following the above rule
% Old value i in the range of 0 ~ indMin => New values: 0
% Old value i in the range of indMin+1 ... indMax-2 => New values: 255/(indMax-indMin-1).*(img
% Old value i in the range of indMax-1 ~ 255 => New values: 255

% How to design your code?
% Here are some ways to write your code.

% method 0: very commmon
imgJ0 = zeros(size(img), 'uint8');
tic;
for ii=1:size(img,1)
    for jj=1:size(img,2)
        if img(ii,jj)<= indMin
            imgJ0(ii,jj) = 0;
        elseif img(ii,jj)>= (indMax-1)
```

```
            imgJ0(ii,jj) = 255;
        else
            imgJ0(ii,jj) = 255/(indMax-indMin-1).*(img(ii,jj)-indMin);
        end
    end
end
time0 = toc;

% method 1
tic;
imgJ1 = (255/(indMax-indMin-1).*(img-indMin)); % You must understand the type conversion in Ma
imgJ1(img<=indMin) = 0;
imgJ1(img>=indMax) = 255;
time1 = toc;

% See this link: https://www.mathworks.com/help/matlab/matlab_prog/integers.html
% "Arithmetic operations that involve both integers and floating-point always result in an inte
% MATLAB rounds the result, when necessary, according to the default rounding algorithm."
% this is different from C, C++
% a "double" type is converted to a "uint8" type.

if isequal(imgJ0, imgJ1); answ = 'True'; else; answ = 'False'; end
fprintf('Are imgJ1 and imgJ2 equal? %s', answ);
```

```
Are imgJ1 and imgJ2 equal? True
```

```
fprintf('Compare the computation time: %f vs %f', time0, time1); % the second one is much faste
```

```
Compare the computation time: 0.857382 vs 0.028258
```

```
% actually, method2 is much more readable and faster.
```

Spatial resolution

```
sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'pout.tif'); % image filename
img = imread(filenameImg); % read an image

img1 = imresize(img,2,'method','nearest');
img1r = repelem(img,2,2);
```

Q.007: Please check what the 'nearest' option is. What is it?

```
if isequal(img1, img1r); answ = 'True'; else; answ = 'False'; end
fprintf('Are img1 and img1r equal? %s', answ);
```

```
Are img1 and img1r equal? True
```

```
img2 = imresize(img,0.5,'method','nearest');
```

```
figure;
[w,h,c] = size(img1); % w: width, h: height, c: channel
fprintf('img: width: %d, height: %d, and channel(depth): %d', w, h, c)
```

```
img: width: 582, height: 480, and channel(depth): 1
```

```
clear w h c

[w,h,c] = size(img2); % w: width, h: height, c: channel
fprintf('img: width: %d, height: %d, and channel(depth): %d', w, h, c)
```

```
img: width: 146, height: 120, and channel(depth): 1
```

```
clear w h c
```

Q.008: Please check that how the code reproduce or reduce the pixels in the original image.

```
img(1:4,1:4)
```

```
ans = 4×4 uint8 matrix
   107   108   107   106
   109   106   108   107
   107   106   110   110
   106   107   108   108
```

```
img1(1:8,1:8)
```

```
ans = 8×8 uint8 matrix
   107   107   108   108   107   107   106   106
   107   107   108   108   107   107   106   106
   109   109   106   106   108   108   107   107
   109   109   106   106   108   108   107   107
   107   107   106   106   110   110   110   110
   107   107   106   106   110   110   110   110
   106   106   107   107   108   108   108   108
   106   106   107   107   108   108   108   108
```

```
img2(1:2,1:2)
```

```
ans = 2×2 uint8 matrix
   106   107
   107   108
```

```
% same with the above
[img(2,2) img(2,4);img(4,2) img(4,4)]
```

```
ans = 2×2 uint8 matrix
   106   107
   107   108
```

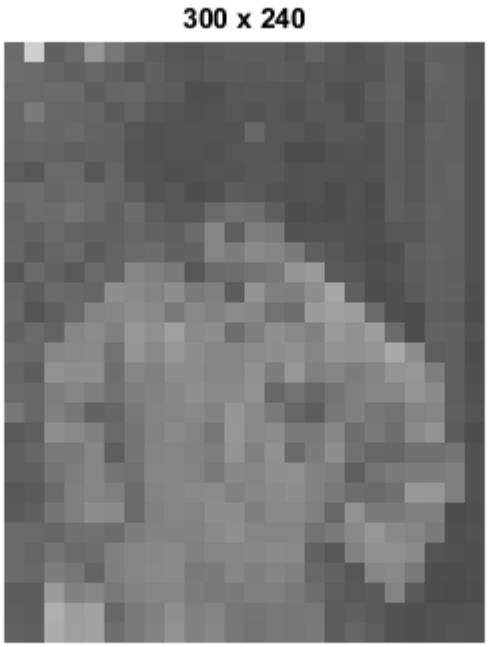
```
figure;imshow(img);title(sprintf('%d x %d', size(img)))
```

291 x 240

```
figure;img01 = imresize(img,0.1,'method','nearest'); imshow(img01);title(sprintf('%d x %d', si
```

30 x 24

```
figure;img10 = imresize(img01,10,'method','nearest'); imshow(img10);title(sprintf('%d x %d', s
```




```
figure;img01_10 = imresize(imresize(img,0.1),10); imshow(img01_10);title(sprintf('%d x %d', si
```

Q.009: Are high resolution images always high quality? Do high-quality photos have high resolution?

Color images

```
sampImgDir = 'img'; % directory of sample images
filenameImgC = fullfile(sampImgDir, 'autumn.tif'); % image filename
imgC = imread(filenameImgC); % read an image

imSize = [size(imgC,1) size(imgC,2)];
imgCR = imgC(:,:,1);
imgCG = imgC(:,:,2);
imgCB = imgC(:,:,3);

figure;
subplot(121); imshow(imgCR);
subplot(122); imshow(rgb2gray(imgC));
```

```
% three dimension matrix
imgCRImg = cat(3,imgCR, zeros(imSize, 'uint8'), zeros(imSize, 'uint8')); % how to use cat
imgCGImg = cat(3, zeros(imSize, 'uint8'), imgCG, zeros(imSize, 'uint8'));
imgCBImg = cat(3, zeros(imSize, 'uint8'), zeros(imSize, 'uint8'), imgCB);
subplot(131); imshow(imgCRImg);
```

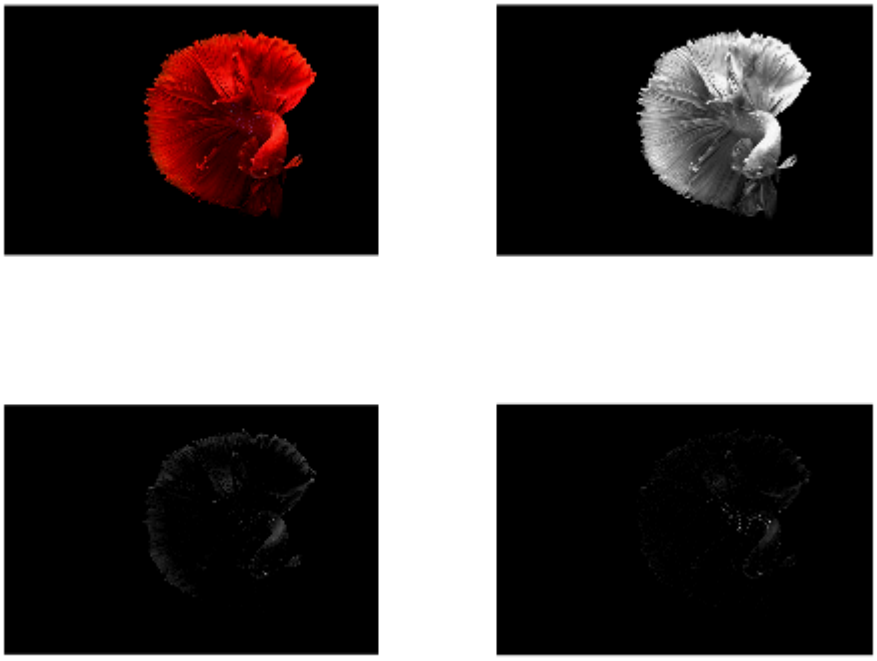
```
subplot(132); imshow(imgCGImg);
subplot(133); imshow(imgCBImg);
```

```
figure; imshow(imgCRImg+imgCGImg+imgCBImg);
```

```
sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'pexels-photo-325044.jpeg'); % image filename
% https://www.pexels.com/photo/close-up-of-fish-over-black-background-325044/
img = imread(filenameImg); % read an image

figure;
```

```
subplot(221); imshow(img);
subplot(222); imshow(img(:,:,1));
subplot(223); imshow(img(:,:,2));
subplot(224); imshow(img(:,:,3));
```



Q.010: What did you learn from this example?

Image file sizes

```
sampImgDir = 'img'; % directory of sample images

filenameImgC = fullfile(sampImgDir, 'autumn.tif'); % image filename
imgC = imread(filenameImgC); % read an image

filenameImgG = fullfile(sampImgDir, 'pout.tif'); % image filename
imgG = imread(filenameImgG); % read an image

whosImgC = whos('imgC');
infoImgC = imfinfo(filenameImgC);

whosImgG = whos('imgG');
infoImgG = imfinfo(filenameImgG);

bit2Byte = 8;
pixelByte = (infoImgC.BitDepth)/bit2Byte;
imgCByte = (infoImgC.Width) * (infoImgC.Height) * pixelByte;
```

```
if isequal(whosImgC.bytes, imgCByte); disp('imgC: True'); else; disp('imgC: False'); end
```

```
imgC: True
```

```
pixelByte = (infoImgG.BitDepth)/bit2Byte;
imgGByte = (infoImgG.Width) * (infoImgG.Height) * pixelByte;
if isequal(whosImgG.bytes, imgGByte); disp('imgG: True'); else; disp('imgG: False'); end
```

```
imgG: True
```

```
% tif: loseless compression
fprintf('The size of imgC in bytes: %d and its size in bytes after loading into workspace: %d'
```

```
The size of imgC in bytes: 214108 and its size in bytes after loading into workspace: 213210
```

```
fprintf('The size of imgG in bytes: %d and its size in bytes after loading into workspace: %d'
```

```
The size of imgG in bytes: 69296 and its size in bytes after loading into workspace: 69840
```

Q.011: What is loseless image compresssion?

Q.012: With this result, please estimate the filesize of a 8,688 x 5,792 image stored as tif (lossless compression)?

```
% jpg image
filenameImgJPG = fullfile(sampImgDir, 'car1.jpg'); % image filename
imgJPG = imread(filenameImgJPG); % read an image

% double = 64bit
whosImgJPG = whos('imgJPG');
infoImgJPG = imfinfo(filenameImgJPG);

pixelByte = (infoImgJPG.BitDepth)/bit2Byte;
imgJPGByte = (infoImgJPG.Width) * (infoImgJPG.Height) * pixelByte;
if isequal(whosImgJPG.bytes, imgJPGByte); disp('imgJPG: True'); else; disp('imgJPG: False'); en
```

```
imgJPG: True
```

```
byte2MegaByte = 1024*1024;
imgJPGD = im2double(imgJPG);
whosImgJPGD = whos('imgJPGD');
fprintf('The size of imgJPG in megabytes: %f and its size in megabytes after loading into works
```

```
The size of imgJPG in megabytes: 4.721160 and its size in megabytes after loading into workspace: 23.418457
```

```
fprintf('The size of imgJPG (uint8) and imgJPGD (double) in megabytes after loading into worksp
```

```
The size of imgJPG (uint8) and imgJPGD (double) in megabytes after loading into workspace: 23.418457 and 187.347656
```

Q.013: What do you learn from this example?

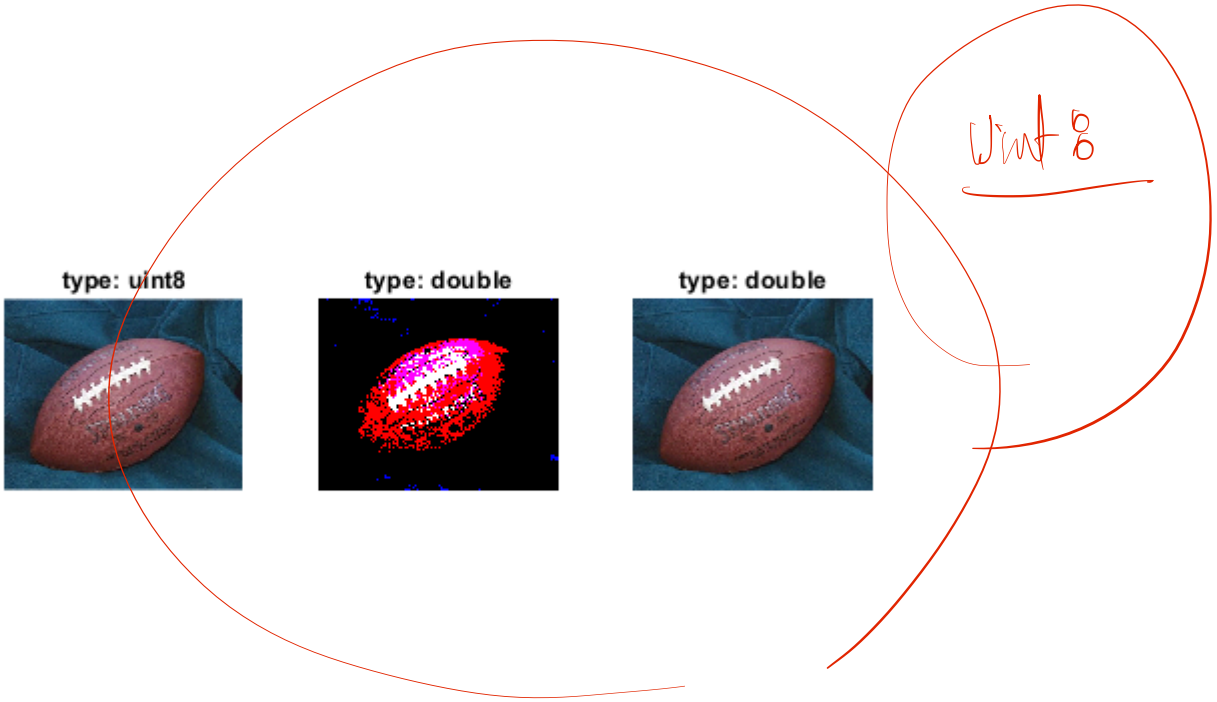
Understanding uint8 and double

You should use im2double for image arithmetic. Do not make mistake artithmetic operation on uint8 values. The values may be saturated without notice.

```
sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'football.jpg'); % image filename
img = imread(filenameImg); % read an image

img1 = double(img/255);
img2 = im2double(img);

figure;
subplot(131); imshow(img); title(sprintf('type: %s', class(img)))
subplot(132); imshow(img1); title(sprintf('type: %s', class(img1)))
subplot(133); imshow(img2); title(sprintf('type: %s', class(img2)))
```




```
if isequal(img, uint8(img2*255)); disp('True'); else; disp('False'); end
```

```
True
```

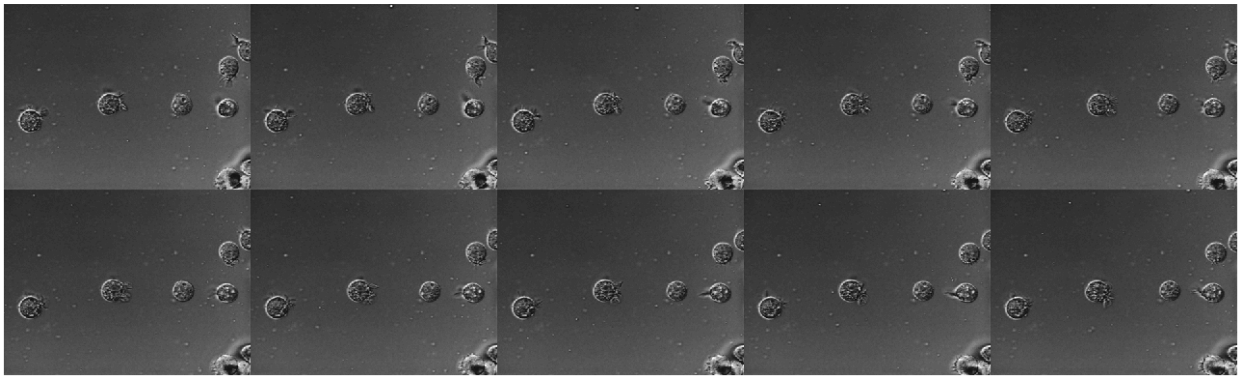
Q.014: What is the difference between im2double and double?

Q.015: What do you learn from this example?

Montage

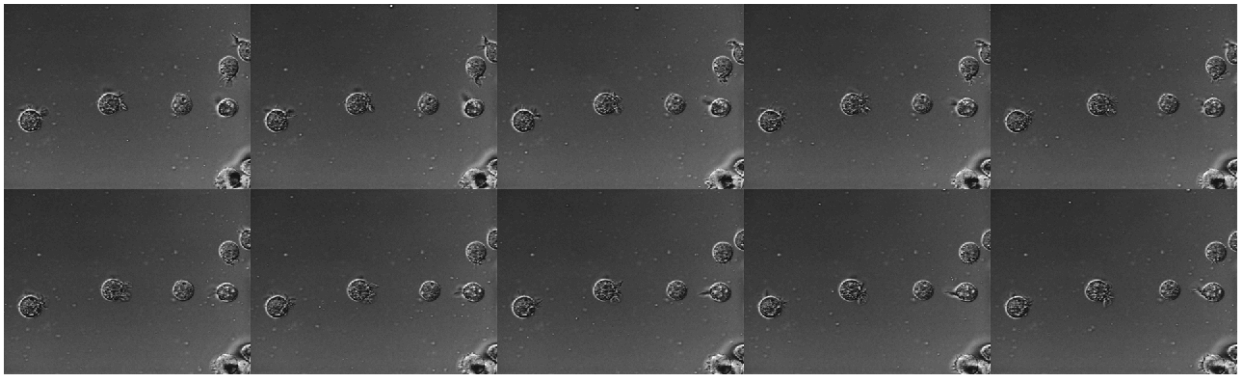
```
close all;
sampImgDir = 'img'; % directory of sample images
dirOutput = dir(fullfile(sampImgDir,'AT3_1m4_*.tif'));
fileNames = string({dirOutput.name});

figure(1)
montage(fileNames, 'Size', [2 5]);
```



```
figure(2)
imgList = cell(numel(fileNames),1);
for ii=1:numel(fileNames)
    imgList{ii} = imread(fileNames{ii});
end

montage(imgList, 'Size', [2 5]);
```



Q.016: Do you see what is the difference between them?

```
clearvars fileNames imgList
fileNames = cell(4,1);
fileNames{1} = fullfile(sampImgDir,'eSFRTestImage.jpg');
fileNames{2} = fullfile(sampImgDir,'indiancorn.jpg');
fileNames{3} = fullfile(sampImgDir,'coloredChips.png');
fileNames{4} = fullfile(sampImgDir,'kids.tif');

figure(3)
montage(fileNames, 'Size', [2 2]);
```

Write images

```
sampImgDir = 'img'; % directory of sample images
filenameImg = fullfile(sampImgDir, 'pout.tif'); % image filename
img = imread(filenameImg); % read an image

miscDir ='misc';
if ~exist(miscDir,'dir')
    mkdir(miscDir);
end

img2 = imresize(img,2);
imwrite(img2, fullfile(miscDir, 'img2.jpg'));

img2_re = imread(fullfile(miscDir, 'img2.jpg'));
```

```
[w,h,c] = size(img2); % w: width, h: height, c: channel
fprintf('img: width: %d, height: %d, and channel(depth): %d', w, h, c)
```

```
img: width: 582, height: 480, and channel(depth): 1
```

```
[w,h,c] = size(img2_re); % w: width, h: height, c: channel
fprintf('img: width: %d, height: %d, and channel(depth): %d', w, h, c)
```

```
img: width: 582, height: 480, and channel(depth): 1
```

```
% method 1
h=figure; imshow(img, 'Border', 'tight'); hold on;
plot(50, 150,'ro');
plot(150, 50,'bo'); hold off; axis off;
resolution = 150;
h.PaperUnits = 'inches';
h.PaperPositionMode = 'auto';
h.PaperPosition = [0 0 size(img,2)/resolution  size(img,1)/resolution];
print(fullfile(miscDir,'img3_1'),'-djpeg','-r150')
```

```
% method 2
img3 = insertShape(img, 'circle', [50 150 5], 'lineWidth', 1, 'Color', 'Red');
img3 = insertShape(img3, 'circle', [150 50 5], 'lineWidth', 1, 'Color', 'Blue');
imwrite(img3, fullfile(miscDir, 'img3_2.jpg'));

figure;
subplot(121); imshow(imread(fullfile(miscDir, 'img3_1.jpg')))
subplot(122); imshow(imread(fullfile(miscDir, 'img3_2.jpg')))
```

Q17: Please think about the difference between method 1 and method 2.

Exercise 001

Make a chaseboard (using black and white) with 800x800 pixels and plot it.

Exercise 002

1. **Read an image of 'color_patch.png' in the 'sample_free' folder**
2. **Display each of four square color patches in the first column of the image**
3. **Extract the region of *rect* from the image (*rect* = [101, 101, 50, 50]: *x, y, width, heigt*) and replace its color to blue. Please compare before and after processing the images.**

Exercise 003

1. **Load and show cameraman.tif in the 'sample_images' folder.**
2. **Roate the image 90, 180, 270° (counterclockwise)and show them.**
3. **Mirror the image in horizontal and vertical directions and show them.**
4. **Repeat this process using llama.jpg**