ENEL503 Computer Vision

Lab 1
Basic Image Representation and Processing with MATLAB

Due: 5:00pm, Wednesday, Feb. 8, 2023

Name: ______Christina Truong_____________________

UCID: ____30064426___________________________

Purpose

The purposes of this lab assignment are as follows:

1. Practice MATLAB image I/O techniques;
2. Manipulate MATLAB image show and plot formatting;
3. Analyze image characteristic values (κ) by implementing a Kappa Test Algorithm in MATLAB;
4. Analyze image similarity measures (σ) by implementing a Sigma Test Algorithm in MATLAB;
5. Become familiar with essential MATLAB programming skills.

Marks

Question	Mark allocation	Mark received
1	20	
2	20	
3	25	
4	25	
5	10	
Total	**100**	

General Instructions

To ensure consistent and efficient marking, a Word template will be provided for each ENEL 503 lab. Complete this Word template with your answers, MATLAB code, and plots as required. *Submit a PDF file titled by your full name* before the due date/time by email attachment to TA: Abbas Mahbod at abbas.mahbod1@ucalgary.ca.

Some questions require MATLAB code to be provided in related answers. Make sure that the code is commented sufficiently, but avoid being too verbose, in order to make the marking job for the TA as efficient as possible. Obscure code that is insufficiently or incorrectly commented will result in lost marks.

Equations in the report need to be represented in proper mathematical form using *MathType* or *MS Equation*. MathType equation editor is powerful and recommended. Hand-written math expressions are not encouraged.

MATLAB plots may be copied directly from the figure window of MATLAB by 'Edit > Copy Figure'. Then, you may paste the figure into the Word template. Color reports are encouraged due to the nature of this course.

1. (20) Practice image I/O, format transformation, size unification, and plotting with the set of six given color photos/images as follows. The images can be downloaded from course D2L site in the section: D2L/.../Lab Assignments/Lab1 Materials. It's noteworthy that formats and sizes of the given images may be different, and Fig12 is a composition (morphing) of Fig1 and Fig2.

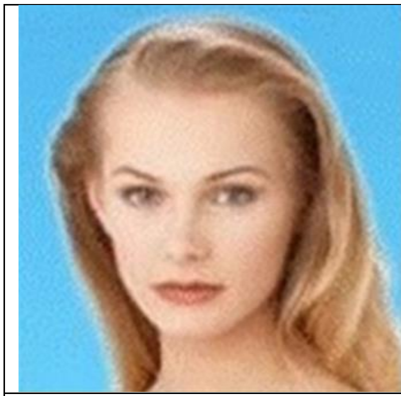
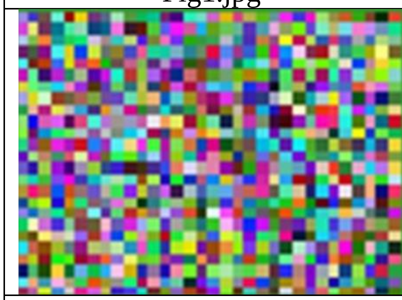
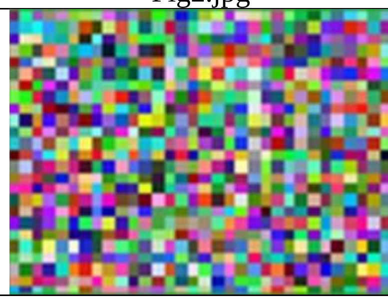


Fig1.jpg


Fig2.jpg


Fig12.jpg


Rand1.jpg


Rand2.jpg

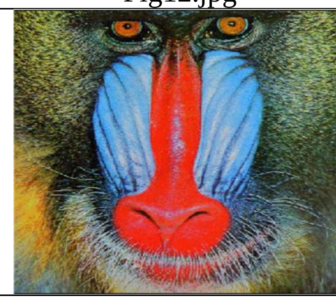

Mandrill.png

Fig. 1 Images for processing

a) (2) Load the six color images from your ENEL503 Lab1 Directory into the MATLAB programming environment.

b) (3) Convert the given images into grayscale.

c) (15) Unify the sizes of the given images to 500 x 500 pixels using `imresize` and then save them onto your hard disk using similar file names. Then, plot the six normalized gray images, respectively, by a MATLAB program with suitable titles.

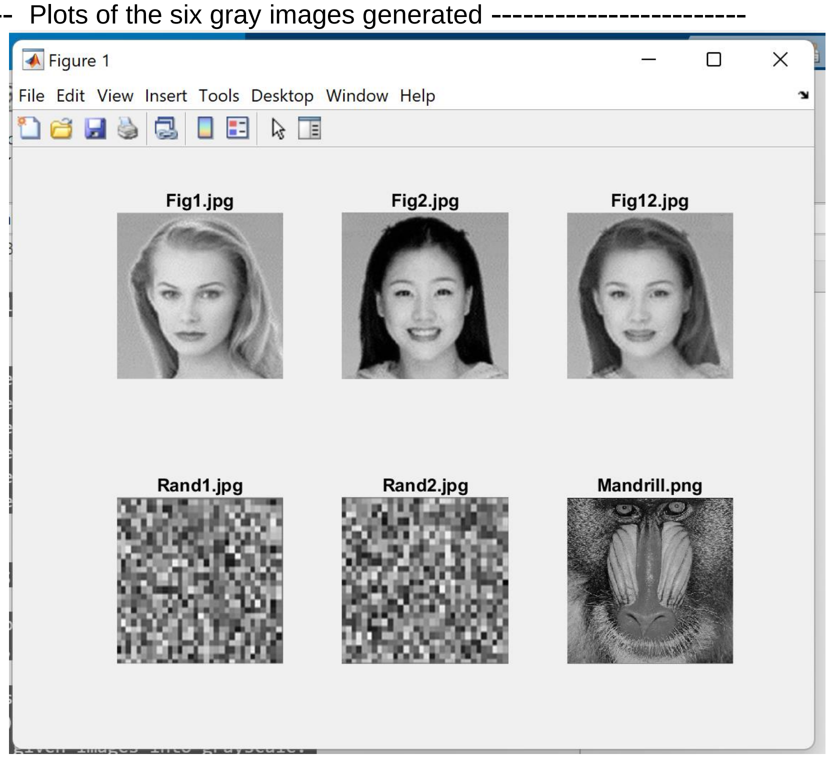
[**Hint:** In order to implement an iterative algorithm, use `Im = repmat({}, 6)` to initialize the date structure for holding the six target images such as `Im(1,1).image = imread('Fig1.jpg'); Im(1,6).image = imread('Mandrill.png'), etc.` [Other approaches may also be acceptable.]

--------------- MATLAB code for Steps (a) to (c) -----------------

```
%1a) Load the six color images from directory folder
Im = repmat({}, 6);
Im1 = Im;
Im1(1,1).image = imread('Fig1.jpg');
Im1(1,2).image = imread('Fig2.jpg');
Im1(1,3).image = imread('Fig12.jpg');
Im1(1,4).image = imread('Rand1.jpg');
```

```
Im1(1,5).image = imread('Rand2.jpg');
Im1(1,6).image = imread('Mandrill.png');

%gray image array
Im1G = repmat({}, 6);

%image names array to save as
ImName = {'Fig1.jpg','Fig2.jpg','Fig12.jpg','Rand1.jpg','Rand2.jpg','Mandrill.png'};

%iterate to apply resizing and gray scale to each image
for k = 1:length(Im1)
    %1b) convert the given images into grayscale.
    Im1G(1,k).image = rgb2gray(Im1(1,k).image);

    %1c) unify the sizes of the given images to 500 x 500 pixels
    Im1G(1,k).image = imresize(Im1G(1,k).image, [500 500]);

    %1c) Save onto hard disk. Naming files with "gray-" in the beginning
    %since if it's saved with the original name, grayscale conversion will
    %give error on second run since the image is already gray
    imwrite(Im1G(1,k).image, strcat("gray-",string(ImName(k))));

    %1c) plot the six normalized gray images
    subplot(2,3,k);

    %1c) suitable titles
    imshow(Im1G(1,k).image);
    title(string(ImName(k)));
end
```

-------------------- Plots of the six gray images generated ------------------------



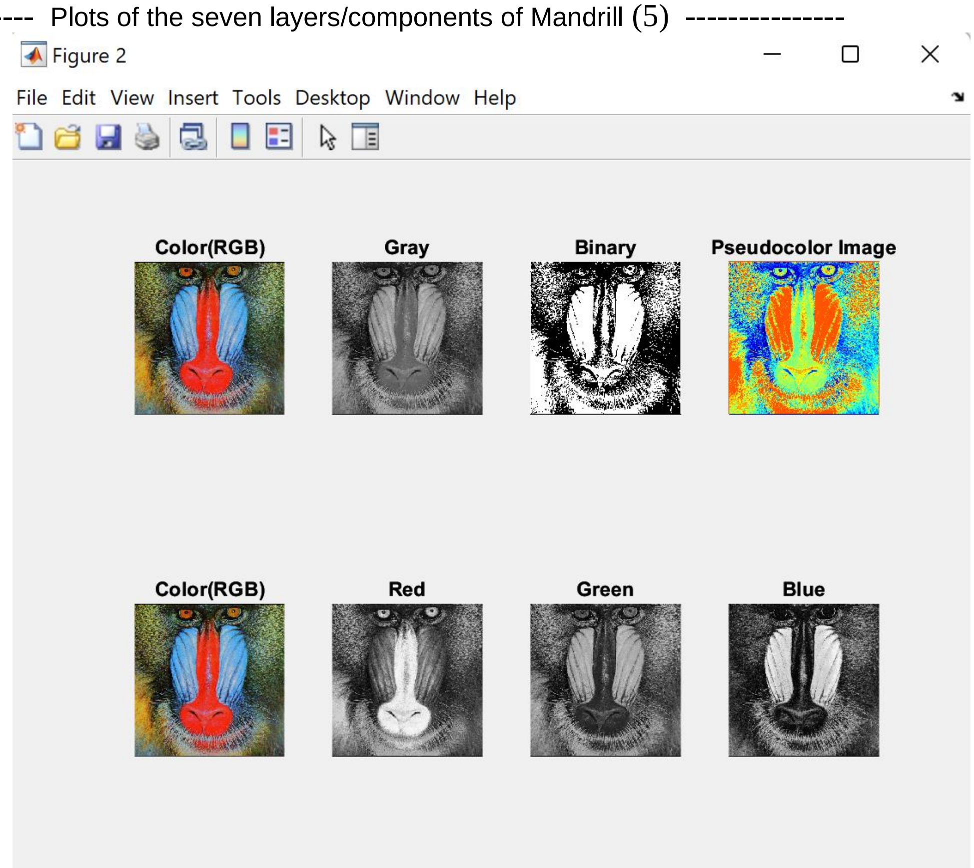
2. (20) In order to get familiar with image representation technologies and dataset structures, try to convert *Mandrill.png* into the following 7 formats: C/G/B/SC/R1/G2/B3, by developing a MATLAB function. The color format SC denotes the indexed pseudocolor scheme.

[**Hint:** Refer to L3-Section 4.2. The color image may be plotted for two times in the beginning of each row in a 2 x 4 subplot.]

------------------------------------ MATLAB code (15) --------------------------------

```
%2 convert Mandrill.png into the following 7 formats: C/G/B/SC/R1/G2/B3
ImC = imread('Mandrill.png');    %C
ImG = rgb2gray(ImC);             %G
ImB = imbinarize(ImG, 0.4);      %B
ImSC = grayslice(ImG, 16);       %SC
ImRed = ImC(:, :, 1);            %R1
ImGreen = ImC(:, :, 2);          %G2
ImBlue = ImC(:, :, 3);           %B3

figure
subplot (2,4,1), imshow (ImC), title('Color(RGB)');
subplot (2,4,2), imshow (ImG), title('Gray');
subplot (2,4,3), imshow (ImB), title('Binary');
subplot (2,4,4), imshow (ImSC, jet(10)), title('Pseudocolor Image');
subplot (2,4,5), imshow (ImC), title('Color(RGB)');
subplot (2,4,6), imshow (ImRed), title('Red');
subplot (2,4,7), imshow (ImGreen), title('Green');
subplot (2,4,8), imshow (ImBlue), title('Blue');
```

--------------- Plots of the seven layers/components of Mandrill (5) ---------------



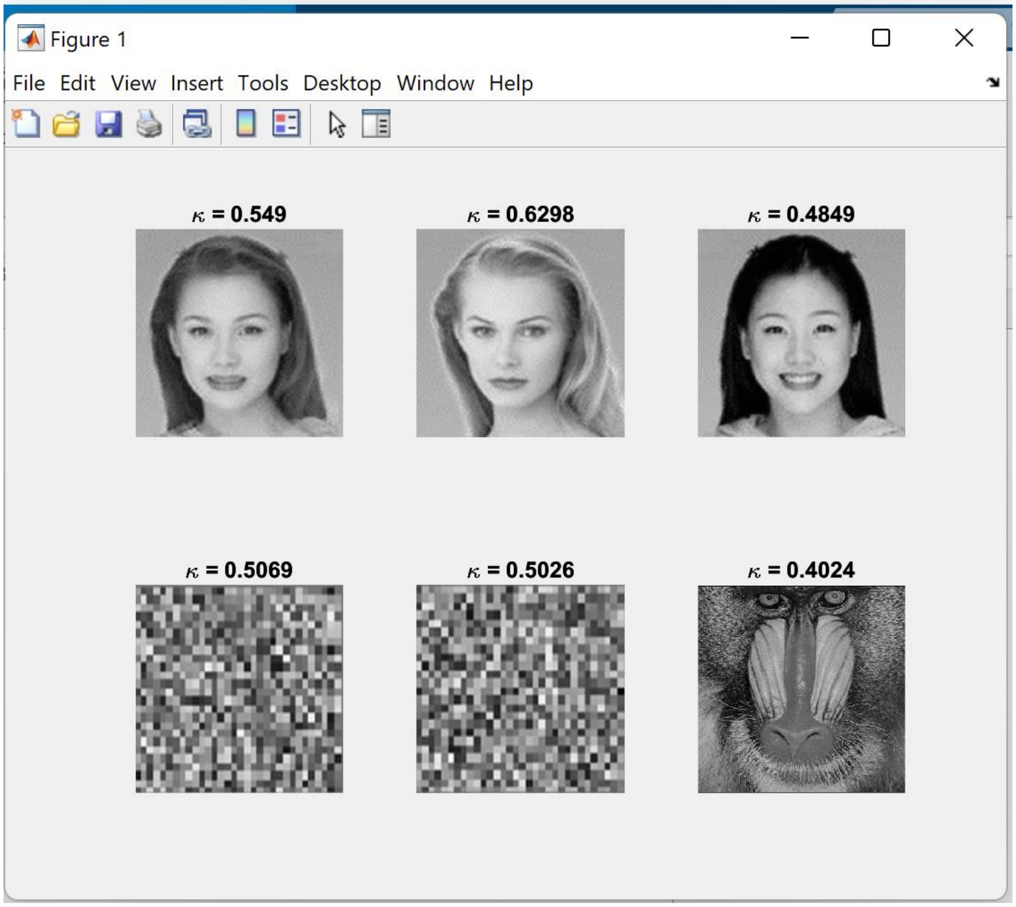
3. (25) Design a MATLAB function to analyze the *characteristic values* (κ) of the six gray-scale images as obtained and normalized in Problem (1). The algorithm will be based on the mathematical models in L3-Section 2.

[**Hint:** Reuse similar data structures designed for solving Problem (1).]

---------------------------------- MATLAB code (25) --------------------------------

```
function [Kappa] = CharacteristicValues
X = 500; Y = X;
Kappa = zeros(1, 6);
Imk = zeros (X, Y);
Im = repmat({}, 6);
% resize to 500x500 and turn it gray
Im(1,1).image = imresize(rgb2gray(imread('Fig12.jpg')), [500 500]);
Im(1,2).image = imresize(rgb2gray(imread('Fig1.jpg')), [500 500]);
Im(1,3).image = imresize(rgb2gray(imread('Fig2.jpg')), [500 500]);
Im(1,4).image = imresize(rgb2gray(imread('Rand1.jpg')), [500 500]);
Im(1,5).image = imresize(rgb2gray(imread('Rand2.jpg')), [500 500]);
Im(1,6).image = imresize(rgb2gray(imread('Mandrill.png')), [500 500]);

% calculate kappa
for k = 1 : 1:length(Im)
    Sum = 0;
    for i = 1 : X
        for j = 1 : Y
        Sum = Sum + abs(double(Im(1,k).image(i,j)));
        end
    end
    Kappa(k) = Sum /(255*X*Y);
    %plot the six normalized gray images
    subplot(2,3,k);

    %suitable titles of Kappa
    imshow(Im(1,k).image);
    title(strcat("\kappa = ", string(round(Kappa(k),4))));
end
```

-------------------------- Show test results of kappas (5) --------------------------



4. (25) Design a MATLAB function for implementing *image similarity* (σ) analyses (slide L3-14) on each pair of the six gray-scale images as obtained and normalized in Problem (1). Multilayer (6 x 6) iterative code, rather than linear repetitions for the 36 pairs of them, are expected.

[**Hint:** a) Reuse similar data structures designed for solving Problem (1).
b) Consider the following code for implementing `abs(x-y)` in the equation of σ:

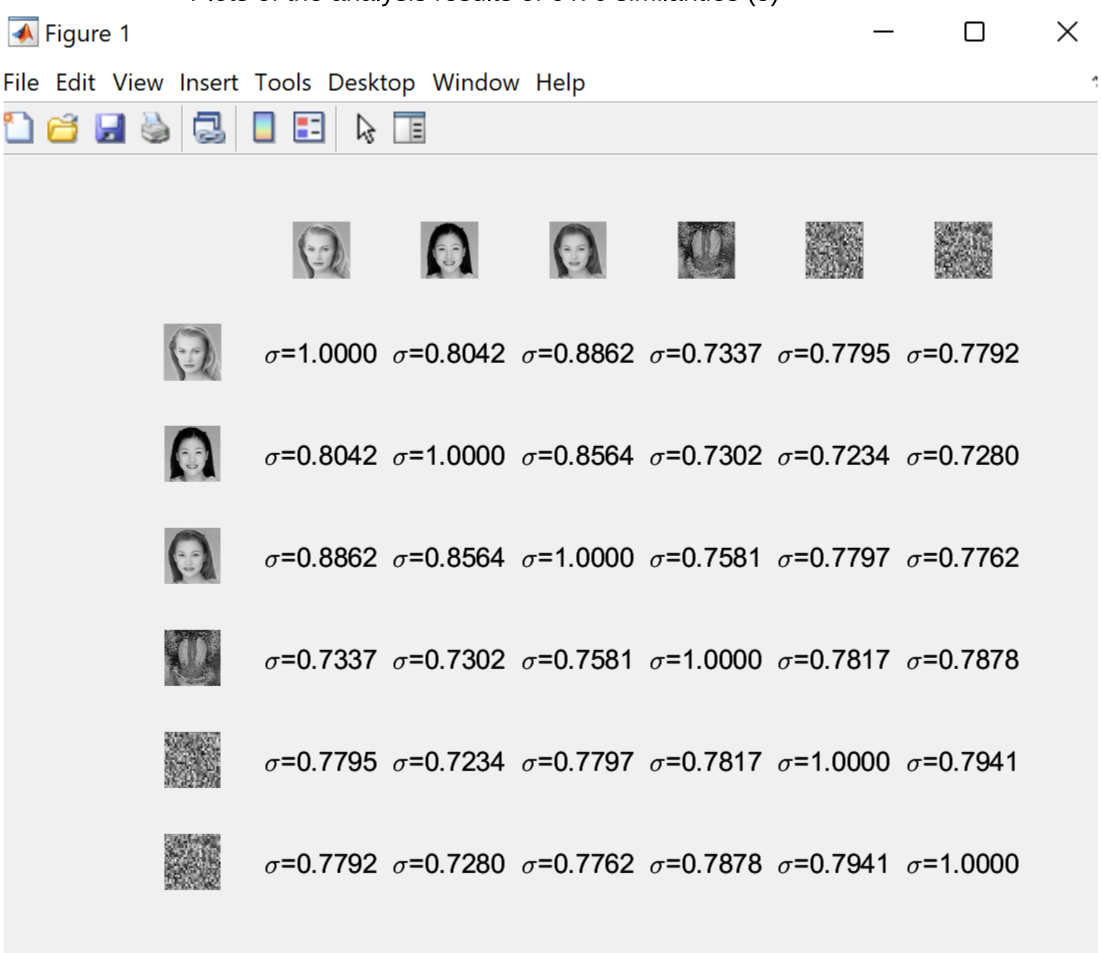
```
Sum = Sum + abs(double(Im1(i,j)) - double(Im2(i,j))); 
```
]

----------------------------------- MATLAB code (25) -----------------------------------

```
%4
function [Sim] = SimilarityDetermination
% Load Fig1, Fig2, and Fig12
X = 500; Y = X;
Sim = zeros(1, 7);
Imk = zeros (X, Y);
Im = repmat({}, 7);
Im(1,1).image = imresize(rgb2gray(imread('Fig1.jpg')), [500 500]);
Im(1,2).image = imresize(rgb2gray(imread('Fig2.jpg')), [500 500]);
Im(1,3).image = imresize(rgb2gray(imread('Fig12.jpg')), [500 500]);
Im(1,4).image = imresize(rgb2gray(imread('Mandrill.png')), [500 500]);
Im(1,5).image = imresize(rgb2gray(imread('Rand1.jpg')), [500 500]);
Im(1,6).image = imresize(rgb2gray(imread('Rand2.jpg')), [500 500]);
```

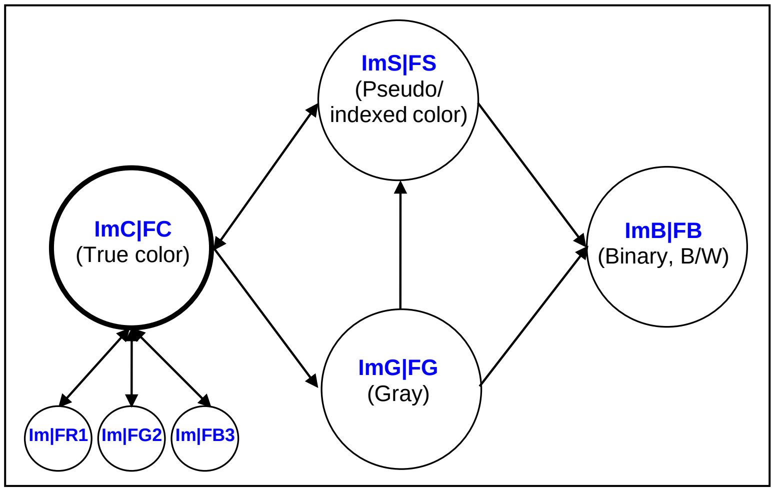
```
% Similarity Test
for m = 1 : 1:length(Im)
    subplot(7,7, m+1);
    imshow(Im(1,m).image);
    subplot(7,7, (m)*7+1);
    imshow(Im(1,m).image);
    for k = 1 : 1:length(Im)
        Imk = Im(1,k).image;
        Sum = 0;
        for i = 1 : X
            for j = 1 : Y
            Sum = Sum + abs(double(Im(1,m).image(i,j))-double(Imk(i,j)));
            end
            Sim(k) = 1 - Sum /(255*X*Y);
        end
        subplot(7,7, (m)*7+(k+1));
        axis off;
        text(0.5,0.5,
sprintf("\\sigma=%0.4f",Sim(k)),"HorizontalAlignment","center");
    end
end
```

---------------- Plots of the analysis results of 6 x 6 similarities (5) ------------------



5. (10) Complete the summary of the framework of MATLAB color scheme transformations in the following figure by referring to the slides in Lectures 3 and 2, as well as by surveying the related MATLAB toolboxes. Fill in the correct MATLAB function for each directed edge. Note there may be multiple built-in function for a certain transformation, and some edges are bidirectional.

[**Hint:** Double click on the figure below to open it for inserting your answers.]



------------------ Plots of expected answers (10) -------------------