ENEL503 Computer Vision

Lab 2
Mathematical, Geometrical and Histogram Operations on Images

Due: 5:00pm, Wednesday, Feb. 28, 2023

Name: _____Christina Truong_____

UCID: _____30064426_____

Purpose

The purposes of this lab assignment are as follows:

1. Practice MATLAB image processing by math, geometric, and tonicity operations;
2. Apply typical principles and algorithms for image processing and analytic ability;
3. Improve programming skills for computer vision in MATLAB;
4. Become familiar with MATLAB toll boxes for real-world problem solving.

Marks

Question	Mark allocation	Mark received
1	20	
2	30	
3	20	
4	30	
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 hard copy by the due date in the assignment drop boxes on the second floor of ICT Building. Also submit an electronic copy of your lab report by email attachment to the TA, Thoshara Malathinawar at thosharamalathinawar@ucalgary.ca .

Some questions require for MATLAB code to be inserted. 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 the equation editor in Word (under Insert > Object > Microsoft equation). The MathType equation editor is highly recommended. Hand written math expressions are not expected.

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

1. (10) Let ImG, ImC, ImR1, ImG2, and ImB3 represent a gray, color, red, green, and blue representation of an arbitrary image, respectively. Try to prove ImG = ImGs where:

$$\text{ImGs} = 0.2989*\text{ImR1} + 0.5870*\text{ImG2} + 0.1140*\text{ImB3} \quad (1)$$

$$\text{ImG} = \text{rgb2gray}(\text{ImC}) \quad (2)$$

by a MATLAB program on the given color image *Mandrill.png* as provided in the Lab 2 Materials site on D2L.

[**Hint:** The proof may be given by imperially showing that the difference of characteristic values, $\delta(\text{ImG}, \text{ImGws}) \rightarrow 0$, based on your solution for Question 4 in Lab 1.]

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

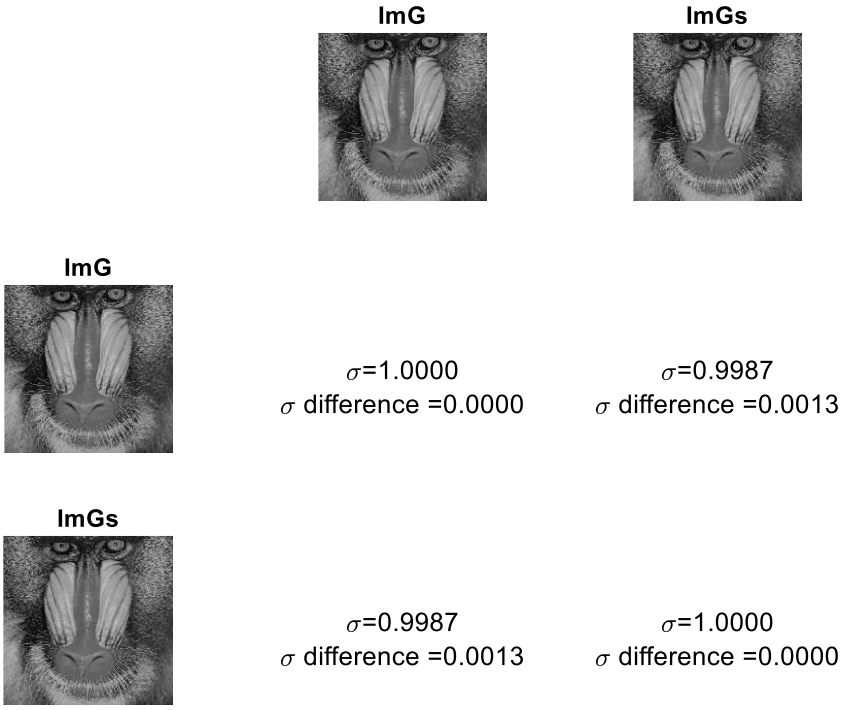
```
ImC = imresize(imread('Mandrill.png'), [500 500]);
ImG = rgb2gray(ImC);            %G
ImR1 = ImC(:, :, 1);            %R1
ImG2 = ImC(:, :, 2);            %G2
ImB3 = ImC(:, :, 3);            %B3

ImGs = 0.2989*ImR1 + 0.5870*ImG2 + 0.1140*ImB3;

X = 500; Y = X;
Sim = zeros(1, 7);
Imk = zeros (X, Y);
Im = repmat({}, 2);
Im(1,1).image = ImG;
Im(1,2).image = ImGs;

% Similarity Test
for m = 1 : 1:length(Im)
    subplot(3,3, m+1);
    imshow(Im(1,m).image);
    if(m == 1); title("ImG"); else; title("ImGs");end
    subplot(3,3, (m)*3+1);
    imshow(Im(1,m).image);
    if(m == 1); title("ImG"); else; title("ImGs");end
    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(3,3, (m)*3+(k+1));
        axis off;
        text(0.5,0.5,
sprintf("\\sigma=%0.4f",Sim(k)),"HorizontalAlignment","center");
        text(0.5,0.5, sprintf("\n\n\\sigma difference =%0.4f",1-
Sim(k)),"HorizontalAlignment","center");
    end
end
```

---------- Plots of ImG, ImGs, and the difference value (5) ----------



2. (30) According to the principle of mathematical operations, any image may be processed at either the frame level or pixel level, fundamentally the later. Apply this principle to image morphing technologies in MATLAB for merging the given photos *Im1.jpg* and *Im2.jpg* in order to generate a series of fusions of their sequential morphs.

a) (10). Morphing at frame level according to slide L4-40, i.e.:

$$\text{ImMorphSeries|FC} = \underset{k=1}{\overset{n}{R}} (\text{Im2|FC} + (1 - k/n) * (\text{Im1|FC-Im2|FC})) \qquad (3)$$

using the algebraic operation of Eq. 3 to generate the series of images morphing in MATLAB.

---------- MATLAB code (7) ----------

```
Im = repmat({}, 4);
Im1 = double(imread('Im21.jpg'));
Im2 =  double(imread('Im22.jpg'));

for k = 1:5
    subplot(1,5,k);
     morph = Im2 + (1.0-(k-1.0)/5.0)*(Im1-Im2);
     imshow(uint8(morph));
    title(string(k));
end
```

---------- Plots of the 4 morphing series from Im1 to Im5 (3) ----------

---

b) (20). Design an equivalent program to implement the same morphing algorithm of (a) by pixel-by-pixel operations of serial morphing. Report your code in MATLAB and plot the testing result.

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

```
Im1 = double(imread('Im21.jpg'));
Im2 = double(imread('Im22.jpg'));
for k = 1:5
    subplot(1,5,k);
    image_morph = double(zeros(size(Im1)));
    for l = 1:3
        for i = 1 :size(Im1,1)
            for j = 1 :size(Im1,2)
                image_morph(i,j,l) = Im2(i,j,l) + (1.0-(k-1.0)/5.0)*(Im1(i,j,l)-
Im2(i,j,l));
            end
        end
    end
    imshow(uint8(image_morph));
    title(string(k));
end
```

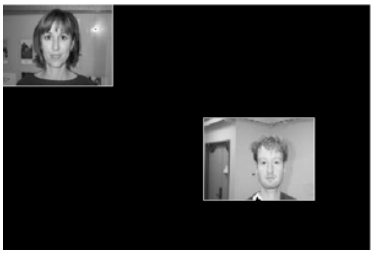
---------- Plots of the 4 morphing steps from Im1 to Im5 (5) ----------

```
Im1 = double(imread('Im21.jpg'));
Im2 = double(imread('Im22.jpg'));
for k = 1:5
    subplot(1,5,k);
```

```
    image_morph = double(zeros(size(Im1)));
    for l = 1:3
        for i = 1 :size(Im1,1)
            for j = 1 :size(Im1,2)
                image_morph(i,j,l) = Im2(i,j,l) + (1.0-(k-1.0)/5.0)*(Im1(i,j,l)-
Im2(i,j,l));
            end
        end
    end
    imshow(uint8(image_morph));
    title(string(k));
end
```

---

3. (20) Design a MATLAB function for implementing object detection by *spatial histograms* (SH) on the image *HS_Test.tif* as given below. Once the histogram is generated, try to analyze and locate these two objects by their coordinates at the upper-left and lower-right corners, i.e., O1((x1, y1),(x2, y2)) and O2((x3, y3),(x4, y4)), by the approximate values shown in it.



[**Hint:** Refers to L5(35-39).]

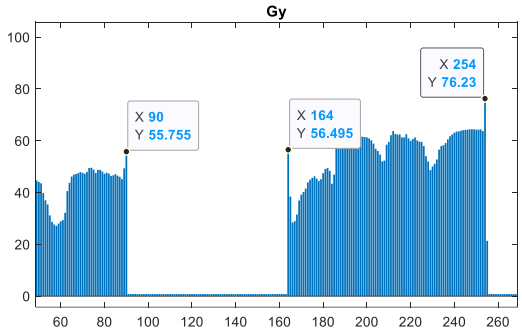
---------- MATLAB code (15) ----------

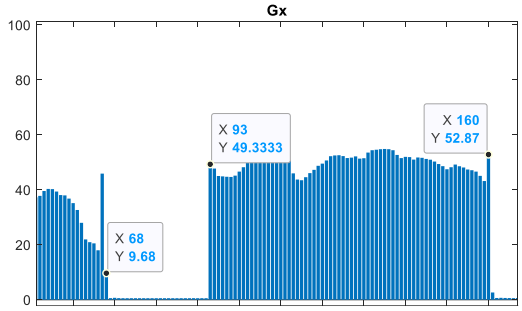
```
Im = imread('SH_Test.tif');
[X,Y] = size (Im);
Hx = zeros(1,X);
Hy = zeros (1, Y);
for i = 1 : X
    Sum = 0;
    for j = 1 : Y
        Sum = Sum + double(Im(i, j));
    end
    Hx(1,i) = Sum / Y;
end

for j = 1 : Y
    Sum = 0;
    for i = 1 : X
        Sum = Sum + double(Im(i, j));
    end
    Hy(1,j) = Sum / X;
end

subplot(2,2, 2), plot(1:Y,Hy), title("Gy");
subplot(2,2, 3),plot(1:X,Hx), title("Gx");
subplot(2,2, 4), imshow(Im), title("SHTest.tif");
```

---------- Plot the spatial histograms and analyze the location of the two objects (5) ----------





O1((0, 0),(67, 90)) and O2((93, 154),(160, 264))

---

4. (30) In object detection technologies, there is often a need to remove the shadow from the real object in an image in order to actually locate the position of the object in the frame. For this purpose, the degradation of the target's image quality is not the major concern as shown in the given case study on *Car1.jpg*.

Develop a MATLAB function for detecting the shadow around the car and try to remove the shadow by resetting its pixel value as the same of the surrounding background. Your program will need to automatically test the average shadow and background values in sample boxes of 4 x 4 and 10 x 10 pixels, respectively.

[**Hint:** The spatial histogram technology as implemented in Solution 3 may be applied to guide the detection of the average shadow and background values at suitable sample places.]

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

```
I = imread('Car1.jpg');
INoShadow = I;
INoShadowGray = rgb2gray(I);
[X,Y] = size (Im);
```

```
SumShadow = 0;
for i = X-13:X-10
    for j = (Y/2):(Y/2)+4
        SumShadow = SumShadow + double(I(i,j));
    end
end
Tshadow = uint8(SumShadow/16);

%background
SumBackground = 0;
for i = X-20:X-11
    for j = Y-60:Y-51
        SumBackground = SumBackground  + double(I(i,j));
    end
end
Tbackground = uint8((SumBackground+50)/100);

for i = 90:180
    for j = 40:280
        if INoShadowGray(i,j) > Tshadow- 40 && INoShadowGray(i,j) < Tshadow
            INoShadowGray(i,j) = Tbackground;
            INoShadow(i,j,1) = Tbackground;
            INoShadow(i,j,2) = Tbackground;
            INoShadow(i,j,3) = Tbackground;
        end
    end
end
```

---------- Plots of the results color and gray pairs as follows (5) ----------

Car

Shadow Removed (G)

Shadow Removed (C)

Dark regions for X = [91,153] and Y = [42, 279]
