ENEL503 Computer Vision

Lab 3

Neighborhood, Morphological, and Filtering Operations on Images

Instructor: Dr. Yingxu Wang
TA: Thoshara Malathinawar (Thosharamalathinawar@ucalgary.ca)

ENG 203
Due: 5:00pm, Wednesday, March 13, 2024

Name: _Gustavo Da Costa Gomez______________________

UCID: _30085980____________________________________

Purposes

The purposes of this lab assignment are as follows:

1. Practice MATLAB image processing by geometrical, morphological, and filtering operations
2. Apply typical principles and algorithms for image processing and analyses;
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	15	
3	20	
4	10	
5	35	
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 an electronic copy of your lab report by email attachment to TA: Thoshara Malathinawar at thosharamalathinawar@ucalgary.ca

Some questions ask 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 also 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.

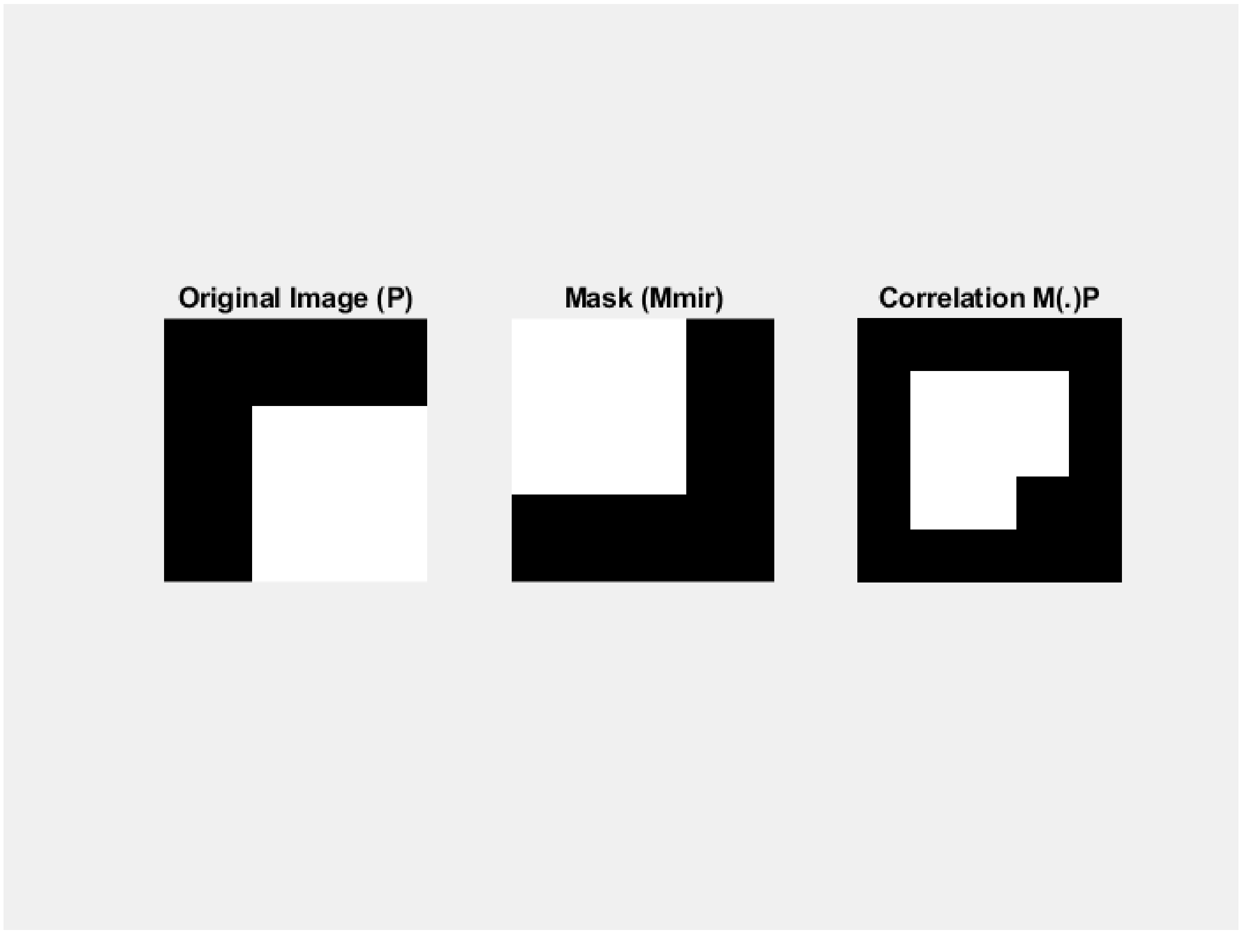
MATLAB plots can be copied directly from the figure window of MATLAB by ‘Edit > Copy Figure’. Then, you can paste the figure into the Word template. Color reports are encouraged due to the nature of this course.

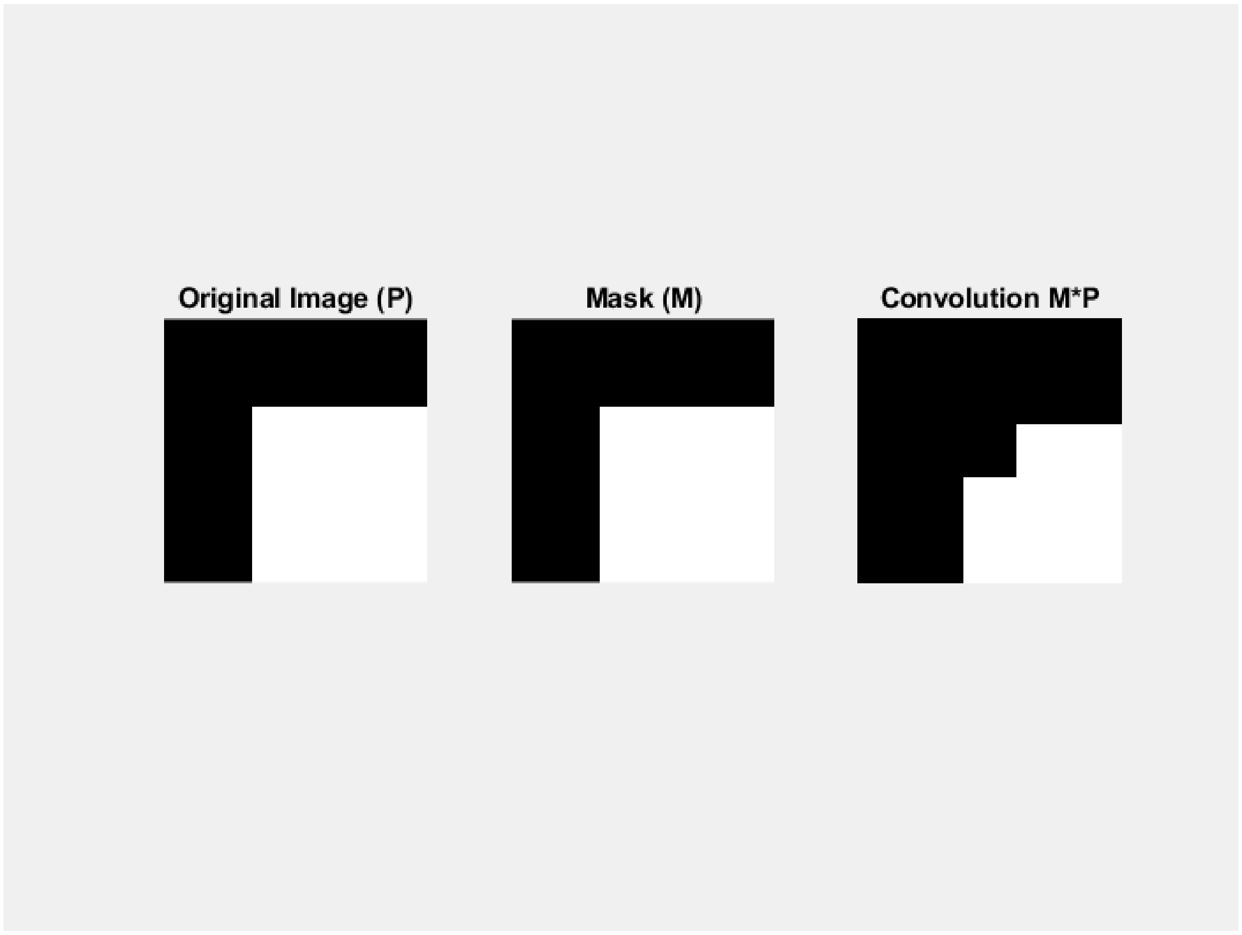
1. (20) The key concept and mathematical model of *masks (M)* play important roles in neighborhood, convolution, correlation, filter, and morphological operations. Try to empirically prove or test the following principles by your own MATLAB programs where the textbook might be incorrect.

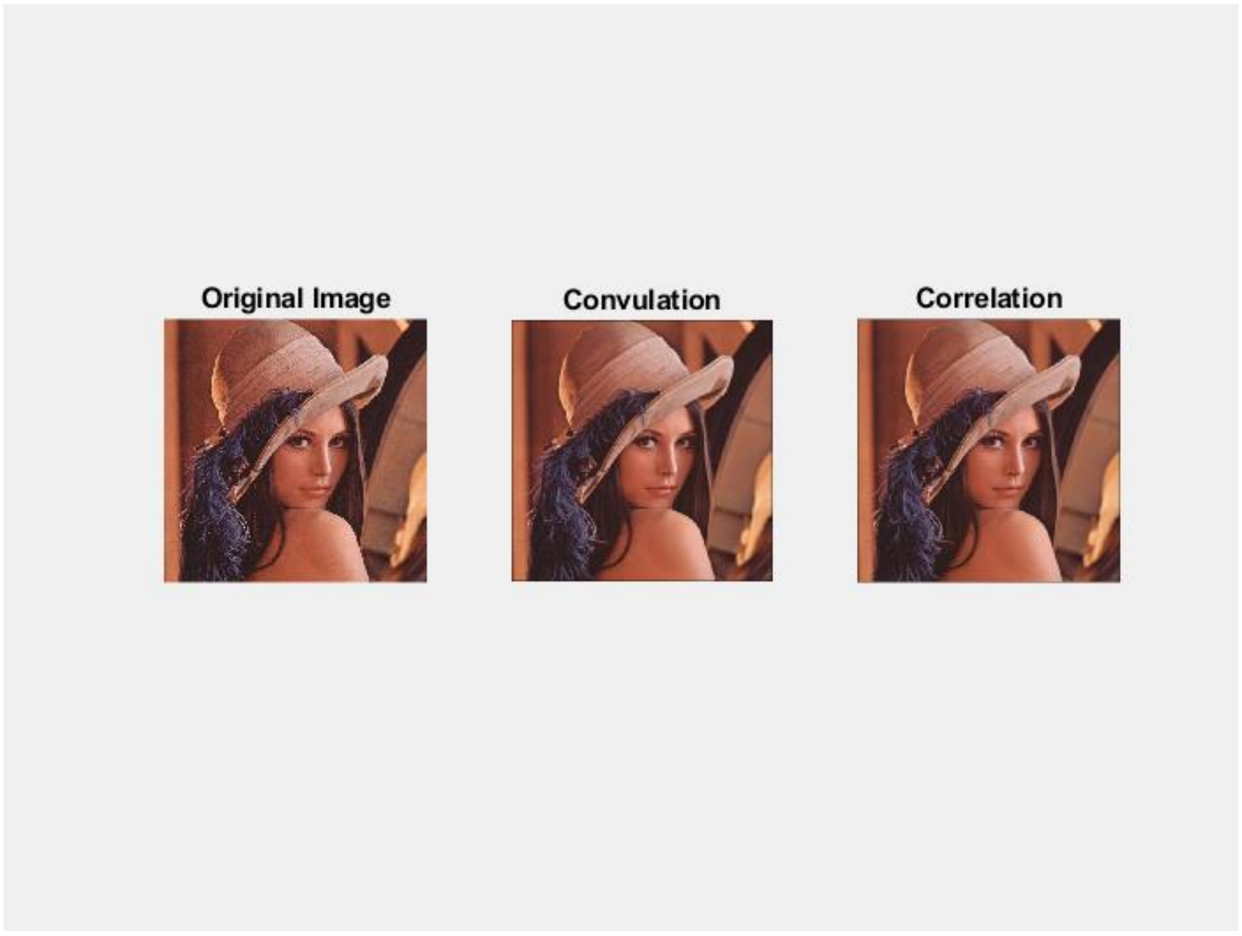
a) (10) Test the conceptual differences of image *Convolution* and *Correlation* using the codes in Section 4 of Lecture 6. Discuss on what conditions the MATLAB correlation and convolution operations will be equivalent.

-------------------- Q1.a: MATLAB code (4), plots (2), and analyses (4) ----------------------

```
%% ENEL 503 Lab 3
% Gustavo Da Costa Gomez, 30085980

% Clear workspace, close figures, and clear command window
clear
close all
clc

%% Question 1

% Q1 a)
% Correlation
P = [0 0 0; 0 5 8; 0 5 2];
M = [-2 -1 -0; -1 1 1; 0 1 1];
Mmir = [1 1 0; 1 1 -1 ; 0 -1 -2];
Corr = conv2(Mmir, P);

% Display and plot results
figure(1)
subplot(1,3,1), imshow(P), title('Original Image (P)');
subplot(1,3,2), imshow(Mmir), title('Mask (Mmir)');
subplot(1,3,3), imshow(Corr), title('Correlation M(.)P');

% Convolution
Conv = conv2(M, P);

% Display and plot results
figure(2)
subplot(1,3,1), imshow(P), title('Original Image (P)');
subplot(1,3,2), imshow(M), title('Mask (M)');
subplot(1,3,3), imshow(Conv), title('Convolution M*P');

% Lenna Example
lenna = imread("Lenna.jpg");
Mask = [1/9 1/9 1/9; 1/9 1/9 1/9; 1/9 1/9 1/9;];
Conv = imfilter(lenna, Mask, 'full', 'conv');
Corl = imfilter(lenna, Mask, 'full', 'corr');

% Display and plot results
figure(3)
subplot(1,3,1), imshow(lenna), title('Original Image');
subplot(1,3,2), imshow(Conv), title('Convolution');
subplot(1,3,3), imshow(Corl), title('Correlation');
```


Original Image (P)
Mask (Mmir)
Correlation M(.)P


Original Image (P)
Mask (M)
Convolution M*P



The condition in which the convolution and correlation operations will produce the same output is if the mask that is used is uniform.

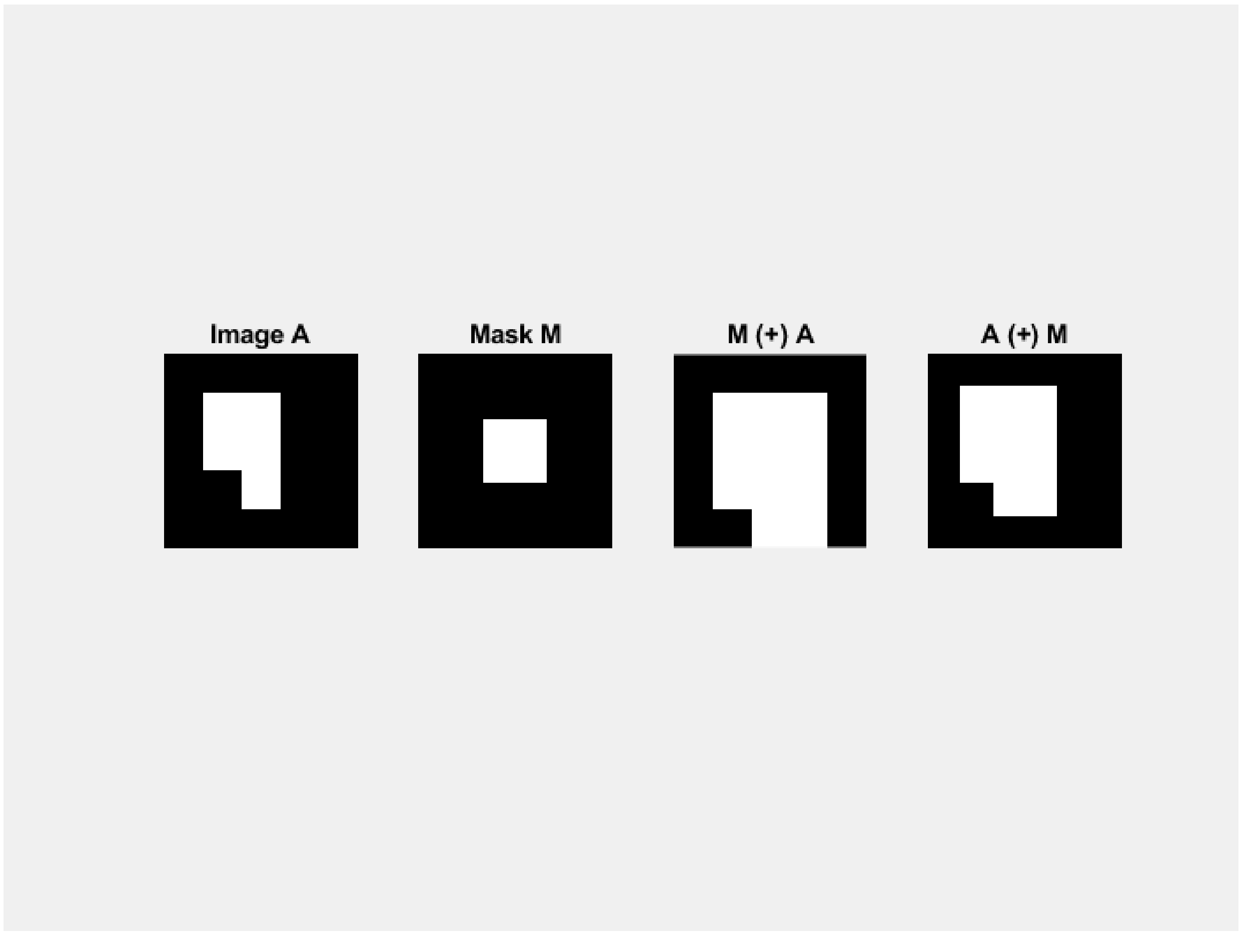
----

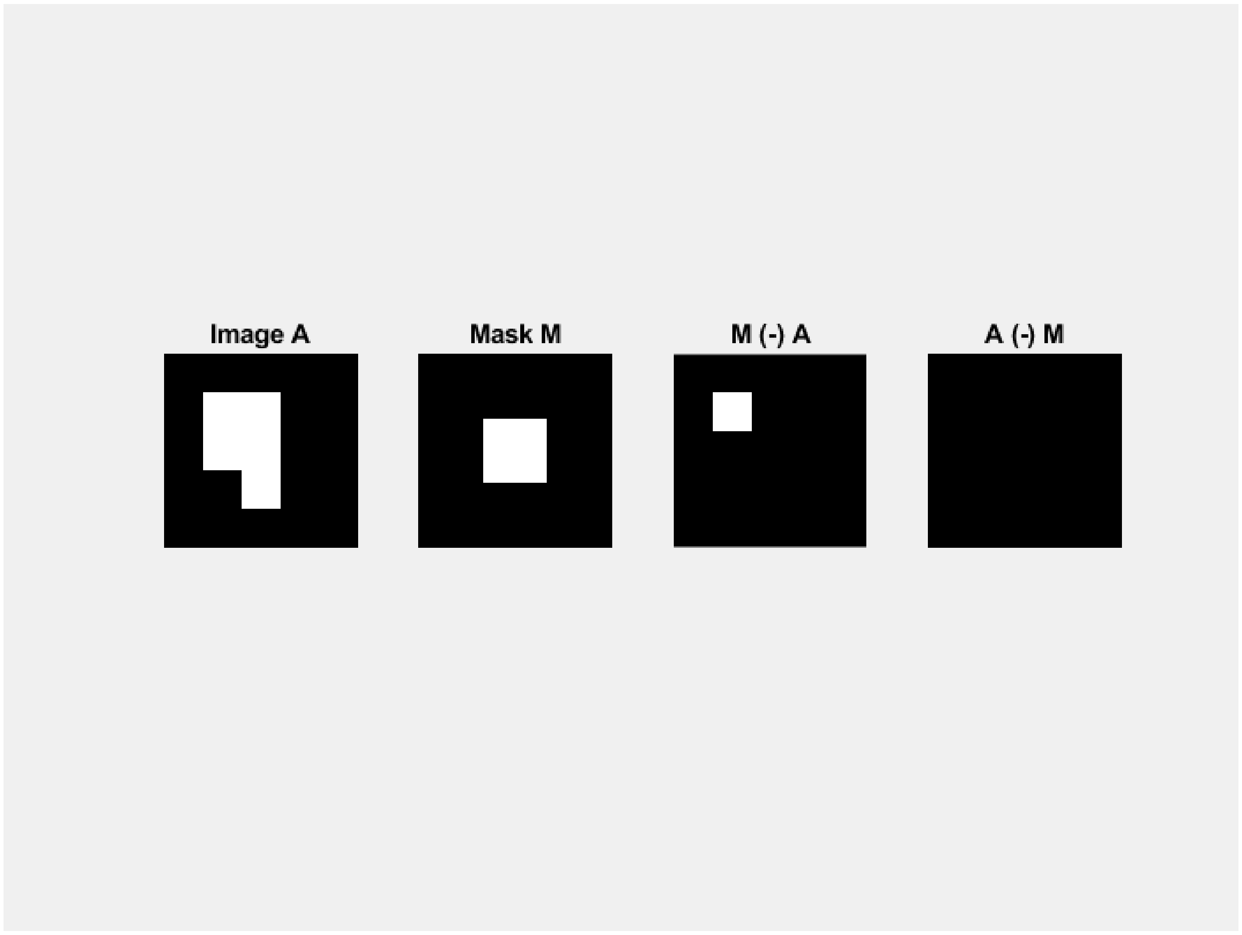
b) (5) Using the case studies in L7-15 and L7-17, test if the basic morphological operations *dilation* and *erosion* are *commutative*, i.e., if $A \oplus B = B \oplus A$ and A ! $B = B$! A?

-------------------- Q1.b: MATLAB code, plots, and analysis (5) ----------------------

```
% Q1 b)
% Dilation
A = [0 0 0 0 0; 0 1 1 0 0; 0 1 1 0 0; ...
    0 0 1 0 0; 0 0 0 0 0];
M = strel('square', 2);
Mpad = padarray([1 1; 1 1], [2 2], 0, 'both');
D1 = imdilate(A, Mpad);
D2 = imdilate(Mpad, A);

% Display and plot results
```

```
figure(4)
subplot(1,4,1), imshow(A), title('Image A');
subplot(1,4,2), imshow(Mpad), title('Mask M');
subplot(1,4,3), imshow(D1), title('M (+) A');
subplot(1,4,4), imshow(D2), title('A (+) M');

% Erosion
E1 = imerode(A, Mpad);
E2 = imerode(Mpad, A);

% Display and plot results
figure(5)
subplot(1,4,1), imshow(A), title('Image A');
subplot(1,4,2), imshow(Mpad), title('Mask M');
subplot(1,4,3), imshow(E1), title('M (-) A');
subplot(1,4,4), imshow(E2), title('A (-) M');
```





As shown in the figures above, the basic morphological operations dilation and erosion are not commutative.

----

c) (5) Based on the results of Question (b), discuss why *dilation* must be operated as $D = M \oplus X$ and *erosion* as $E = M ! X$.

----------------------------------- Q1.c: Analysis (5) ------------------------------------

To understand why dilation and erosion must be operated as $D = M \oplus X$ and $E = M ! X$ respectively, why must first understand what dilation and erosion are. Dilation is an operation that enlarges or thickens regions in an image. It involves placing a structuring element (a small pattern or kernel) at each pixel of the image and expanding the pixel values based on the presence of neighboring pixels. Dilation is denoted by $X \oplus Y$, where X is the input image and Y is the structuring element.

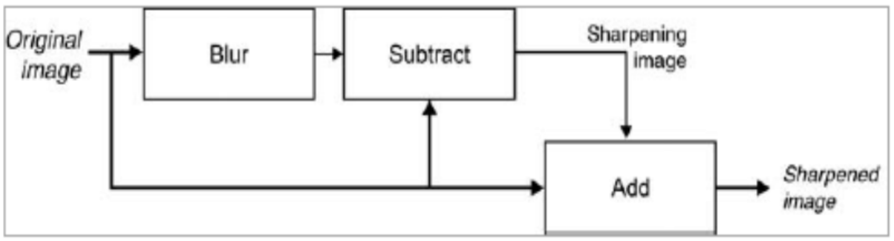
Erosion, on the other hand, is an operation that shrinks or thins regions in an image. It involves placing a structuring element at each pixel of the image and checking if all the pixels in the neighborhood defined by the structuring element are present. If they are, the pixel at the center remains unchanged; otherwise, it is eroded. Erosion is denoted by $X \ominus Y$, where X is the input image and Y is the structuring element.

The non-commutativity arises from the way these operations affect the image. If we are to flip X and Y around, Y becomes the input image and X becomes the structuring element. So now, the image will have certain regions expanded or shrunken based on the structure X instead of Y, so the result will be different than to have X the input image and Y as the structuring element.

----

2. (15) It is recognized that multiple filters and spatial operations may be composed in certain patterns in order to enhance image restoration in practice. Formally, these patterns can be described by mathematical models of *Image Frame Algebra* (IFA) as presented in the course.

A useful image enhancing technology known as the *image sharpening pattern* as shown in the following figure:



Its algebraic model is: SharpIm = (Im – Iblur) + Im (1)

In the above model, *Iblur* may be generated by the imfilter function, i.e.: `Inoi = imnoise(Im,'gaussian',0.01); Fblur = fspecial('average', 5); Iblur = imfilter(Inoi,Fblur).`

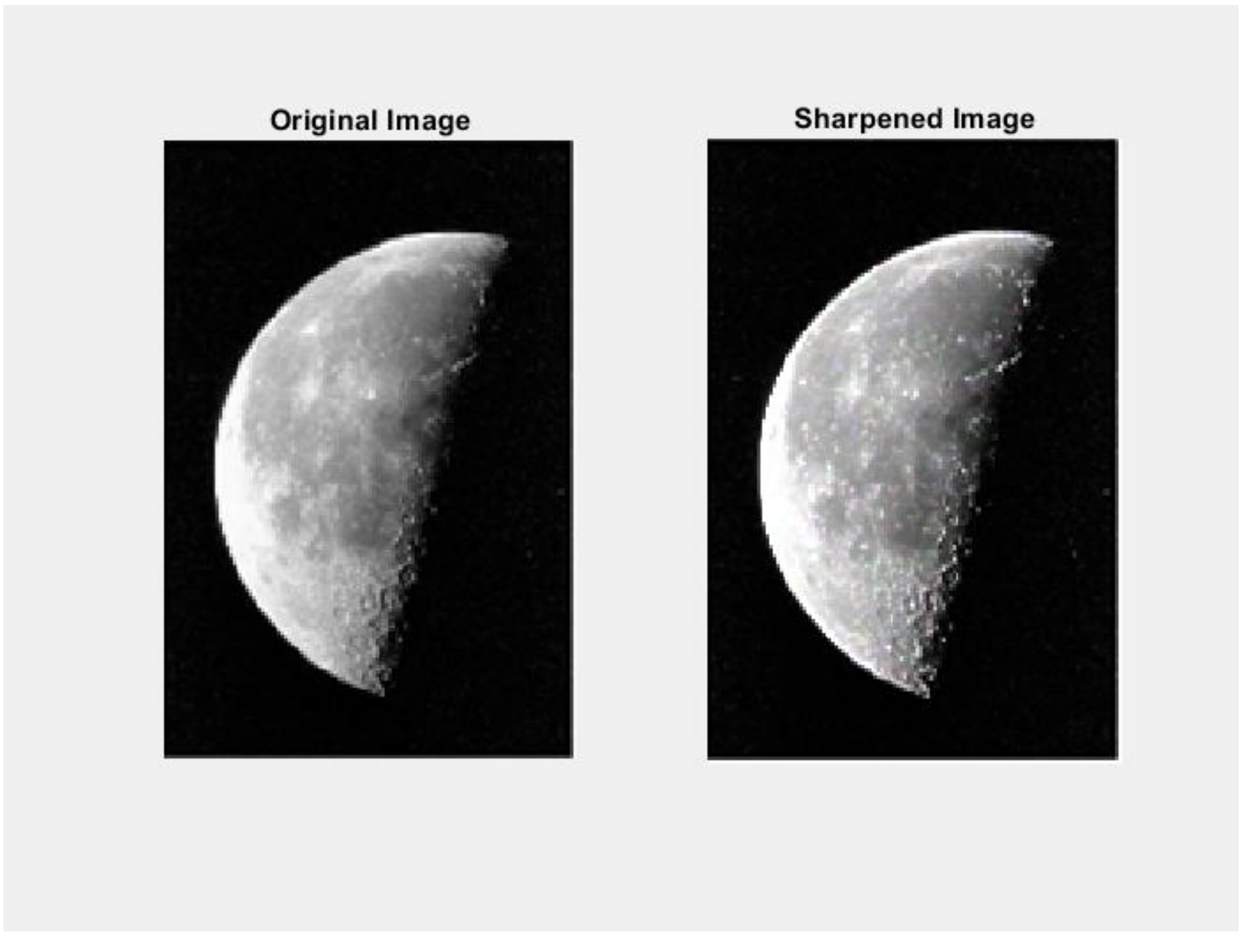
Given a sample image `'Moon.tif'` as provided in Lab3 Materials, implement the mathematical model of Ep. (1) in MATLAB and show your testing results.

---------------------------------- MATLAB code (10) ---------------------------------

```
%% Question 2

% Read Moon.tif image
Im = imread("Moon.tif");

% Add Gaussian noise to the image
Inoi = imnoise(Im,'gaussian',0.01);

% Apply a 5x5 average filter to blur the image
Fblur = fspecial('average', 5);
Iblur = imfilter(Inoi, Fblur);
```

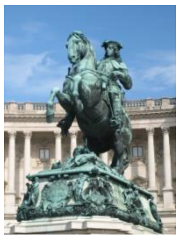
```
% Sharpen the image
SharpIm = (Im - Iblur) + Im;

% Display and plot results
figure(6)
subplot(1,2,1), imshow(Im), title('Original Image');
subplot(1,2,2), imshow(SharpIm), title('Sharpened Image');
-----------------------------------------------------------------------------------
```

--------------------------------- Plots of results (5) ---------------------------------



----

3. (20) Morphology provides a new perspective on image topology and geography. Test how the following morphological expressions are functioned in MATLAB for Image Contrast Enhancement (*ImEnhance*) on the given image *Statue.png*.



$$ImEnhance = X^g + \hat{H}^g - \underset{\smile}{H}^g$$
$$where$$

$$\hat{H}^g = M^b \hat{\otimes} X^g \triangleq X^g - (M^b \circ X^g) \quad \text{// Top-hat}$$
$$\underset{\smile}{H}^g = M^b \underset{\smile}{\otimes} X^g \triangleq (M^b \bullet X^g) - X^g \quad \text{// Buttom-hat}$$

a) (5) Prove $ImEnhance = X + [X - (M \circ X)] - [(M \bullet X) - X]$ based on the definitions of *top-hat* and *bottom-hat* as given above. Explain the physical meaning of the principle that you proved.

--------------------------- Your proof and analysis (5) ----------------------------

The definition of top-hat is $\hat{H}^g = M^b \hat{\otimes} X^g \triangleq X^g - (M^b \circ X^g)$ and the definition of bottom-hat is $\underset{\smile}{H}^g = M^b \underset{\smile}{\otimes} X^g \triangleq (M^b \bullet X^g) - X^g$. In the definition of ImEnhance = $X + [X - (M \circ X)] - [(M \bullet X) - X]$, the terms $[X - (M \circ X)]$ is equal to tophat and $[(M \bullet X) - X]$ is equal to bottom hat. I.e. $[X - (M \circ X)]$ = $X^g - (M^b \circ X^g)$ and $[(M \bullet X) - X]$ = $(M^b \bullet X^g) - X^g$. $X^g = X$. The physical meaning is that both definitions have the same terms.

----

b) (15) Question 3(a) has indicated there are two equivalent algorithms for implementing the morphological methodology for image contrast enhancement, i.e., either $X + \hat{H} - \underset{\smile}{H}$ or $X + [X - (M \circ X)] - [(M \bullet X) - X]$. Try to implement both algorithms in MATLAB and show your testing results. The following built-in functions of MATLAB will be applied: `imtophat`, `imbothat`, `imopen`, and `imclose`.

--------------------------- MATLAB code for Algorithm 1 (5) ---------------------------

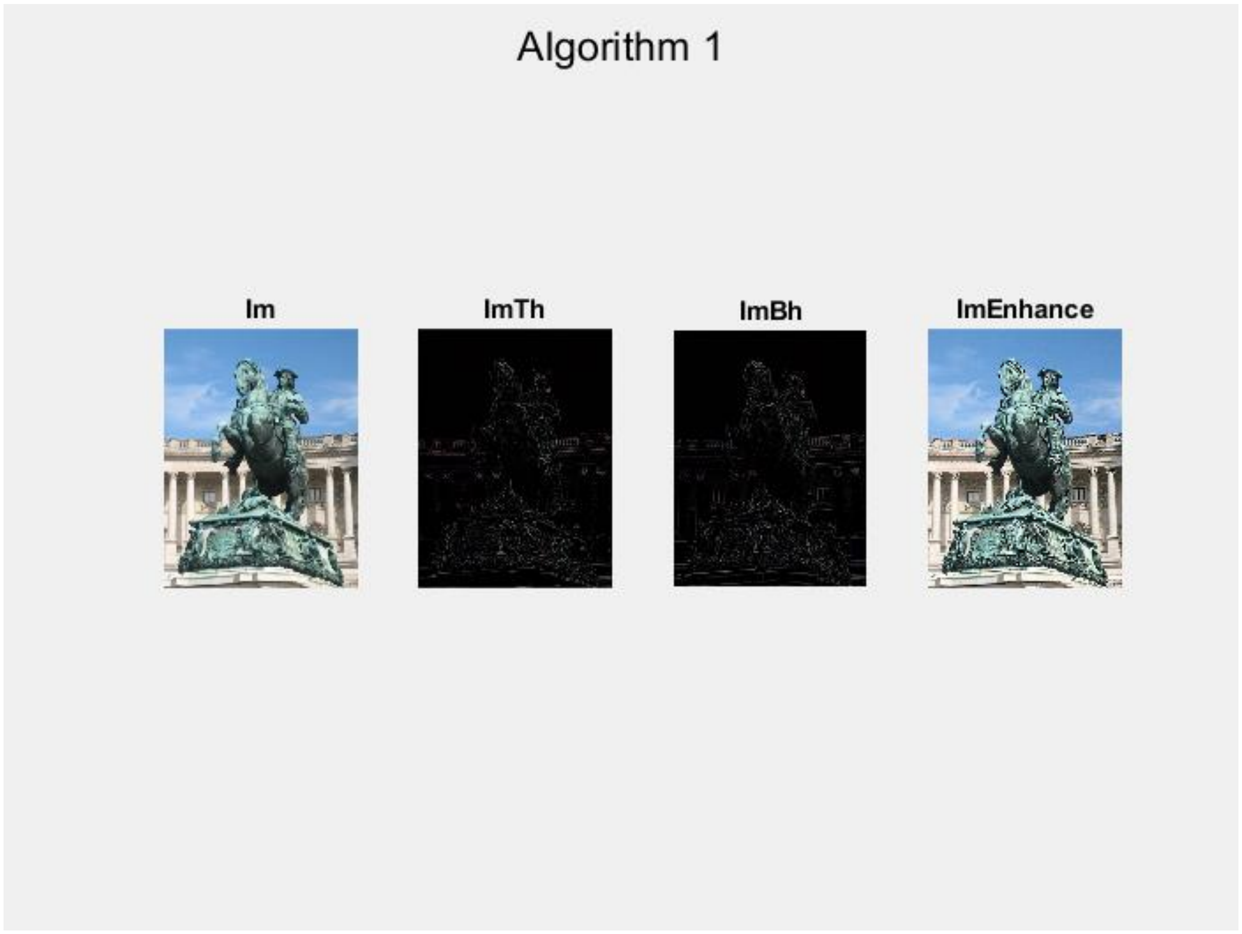
```
%% Question 3

% Read Statue.png image
Im = imread("Statue.png");

% Create a disk-shaped structuring element with radius 3
Mse = strel('disk', 3);

% Algorithm 1
ImTh = imtophat(Im, Mse);
ImBh = imbothat(Im, Mse);
ImEnhance = Im + ImTh - ImBh;

% Display and plot results
figure(7)
subplot(1,4,1), imshow(Im), title('Im');
subplot(1,4,2), imshow(ImTh), title('ImTh');
subplot(1,4,3), imshow(ImBh), title('ImBh');
subplot(1,4,4), imshow(ImEnhance), title('ImEnhance');
sgtitle('Algorithm 1');
```

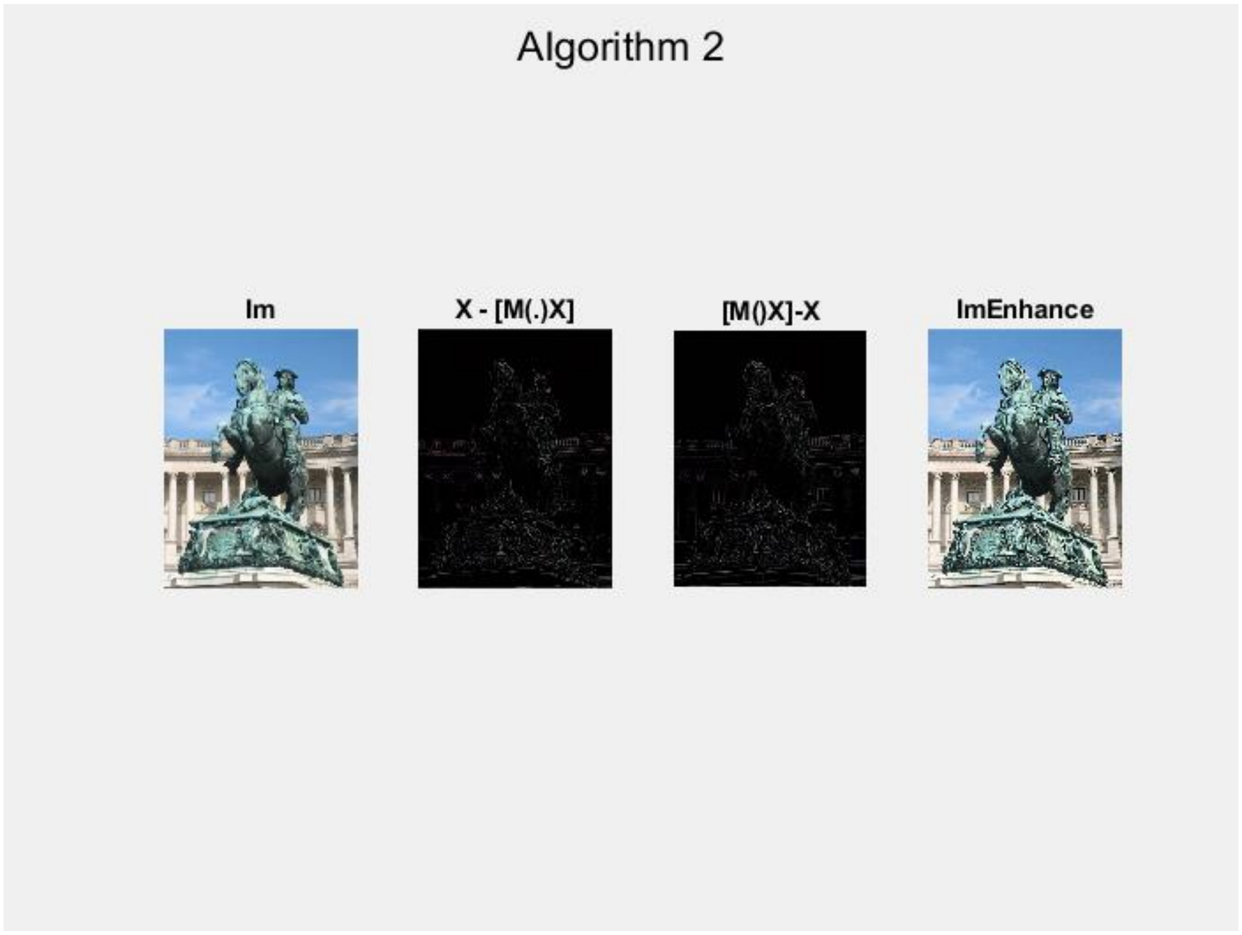
--------------------------- MATLAB code for Algorithm 2 (7) ---------------------------

```
% Algorithm 2
term_1 = Im - (imopen(Im, Mse));
term_2 = (imclose(Im, Mse)) - Im;
```

```
ImEnhance = Im + term_1 - term_2;

% Display and plot results
figure(8)
subplot(1,4,1), imshow(Im), title('Im');
subplot(1,4,2), imshow(term_1), title('X - [M(.)X]');
subplot(1,4,3), imshow(term_2), title('[M()X]-X');
subplot(1,4,4), imshow(ImEnhance), title('ImEnhance');
sgtitle('Algorithm 2');
```

---------------------------- Plots of testing results (3) ----------------------------





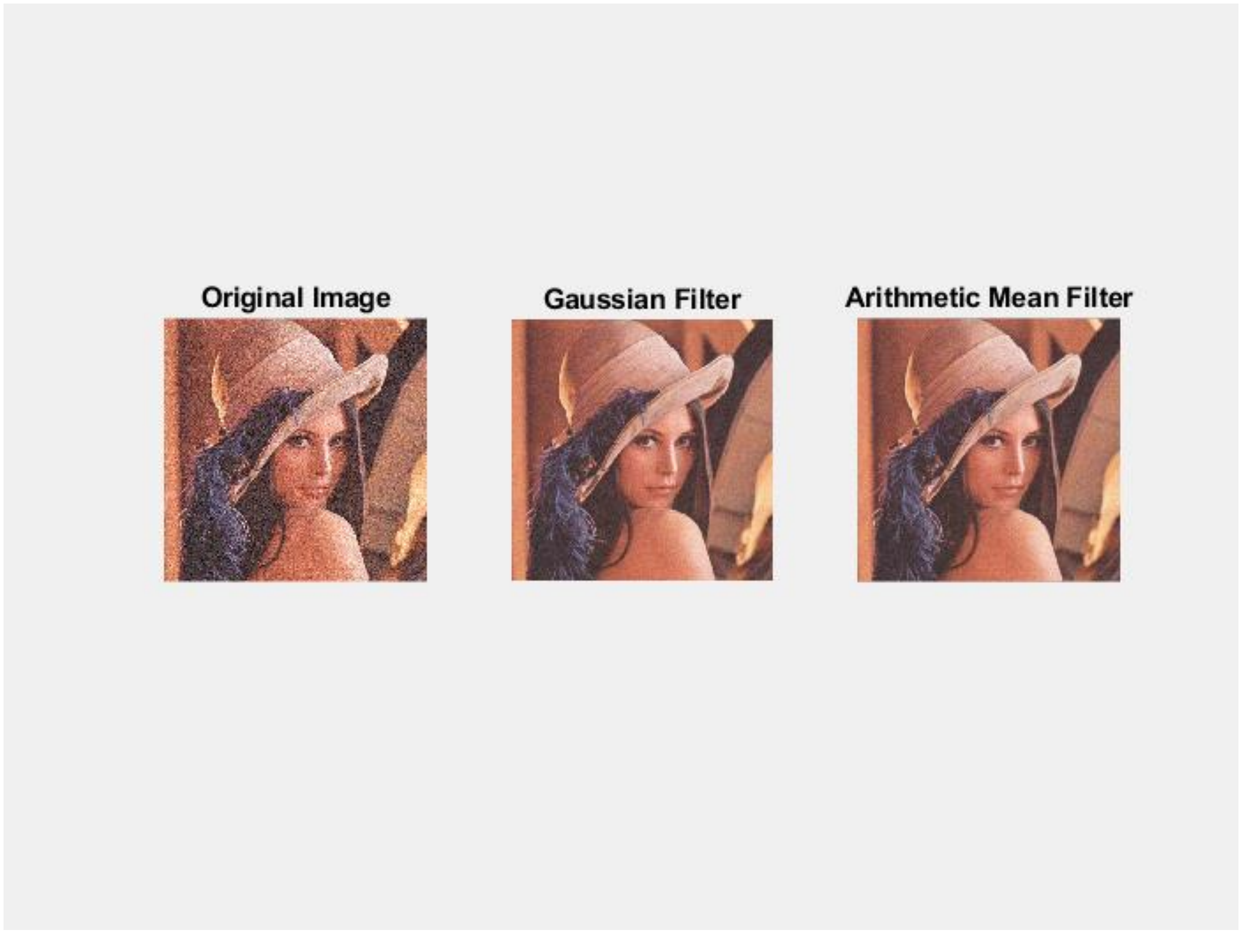
----

4. (10) Suppose a color image, In = LinnaGNoise.jpg, was affected by Gaussian noise in its R/G/B components as given in Lab3 materials. Try to recover (denoising) it by both 2D filtering technologies based on MATLAB functions `imgaussfilt(…)` and `imfilter(…)`. Plot and compare the original and filtered images.

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

```
%% Question 4

% Read LennaGNoise.jpg image
Im = imread("LennaGNoise.jpg");

% Apply Gaussian filter with standard deviation 0.8
Im1 = imgaussfilt(Im, 0.8);

% Apply 3x3 arithmetic mean filter
Filter1 = fspecial('average', [3 3]);
Im2 = imfilter(Im, Filter1);

% Display and plot results
figure(9)
subplot(1,3,1), imshow(Im), title('Original Image');
subplot(1,3,2), imshow(Im1), title('Gaussian Filter');
subplot(1,3,3), imshow(Im2), title('Arithmetic Mean Filter');
```

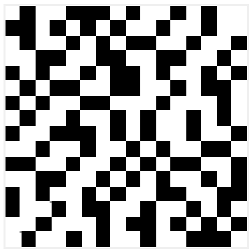
---------------------------- Plots of results (3) --------------------------



----

5. (35) Design your own 2D barcode

The 2D barcode is a popular technology of computer vision developed in 2000. The geographical and spatial operations of MATLAB can be applied to develop an arbitrary barcode system. In this given problem, design and implement a 16 x 16 2D barcode as follows

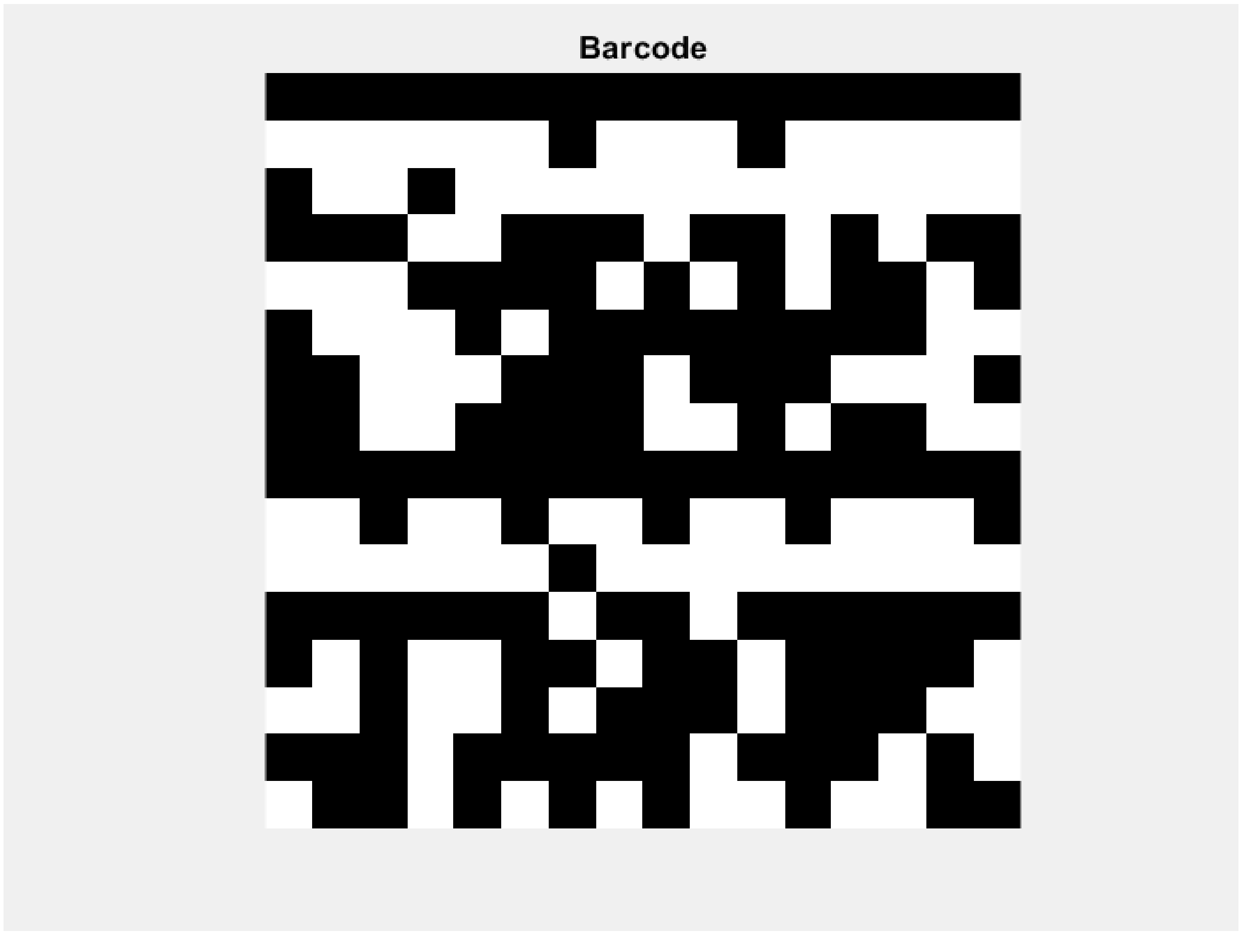


a) (20) In the barcode generation process, the 2D 16 x 16 *Barcode* will be structured as zeros (1, 16). Then an arbitrary *Str* = `'Hello World! This is my barcode.'` will be coded into *Barcode* using string conversion technologies: `StrB = reshape(dec2bin(Str, 8).'-'0',1,[]).` The final step will write *StrB* into the 16 x (8 + 8) segments of *Barcode* where each segment representing an ASCII letter as inputted in *Str*.

-------------------------- MATLAB code for barcode generation (15) ------------------------

```
%% Question 5

% Initialize a 16x16 barcode matrix
Barcode = zeros(16, 16);

% Arbitrary string for the barcode
Str = 'Hello World! This is my barcode.';
StrB = reshape(dec2bin(Str, 8).' - '0', 1, []);

% Populate the barcode matrix with the binary representation of the string
for i = 1:length(Barcode)
    for j = 1:length(Barcode)
        index = ((j-1)*16) + i;
        Barcode(i, j) = StrB(index);
    end
end

% Display and plot the generated barcode
figure(10),
imshow(Barcode, 'InitialMagnification', 'fit'), title("Barcode")
```

------------- Plots of your barcode generated (5) --------------



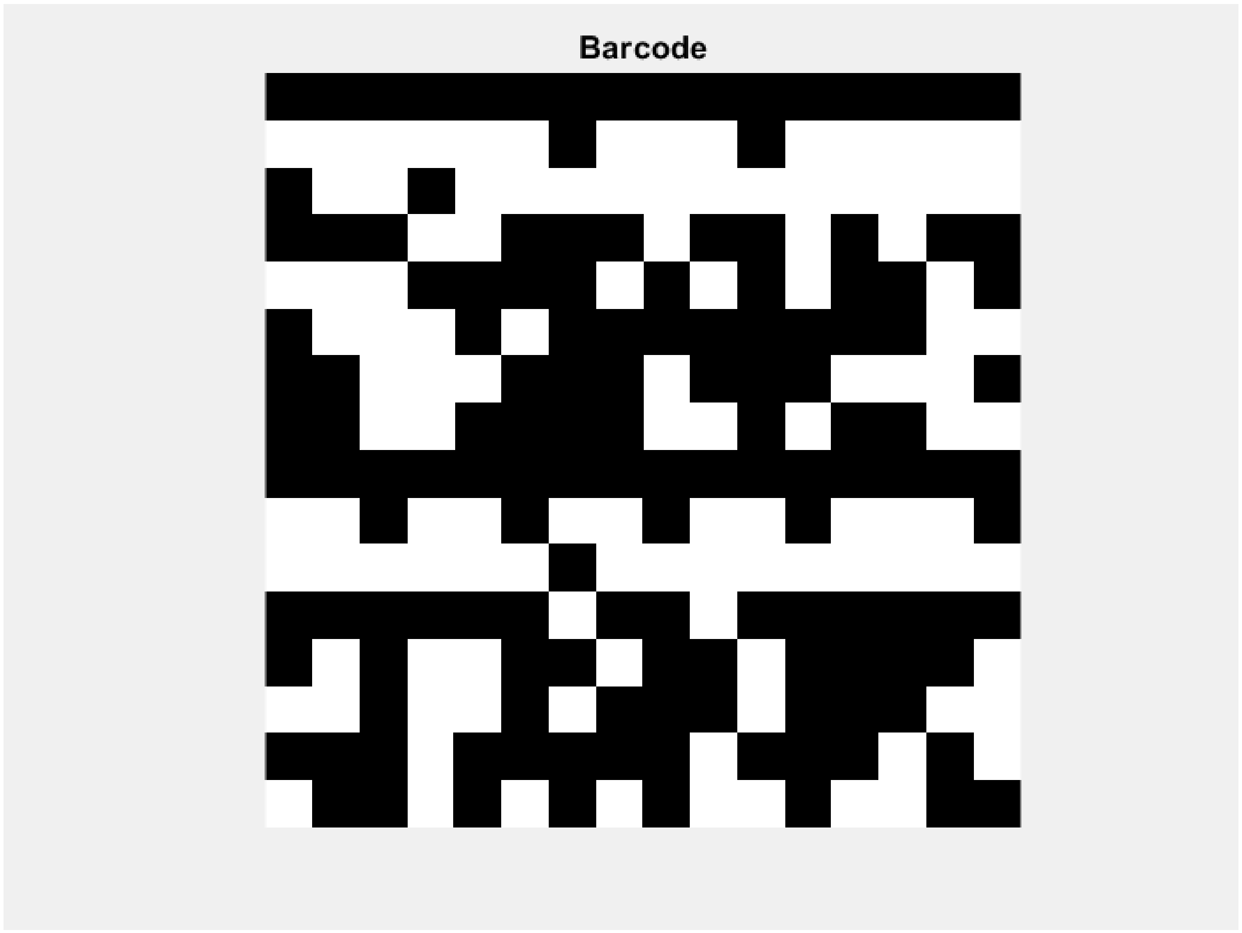
----

b) (15) In the barcode recognition process, the 16 x (8 + 8) binary segments of *Barcode* will be converted from 16 x 2 integers (uint8) to characteristics (char) in iteration. Then, the assembled string will be the output of your recognized barcode `BarCode = char(String)`.

---------------------------- MATLAB code for barcode recognition (10) ----------------------------

```
% Convert barcode segments to integers and then to characters
BinarySegments = reshape(Barcode, 8, []).';
IntValues = bin2dec(char(BinarySegments + '0'));
RecognizedStr = char(IntValues.');

% Display the recognized barcode information
disp('Recognized Barcode Information:');
disp(RecognizedStr);
```

------------- Plots of your barcode and the recognized information (5) --------------



Recognized Barcode Information:
Hello World! This is my barcode.
