江西理工大學信息工程学院

JIANGXI UNIVERSITY OF SCIENCE AND TECHNOLOGY SCHOOL OF INFORMATION ENGINEERING

Dr AJM @ 2021

drajm@yahoo.com

Contents

Practice ONE

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	1	Practical title	**How to install MATLAB? & How to use MATLAB? (Basic Programming)**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					

The purpose and requirement (fill in)

Purpose:

1. To install MATALB Tool.
2. Learn how to use and awareness about different options of MATALB.
3. Learn how to programming in command window and also script (.m) file.
4. Lean fundamentals of MATLAB Programming.

Requirement:

1. Each student must have resources(computer)
2. Every student have installed MATLAB2014a version above
3. Every student have lecture slide and file which I send.

Content:

1) Fundamental Operation
   a) Code compilation
   b) Different windows usage
   c) MATLAB Tools usage
   d) Variable
   e) Matrix
   f) Loop
   g) Conditional statements
2) Image reading
   a) RGB image
   b) Gray scale image
   c) Black and white
   d) Image visualization
3) Basic Image Operation
   a) Image conversion
   b) Image resizing
   c) How to find image size
   d) Some other basic operation

1. MATLAB VARIABLES

Matlab Code	Result

clc
clear all;
close all;
x = 3   % defining x and initializing it with a value
Y = sqrt(16) % defining Y and initializing it with an expression
sqrt(78)
x1 = 7 * 8;
y = x1 * 7.89	

Important note about example :	

2. MATLAB VARIABLES

Matlab Code	Result

clc
clear all;
close all;
x=3
clear x   % it will delete x, won't display anything
clear   % it will delete all variables in the workspace   % peacefully and unobtrusively Long Assignments Long assignments can be extended to another line by using an ellipses (...). For example,
initial_velocity = 0;
acceleration = 9.8;
time = 20;
final_velocity = initial_velocity + acceleration * time	

Important note about example :	

3. MATLAB – MATRIX

Matlab Code	Result

clc
clear all;
close all;
m = [1 2 3; 4 5 6; 7 8 9]
% Referencing the Elements of a Matrix
a = [1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];
% see the results of the follwoing
a(2,5)
v = a(:,4)
a(:,2:3)	

Important note about example :	

4. MATLAB – MATRIX: create a sub-matrix

Matlab Code	Result

clc
clear all;
close all;
a = [1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];
sa = a(2:3,2:4)	

Important note about example :	

5. MATLAB – MATRIX: Deleting a Row or a Column in a Matrix

```
clc
clear all;
close all;
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];
a( 4 , : ) = [] %For example, let us delete the fourth row of a
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];
a(: , 5)=[] %, let us delete the fifth column of a ?
```

6. MATLAB – MATRIX: % In this example, let us create a 3-by-3 matrix m, then we will copy the second and third rows of this matrix twice to create a 4-by-3 matrix. Create a script file with the following code

```
clc
clear all;
close all;
a = [ 1 2 3 ; 4 5 6; 7 8 9];
new_mat = a([2,3,2,3],:)
```

7. MATLAB – MATRIX: Special Arrays in MATLAB

```
clc
clear all;
close all;
zeros(5)  %. The zeros() function creates an array5*5 of all zeros
ones(4,3) %The ones() function creates an array 4*3 of all ones
eye(4)    % The eye() function creates an identity matrix 4*4
rand(3,5)%The rand() function creates an array3*5 of uniformly distributed random numbers on (0,1)
magic(4)%The magic() function creates a magic square array. of 4*4
```

1. MATLAB – MATRIX :Multidimensional Arrays

```
clc
clear all;
close all;
a = [7 9 5; 6 1 9; 4 3 2]
% The array a is a 3-by-3 array; we can add a third dimension to a, by providing the values like
a(:, :, 2)= [ 1 2 3; 4 5 6; 7 8 9]
% We can also create multidimensional arrays using the ones(), zeros() or the rand() functions. For
example,
b = rand(4,3,2)
```

8. MATLAB – MATRIX: Multidimensional Arrays concatenate the arrays

```
clc
clear all;
close all;
a = [9 8 7; 6 5 4; 3 2 1];
b = [1 2 3; 4 5 6; 7 8 9];
c = cat(3, a, b, [ 2 3 1; 4 7 8; 3 9 0])
```

9. FUNCTIONS OF ARRY

```
clc
clear all;
close all;
x = [7.1, 3.4, 7.2, 28/4, 3.6, 17, 9.4, 8.9];
length(x)  % length of x vector
y = rand(3, 4, 5, 2);
ndims(y)    % no of dimensions in array y
s = ['Zara', 'Nuha', 'Shamim', 'Riz', 'Shadab'];
numel(s)   % no of elements in s
```

10. FUNCTIONS OF ARRY

```
clc
clear all;
close all;
a = [1 2 3; 4 5 6; 7 8 9]  % the original array a
b = circshift(a,1)         %  circular shift first dimension values down by 1.
c = circshift(a,[1 -1])    % circular shift first dimension values % down by 1 % and second dimension
values to the left % by 1.
```

11. FUNCTIONS OF ARRY

```
clc
clear all;
close all;
% Sorting Arrays
v = [ 23 45 12 9 5 0 19 17]  % horizontal vector
sort(v)  % sorting v
m = [2 6 4; 5 3 9; 2 0 1]    % two dimensional array
sort(m, 1)                   % sorting m along the row
sort(m, 2)                   % sorting m along the column
```

12. Accessing Data in Cell Arrays

```
clc
clear all;
close all;
c = {'Red', 'Blue', 'Green', 'Yellow', 'White'; 1 2 3 4 5};
% See the  Result of the follwoing
c(1:2,1:2)
c{1, 2:4}
```

13. Creating Vectors (horizontal Vectors)

```
clc
clear all;
close all;
r = [7 8 9 10 11]
 %MATLAB will execute the above statement and return the following result ? r = 7    8    9   10   11
```

14. Another example,

```
t = [2, 3, 4, 5, 6];
res = r + t
```

15. Creating Vectors (Vertical Vectors)

```
clc
clear all;
close all;
c = [7;  8;  9;  10; 11]
```

16. Image Reading and show image

```
clc
clear all;
close all;
I=imread('pic.jpg'); % imread function is used to read an image . you can give full path
```

```
                        % of the image if image is not in the same place where you save program.
                          % Like this imread(慍:\Users\Muhammad\Desktop\picture.jpg?;
imshow(I); % this is used to visualize the image. This is use to show one image in program.
```

17. Image resizing

```
clc
clear all;
close all;
I = imread('pic1.jpg');
A=imresize(I,[500 500]);          %used to resize the image I to 256 256 pixels
subplot(1,2,1);imshow(I);
subplot(1,2,2);imshow(A);
```

18. RGB image, Size & visualizing multiple images in multiple windows

```
clc
clear all;
close all;
img = imread('pic.jpg');
[m n d]=size(img);  % it will show the  number of pixels in vertical and horizontal mean (column and
row) and also show the dimension. Which represent the number of cloro. In RGB case it will shoe 3
which mean R G B.
imgR = img(:,:,1);
imgG = img(:,:,2);
imgB = img(:,:,3);
figure;imshow(imgR,[]); % figure;imshow()  function are used to visualize multiple image in one
program. Then we use.
figure;imshow(imgG,[]);
figure;imshow(imgB,[]);
m
n
d
```

% Note: figure;imshow() function are used to visualize multiple image in one program in multiple window.

19. Image conversion

```
clc
clear all;
close all;
I = imread('pic.jpg');
figure;imshow(I,[]);title('RGB Colore Image');    % show image with title
X=rgb2gray(I); % this function is used to convert image from RGb to Gray
figure;imshow(X,[]);title('Gray Colore Image');  % show image with title
Y=im2bw(X);   % this function is used to convert image from gray or RGB to black and white
figure;imshow(Y,[]);title('Black and white Image');         % show image with title
```

20. Gray scale image & Black and white image visualizing multiple images in one window

```
clc
clear all;
close all;
I = imread('pic.jpg');
A=imread('pic1.jpg');
X=size(I)  % in the case of gray scale image it will show the dimension 1 .
subplot(1,2,1);imshow(I)  %this is used to show multiple image in one window(1,2,1)
                          % first 1 show that image will be in one or first window
                          % , 2 mean will be two image in one window, next 1 mean
                          %this image will be in first place.
subplot(1,2,2);imshow(A)
```

21. % for Loop in MATLAB

```
clc
clear all;
close all;
for x = 0:0.05:1  % X start from 0 end to 1 and increment by 0.05 if you want decrement you use -0.05
    x
 end
```

22. Nested For Loop in MATLAB

```
clc
clear all;
close all;
m=5;
n=4;
a = zeros(n,m);
for i = 1:n % auto increment by 1
    for j = 1:m % auto increment by 1
        a(i,j) = 1/(i+j);
    end
 end
a
```

23. while Loop in MATLAB

```
clc
clear all;
close all;
n = 1;
y = zeros(1,10);
while n <= 10
y(n) = 2*n/(n+1);
n = n+1;
end
y
```

n

24. if Condition Statement in MATLAB

```
clc
clear all;
close all;
attn=5;
grade=82;
if (attn>0.9)&(grade>60)
pass = 1
 end
```

25. else if Condition Statement in MATLAB

```
clc
clear all;
close all;
i=5;
j=10;
if i == j
    a(i,j) = 2
elseif i >= j
    a(i,j) = 1
else
    a(i,j) = 0
 end
```

26. switch Condition Statement in MATLAB

```
clc
clear all;
close all;
x = 2;
y = 3;
switch x
    case x==y
    disp('x and y are equal');
    case x>y
    disp('x is greater than y');
    otherwise
    disp('x is less than y');
end
```

27. MATLAB: Plot practice

```
clear;
clc;
close all
vis_ax = 'on';
ftsz=0.85;
```

```
fig_size = 800;
fig_0 = figure('color','w','position',[0, 0, fig_size*1.414,fig_size]);
set(fig_0,'renderer','Painters')
% main
ax_header = axes('position',[0,0,1,1],'visible','off');
```

% make title

```
ax_title  = axes('position',[0,0.88,0.5,0.1],'visible','off');
text(0.01,0.15,'Matlab Plot Cheatsheet','VerticalAlignment','bottom','FontSize',ftsz*60)
text(0.02,0.01,'https://github.com/Pjer-zhang/matlabPlotCheatsheet','VerticalAlignment','bottom','FontSize',ftsz*15,'FontName','consolas');
```

% plot colortable

```
ax_colortable = axes('position',[0.01,0.77,0.35,0.08],'visible',vis_ax);
text(1,0.98,'''color''','HorizontalAlignment','right','VerticalAlignment','top','FontSize',ftsz*12,'FontName','consolas','color','#A020F0')
text(0.01,0.98,'Line Color','VerticalAlignment','top','FontSize',ftsz*12,'color','k')

rectangle('Position',[0.01       ,0.37,0.08,0.23],'FaceColor','y')
rectangle('Position',[0.12+0.01  ,0.37,0.08,0.23],'FaceColor','m')
rectangle('Position',[0.24+0.01  ,0.37,0.08,0.23],'FaceColor','c')
rectangle('Position',[0.36+0.01  ,0.37,0.08,0.23],'FaceColor','r')
rectangle('Position',[0.48+0.01  ,0.37,0.08,0.23],'FaceColor','g')
rectangle('Position',[0.60+0.01  ,0.37,0.08,0.23],'FaceColor','b')
rectangle('Position',[0.72+0.01  ,0.37,0.08,0.23],'FaceColor','w')
rectangle('Position',[0.84+0.01  ,0.37,0.08,0.23],'FaceColor','k')
text(0   +0.04, 0.07,'''y''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.12+0.04, 0.07,'''m''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.24+0.04, 0.07,'''c''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.36+0.04, 0.07,'''r''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.48+0.04, 0.07,'''g''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.60+0.04, 0.07,'''b''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.72+0.04, 0.07,'''w''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.84+0.04, 0.07,'''k''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
xlim([0 1])
ylim([0 1])
xticks([])
yticks([])
box on
```

% marker

```
ax_marker = axes('position',[0.01,0.68,0.35,0.08],'visible',vis_ax);
text(1,0.98,'''marker''','HorizontalAlignment','right','VerticalAlignment','top','FontSize',ftsz*12,'FontName','consolas','color','#A020F0')
text(0.01,0.98,'Marker Style','VerticalAlignment','top','FontSize',ftsz*12,'color','k')
hold on
plot(0   +0.03,0.5, 'Marker','o','MarkerSize',8,'color','k','linewidth',1)
plot(0.07+0.03,0.5, 'Marker','+','MarkerSize',8,'color','k','linewidth',1)
plot(0.14+0.03,0.5, 'Marker','*','MarkerSize',8,'color','k','linewidth',1)
plot(0.21+0.03,0.5, 'Marker','.','MarkerSize',8,'color','k','linewidth',1)
plot(0.28+0.03,0.5, 'Marker','x','MarkerSize',8,'color','k','linewidth',1)
plot(0.35+0.03,0.5, 'Marker','s','MarkerSize',8,'color','k','linewidth',1)
plot(0.42+0.03,0.5, 'Marker','d','MarkerSize',8,'color','k','linewidth',1)
plot(0.49+0.03,0.5, 'Marker','^','MarkerSize',8,'color','k','linewidth',1)
plot(0.56+0.03,0.5, 'Marker','v','MarkerSize',8,'color','k','linewidth',1)
plot(0.63+0.03,0.5, 'Marker','>','MarkerSize',8,'color','k','linewidth',1)
plot(0.70+0.03,0.5, 'Marker','<','MarkerSize',8,'color','k','linewidth',1)
plot(0.77+0.03,0.5, 'Marker','p','MarkerSize',8,'color','k','linewidth',1)
plot(0.84+0.03,0.5, 'Marker','h','MarkerSize',8,'color','k','linewidth',1)
plot(0.91+0.03,0.5, 'Marker','none','MarkerSize',8,'color','k','linewidth',1)
text(0   +0.03, 0.07,'''o'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.07+0.03, 0.07,'''+'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.14+0.03, 0.07,'''*'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.21+0.03, 0.07,'''.'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.28+0.03, 0.07,'''x'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.35+0.03, 0.07,'''s'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.42+0.03, 0.07,'''d'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.49+0.03, 0.07,'''\^'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.56+0.03, 0.07,'''v'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.63+0.03, 0.07,'''>'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.70+0.03, 0.07,'''<'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.77+0.03, 0.07,'''p'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.84+0.03, 0.07,'''h'''   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.91+0.03, 0.07,'''none''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
xlim([0 1])
ylim([0 1])
xticks([])
yticks([])
box on
```

% marker size

```
ax_markersize = axes('position',[0.01,0.59,0.35,0.08],'visible',vis_ax);
text(1,0.98,'''markersize''','HorizontalAlignment','right','VerticalAlignment','top','FontSize',ftsz*12,'FontName','consolas','color','#A020F0')
text(0.01,0.98,'Marker Size','VerticalAlignment','top','FontSize',ftsz*12,'color','k')

hold on
plot(0   +0.06,0.5, 'Marker','o','MarkerSize',1,'color','k','linewidth',1)
plot(0.14+0.06,0.5, 'Marker','o','MarkerSize',2,'color','k','linewidth',1)
plot(0.28+0.06,0.5, 'Marker','o','MarkerSize',4,'color','k','linewidth',1)
plot(0.42+0.06,0.5, 'Marker','o','MarkerSize',8,'color','k','linewidth',1)
plot(0.56+0.06,0.5, 'Marker','o','MarkerSize',12,'color','k','linewidth',1)
plot(0.70+0.06,0.5, 'Marker','o','MarkerSize',16,'color','k','linewidth',1)
plot(0.84+0.06,0.5, 'Marker','o','MarkerSize',18,'color','k','linewidth',1)
text(0   +0.06, 0.07,'1'   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.14+0.06, 0.07,'2'   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.28+0.06, 0.07,'4'   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.42+0.06, 0.07,'8'   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.56+0.06, 0.07,'12'   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.70+0.06, 0.07,'16'   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.84+0.06, 0.07,'18'   ,'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')

xlim([0 1])
ylim([0 1])
xticks([])
yticks([])
box on
```

% line width

```
ax_linewidth = axes('position',[0.01,0.50,0.35,0.08],'visible',vis_ax);
text(1,0.98,'''linewidth''','HorizontalAlignment','right','VerticalAlignment','top','FontSize',ftsz*12,'FontName','consolas','color','#A020F0')
text(0.01,0.98,'Line Width','VerticalAlignment','top','FontSize',ftsz*12,'color','k')
hold on
plot([0.05     ,0.20     ],[0.36 0.55],'k','linewidth',1)
plot([0.05+0.25,0.20+0.25],[0.36 0.55],'k','linewidth',3)
plot([0.05+0.50,0.20+0.50],[0.36 0.55],'k','linewidth',5)
plot([0.05+0.75,0.20+0.75],[0.36 0.55],'k','linewidth',7)
text(     0.125, 0.07,'1','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.25+0.125, 0.07,'3','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.50+0.125, 0.07,'5','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
text(0.75+0.125, 0.07,'7','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','k')
xlim([0 1])
ylim([0 1])
xticks([])
yticks([])
box on
```

% line style

```
ax_linestyle = axes('position',[0.01,0.41,0.35,0.08],'visible',vis_ax);
text(1,0.98,'''linestyle''','HorizontalAlignment','right','VerticalAlignment','top','FontSize',ftsz*12,'FontName','consolas','color','#A020F0')
text(0.01,0.98,'Line Style','VerticalAlignment','top','FontSize',ftsz*12,'color','k')
hold on
plot([0.05     ,0.20     ],[0.36 0.55],'k','linewidth',2,'linestyle','-')
plot([0.05+0.25,0.20+0.25],[0.36 0.55],'k','linewidth',2,'linestyle','--')
plot([0.05+0.50,0.20+0.50],[0.36 0.55],'k','linewidth',2,'linestyle',':')
plot([0.05+0.75,0.20+0.75],[0.36 0.55],'k','linewidth',2,'linestyle','-.')
text(     0.125, 0.07,'''-''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.25+0.125, 0.07,'''--''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.50+0.125, 0.07,''':''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
text(0.75+0.125, 0.07,'''-.''','HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',ftsz*11,'FontName','consolas','color','#A020F0')
xlim([0 1])
ylim([0 1])
xticks([])
yticks([])
box on
```

% 2-D plot

```
data1d=1+sin(0.4*linspace(1,15,15));
data2d=peaks(20);

ax_2d_01 = axes('position',[0.01+0.086*0,0.28,0.077,0.09],'visible',vis_ax);
```

```
plot(data1d);  xticks([]);yticks([]);
text(0,1.01,'plot(y)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_02 = axes('position',[0.01+0.086*1,0.28,0.077,0.09],'visible',vis_ax);
area(data1d); xticks([]);yticks([]);
text(0,1.01,'area(y)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)


ax_2d_03 = axes('position',[0.01+0.086*2,0.28,0.077,0.09],'visible',vis_ax);
stem(data1d);
xticks([]);yticks([]);
text(0,1.01,'stem(y)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_04 = axes('position',[0.01+0.086*3,0.28,0.077,0.09],'visible',vis_ax);
stairs(data1d);
xticks([]);yticks([]);
text(0,1.01,'stairs(y)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_1 = axes('position',[0.01+0.086*0,0.15,0.077,0.09],'visible',vis_ax);
imagesc(data2d);  xticks([]);yticks([]);
text(0,1.01,'imagesc(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_2 = axes('position',[0.01+0.086*1,0.15,0.077,0.09],'visible',vis_ax);
contourf(data2d); xticks([]);yticks([]);
text(0,1.01,'contourf(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_3 = axes('position',[0.01+0.086*2,0.15,0.077,0.09],'visible',vis_ax);
pcolor(data2d);
xticks([]);yticks([]);
text(0,1.01,'pcolor(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_4 = axes('position',[0.01+0.086*3,0.15,0.077,0.09],'visible',vis_ax);
contour(data2d);
xticks([]);yticks([]);
text(0,1.01,'contour(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_5 = axes('position',[0.01+0.086*0,0.02,0.077,0.09],'visible',vis_ax);
surf(data2d);  xticks([]);yticks([]);
text(0,1.01,'surf(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_6 = axes('position',[0.01+0.086*1,0.02,0.077,0.09],'visible',vis_ax);
mesh(data2d); xticks([]);yticks([]);
text(0,1.01,'mesh(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_7 = axes('position',[0.01+0.086*2,0.02,0.077,0.09],'visible',vis_ax);
contour3(data2d);
xticks([]);yticks([]);
text(0,1.01,'contour3(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)

ax_2d_8 = axes('position',[0.01+0.086*3,0.02,0.077,0.09],'visible',vis_ax);
waterfall(data2d);
xticks([]);yticks([]);
text(0,1.01,'waterfall(Z)','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)
```

% axes position

```
ax_posi = axes('position',[0.48,0.5,0.2,0.2],'visible','on');
box on
plot(data1d)
axes(ax_header)

%text(0.38,0.82,'Add axes to frame','Units','normalized','VerticalAlignment','bottom',...
%    'HorizontalAlignment','left','FontSize',ftsz*14,'color',"k")

text(0.73,0.825,'ax=Axes("position", [left,bottom,width,height])','Units','normalized','VerticalAlignment','bottom',...
    'HorizontalAlignment','right','FontSize',ftsz*13,'color',"k","FontName",'consolas')

text(0.73,0.82,'Frame','Units','normalized','VerticalAlignment','top',...
    'HorizontalAlignment','right','FontSize',ftsz*30,'color',"#aaaaaa")
```

```
text(0.68,0.7,'Axes','Units','normalized','VerticalAlignment','top',...
    'HorizontalAlignment','right','FontSize',ftsz*30,'color',"#aaaaaa")
rectangle('Position',[0.38,0.4,0.35,0.42],'FaceColor','none')
annotation('doublearrow','Position',[0.38,0.57,0.1,0])
annotation('doublearrow','Position',[0.6,0.4,0.0,0.1])
annotation('doublearrow','Position',[0.48,0.7,0.2,0])
annotation('doublearrow','Position',[0.68,0.5,0.0,0.2])
```

%[left bottom width height]

```
text(0.42,0.57,'left','Units','normalized','VerticalAlignment','bottom',...
    'HorizontalAlignment','center','FontSize',ftsz*12,'color',"k")
text(0.602,0.46,'bottom','Units','normalized','VerticalAlignment','top',...
    'HorizontalAlignment','left','FontSize',ftsz*12,'color',"k")
text(0.6,0.7,'width','Units','normalized','VerticalAlignment','bottom',...
    'HorizontalAlignment','right','FontSize',ftsz*12,'color',"k")
text(0.681,0.6,'height','Units','normalized','VerticalAlignment','top',...
    'HorizontalAlignment','left','FontSize',ftsz*12,'color',"k")

xticks([])
yticks([])
xlim([0,1])
ylim([0,1])


text(0,1.01,'shading(ax,"flat")','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)
```

% renderer

```
ax_rder1 = axes('position',[0.38,0.17,0.13,0.14],'visible',vis_ax);
h1=pcolor(data2d);
h1.EdgeColor='none';
shading(ax_rder1,'flat')
xticks([]);yticks([]);
text(0,1.01,'shading(ax,"flat")','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)
text(0,1.21,'h=pcolor(Z);','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)
text(0,1.11,'h.EdgeColor="none";','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)


text(0,1.3,'Renderer','Units','normalized','VerticalAlignment','bottom','FontSize',ftsz*15)

ax_rder2 = axes('position',[0.38,0.01,0.13,0.14],'visible',vis_ax);
h2=pcolor(data2d);
h2.EdgeColor='none';
shading(ax_rder2,'interp')
xticks([]);yticks([]);
text(0,1.01,'shading(ax,"interp")','Units','normalized','VerticalAlignment','bottom',...
    'FontName','consolas','FontSize',ftsz*10)
```

% text position

```
ax_txt_posi = axes('position',[0.52,0.015,0.21,0.36],'visible',vis_ax);
text(1,0.98,'text(x,y,str)','HorizontalAlignment','right','VerticalAlignment','top','FontSize',ftsz*12,'FontName','consolas','color','k')
text(0.01,0.98,'Text alignment','VerticalAlignment','top','FontSize',ftsz*12,'color','k')
text(0.72,0.9,'"VerticalAlignment"','FontName','consolas','HorizontalAlignment','center','FontSize',ftsz*10,'color','#A020F0')
text(0.28,0.85,'"HorizontalAlignment"','FontName','consolas','HorizontalAlignment','center','FontSize',ftsz*10,'color','#A020F0')
hold on
plot(0.28,0.15+2*0.25,'k+','markersize',12)
plot(0.72,0.15+2*0.25,'k+','markersize',12)
plot(0.28,0.15+1*0.25,'k+','markersize',12)
plot(0.72,0.15+1*0.25,'k+','markersize',12)
plot(0.28,0.15+0*0.25,'k+','markersize',12)
plot(0.72,0.15+0*0.25,'k+','markersize',12)

text(0.28,0.15+2*0.25,'"left"','HorizontalAlignment','left','FontSize',ftsz*13,'color','#A020F0','fontname','consolas')
text(0.72,0.15+2*0.25,'"middle"','VerticalAlignment','middle','FontSize',ftsz*13,'color','#A020F0','fontname','consolas')
text(0.28,0.15+1*0.25,'"center"','HorizontalAlignment','center','FontSize',ftsz*13,'color','#A020F0','fontname','consolas')
text(0.72,0.15+1*0.25,'"top"','VerticalAlignment','top','FontSize',ftsz*13,'color','#A020F0','fontname','consolas')
text(0.28,0.15+0*0.25,'"right"','HorizontalAlignment','right','FontSize',ftsz*13,'color','#A020F0','fontname','consolas')
text(0.72,0.15+0*0.25,'"bottom"','VerticalAlignment','bottom','FontSize',ftsz*13,'color','#A020F0','fontname','consolas')

plot([0.5 0.5],[0.1,0.79],'k-')
box on
xticks([])
yticks([])
```

```
xlim([0,1])
ylim([0,1])
```

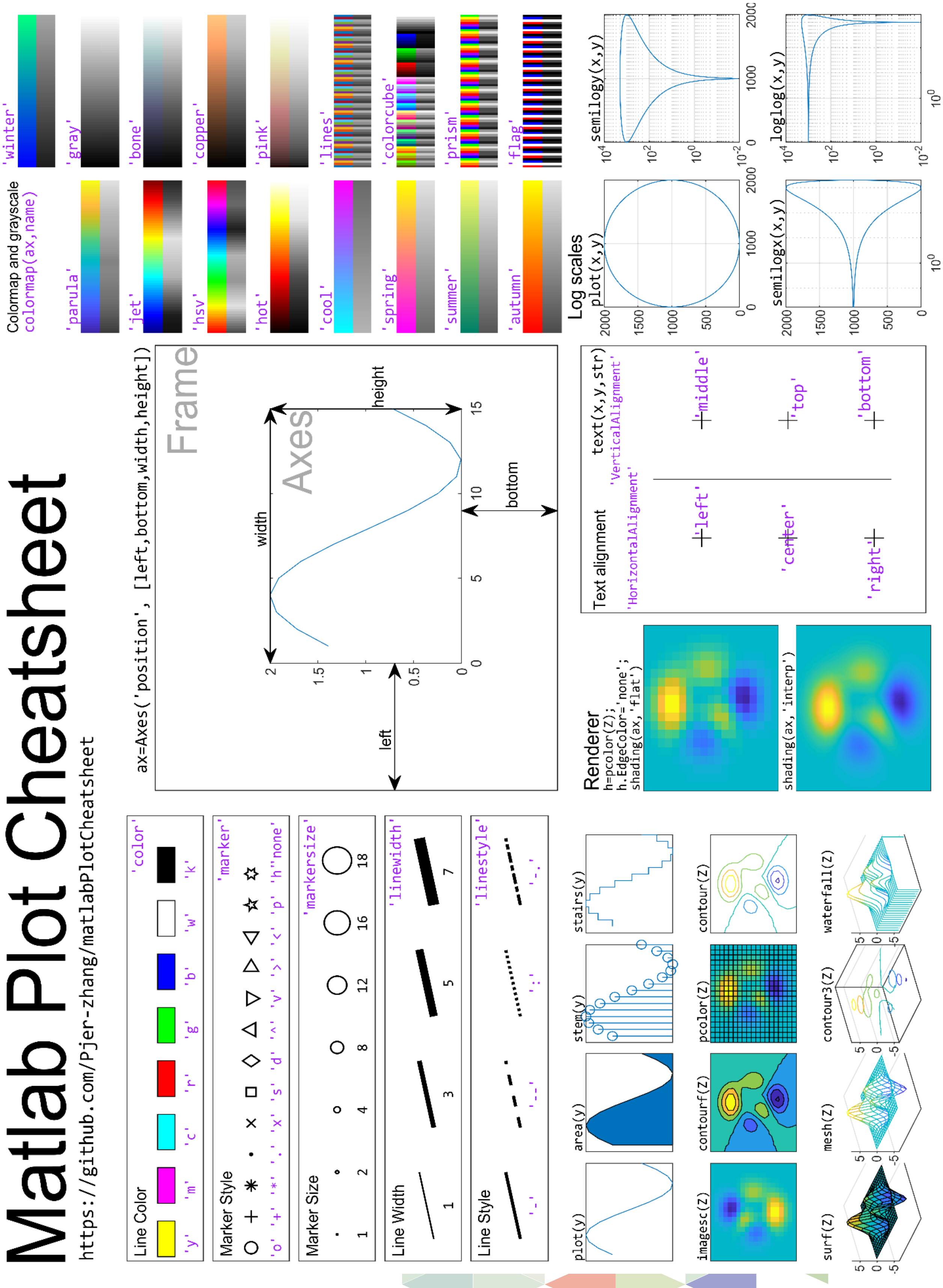
% the colormap

```
axes(ax_header)
% colormap title
cm_label = {'parula','jet','hsv','hot','cool','spring','summer','autumn',...
    'winter','gray','bone','copper','pink','lines','colorcube','prism','flag'};
ax_null = axes('position',[0.74,1.01-1*0.066,     0.12,0.02],'visible','off');
text(0,0.78,"Colormap and grayscale",'Units','normalized','VerticalAlignment','bottom',...
        'FontSize',ftsz*11,'color','k')

text(0,-0.03,"colormap(ax,name)",'Units','normalized','VerticalAlignment','bottom',...
        'FontName','consolas','FontSize',ftsz*12,'color','#A020F0')


for num=1:8
    cm_this=colormap(ax_null,cm_label{num});
    img_tmp = zeros(1,size(cm_this,1),size(cm_this,2));
    img_tmp(1,:,:)=cm_this;
    img_cm = repmat(img_tmp,32,1,1);
    gray_cm = rgb2gray(img_cm);

    axes('position',[0.74,1.01-(num+1)*0.066,     0.12,0.02],'visible',vis_ax);
    imshow(img_cm)
    axis normal
    axes('position',[0.74,1.01-(num+1)*0.066-0.02,0.12,0.02],'visible',vis_ax);
    imshow(gray_cm)
    axis normal


    text(0,2.01,["",cm_label{num},""],'Units','normalized','VerticalAlignment','bottom',...
        'FontName','consolas','FontSize',ftsz*12,'color','#A020F0')

    %set(gca,'position',[0 0 1 1])
end


for num=9:length(cm_label)
    cm_this=colormap(ax_null,cm_label{num});
    img_tmp = zeros(1,size(cm_this,1),size(cm_this,2));
    img_tmp(1,:,:)=cm_this;
    img_cm = repmat(img_tmp,32,1,1);
    gray_cm = rgb2gray(img_cm);

    axes('position',[0.87,1.01-(num-8)*0.066,0.12,0.02],'visible',vis_ax);
    imshow(img_cm)
    axis normal
    axes('position',[0.87,1.01-(num-8)*0.066-0.02,0.12,0.02],'visible',vis_ax);
    imshow(gray_cm)
    axis normal


    text(0,2.01,["",cm_label{num},""],'Units','normalized','VerticalAlignment','bottom',...
        'FontName','consolas','FontSize',ftsz*12,'color','#A020F0')

    %set(gca,'position',[0 0 1 1])
end
```

% the log scale

```
xx = 0.01+ 1000*(1+cos(2*pi*linspace(0,1,800)));
yy = 0.01+ 1000*(1+sin(2*pi*linspace(0,1,800)));

ax_log1 = axes('position',[0.76,0.21,0.10,0.1414],'visible',vis_ax);
plot(xx,yy)
text(0,1.01,"plot(x,y)",'Units','normalized','VerticalAlignment','bottom',...
        'FontName','consolas','FontSize',ftsz*12,'color','k')
grid on

text(-0.1,1.13,"Log scales",'Units','normalized','VerticalAlignment','bottom',...
        'FontSize',ftsz*14,'color','k')

ax_log2 = axes('position',[0.76,0.02,0.10,0.1414],'visible',vis_ax);
semilogx(xx,yy)
text(0,1.01,"semilogx(x,y)",'Units','normalized','VerticalAlignment','bottom',...
        'FontName','consolas','FontSize',ftsz*12,'color','k')
grid on
```

```
ax_log3 = axes('position',[0.89,0.21,0.10,0.1414],'visible',vis_ax);
semilogy(xx,yy)
text(0,1.01,'semilogy(x,y)','Units','normalized','VerticalAlignment','bottom',...
        'FontName','consolas','FontSize',ftsz*12,'color','k')
grid on

ax_log4 = axes('position',[0.89,0.02,0.10,0.1414],'visible',vis_ax);
loglog(xx,yy)
text(0,1.01,'loglog(x,y)','Units','normalized','VerticalAlignment','bottom',...
        'FontName','consolas','FontSize',ftsz*12,'color','k')
grid on

%orient(fig_0,'landscape')
%print('v0.pdf','-dpdf','-fillpage')
print('cheatsheet.png','-dpng','-r500')
```


Matlab Plot Cheatsheet
https://github.com/Pjer-zhang/matlabPlotCheatsheet
Line Color
'color'
'y' 'm' 'c' 'r' 'g' 'b' 'w' 'k'
Marker Style
'marker'
'o' '+' '*' '.' 'x' 's' 'd' '^' 'v' '>' '<' 'p' 'h' 'none'
Marker Size
'markersize'
1 2 4 8 12 16 18
Line Width
'linewidth'
1 3 5 7
Line Style
'linestyle'
'-' '--' ':' '-.'
ax=Axes('position', [left,bottom,width,height])
Frame
Axes
width
height
left
bottom
2 1.5 1 0.5 0
0 5 10 15
Colormap and grayscale
colormap(ax,name)
'parula' 'jet' 'hsv' 'hot' 'cool' 'spring' 'summer' 'autumn'
'winter' 'gray' 'bone' 'copper' 'pink' 'lines' 'colorcube' 'prism' 'flag'
Log scales
plot(x,y)
semilogy(x,y)
semilogx(x,y)
loglog(x,y)
2000 1500 1000 500 0
0 1000 2000
10^4 10^2 10^0 10^{-2}
10^0
Text alignment
'HorizontalAlignment'
text(x,y,str)
'VerticalAlignment'
'left' 'center' 'right'
'middle' 'top' 'bottom'
Renderer
h=pcolor(Z);
h.EdgeColor='none';
shading(ax,'flat')
shading(ax,'interp')
plot(y) area(y) stem(y) stairs(y)
imagesc(Z) contourf(Z) pcolor(Z) contour(Z)
surf(Z) mesh(Z) contour3(Z) waterfall(Z)
5 0 -5

Practice Two

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	2	Practical title	**Image transformation**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					

The purpose and requirement (fill in)

Purpose:

- Basic idea behind the practice is to learn practically about the image transformation because it's a common step mostly in every application (using image processing technique).
- Learn how to deal with the pixels, performing different operation.
- Understand how we can change intensity of an image.
- To get knowledge about the detail in an image (different slice contain).

Requirement:

- Each student must have resources(computer)
- Every student have installed MATLAB2014a version.

Every student have lecture slide and file which I send.

Content:

Image transformation.

a) Scaling
b) Thresholding
c) Log Transformation
d) Power low transformation
e) Contrast starching
f) Piece wise transformation
g) image slicing

1. Image Scaling (??)

```
clc;
clear all;
close all;
r=imread('pic1.jpg');
r=rgb2gray(r);
a=2;
[m n]=size(r);
for x=1:m
   for y=1:n
      s(x,y)=a*r(x,y);
   end
end
figure;imshow(r);
figure;imshow(s);
```

2. Image Threshold

```
clc;
clear all;
close all;
r=imread('pic1.jpg');

r=rgb2gray(r);
t=100;
[m n]=size(r);
for x=1:m
   for y=1:n
        if r(x,y)>t;
             s(x,y)=1;
        else
             s(x,y)=0;
       end
   end
end
figure;imshow(r);
figure;imshow(s);
```

3. Image Log Transformations

```
clc;
clear all;
close all;
r=imread('pic1.jpg');
r=imresize(r,[256 256]);
c=2;
[m n]=size(r);
for x=1:m
```

```
  for y=1:n
    h=double(r(x,y));
    s(x,y)=c.*log10(1+h);
  end
end
figure;imshow(s);
```

4. Image Power?Law (Gamma) Transformations

```
clc;
clear all;
close all;
r=imread('pic1.jpg');
G=rgb2gray(r);
G=im2double(G);
[m n]=size(G);
for x=1: m
  for y=1: n
    S(x,y)=G(x,y)^5;
  end
end
figure;imshow(S);
```

5. Another Contrast Stretching Function

```
clc;
clear all;
close all;
I=imread('pic1.jpg');
G=rgb2gray(I);
I = im2double(G);
m=0.75;
E=0.55;
g = 1./(1+(m./(I+eps)).^E);
figure,imshow(I),title('Original Image');
figure,imshow(g),title('Contrast stretched Image');
```

6. Piece wise Linear Transformations

```
clc;
clear all;
close all;
I=imread('pic1.jpg');
G=rgb2gray(I);
H = G;
[m n]=size(G);
T1= 100;
T2= 15;
for x=1:m
```

```
    for y = 1:n
        if G(x,y)< T1 && G(x,y)> T2
            H(x,y)=G(x,y)+20;
         else
             H(x,y)=G(x,y);

        end
    end
end
subplot(3,2,1:2);imhist(G)
subplot(3,2,3:4);imhist(H)
subplot(3,2,5);imshow(G)
subplot(3,2,6);imshow(H)
```

7. Image slicing

```
clc;
clear all;
close all;
I=imread('pic1.jpg');
im=rgb2gray(I);
bit1 = bitget(im, 1);
bit2=bitget(im,2);
bit3=bitget(im,3);
bit4=bitget(im,4);
bit5=bitget(im,5);
bit6=bitget(im,6);
bit7=bitget(im,7);
bit8=bitget(im,8);
figure,imshow(bit1, [])
figure,imshow(bit2, [])
figure,imshow(bit3, [])
figure,imshow(bit4, [])
figure,imshow(bit5, [])
figure,imshow(bit6, [])
figure,imshow(bit7, [])
figure,imshow(bit8, [])
```

Practice Three

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	3	Practical title	**Image Enhancement.**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					

The purpose and requirement (fill in)

Purpose:

- Basic idea behind the practice is to learn practically about the image transformation because it's a common step mostly in every application (using image processing technique).
- Learn how to deal with the pixels, performing different operation.
- Understand how we can change intensity of an image.
- To get knowledge about the detail in an image (different slice contain).

Requirement:

- Each student must have resources(computer)
- Every student have installed MATLAB2014a version.
- Every student have lecture slide and file which I send.

Content:

Image Enhancement.

- Histogram generation
- Histogram equalization
- Local Histogram Processing
- Mathematical/Logical Operations on Images
- Filtering (all include in lecture)
- Mean(Averaging) Filter
- Median filtering
- Second order derivative
- Laplace operator
- High Boost Filtering
- Gradient Operators
- Sobal Filter
- Gaussian Filter

1. Histogram generation

```
clc;
clear all;
close all;
I=imread('pic1.jpg');
G=rgb2gray(I);
subplot(2,2,1:2);imhist(G)
subplot(3,2,5);imshow(G)
```

2. Histogram equalization

```
clc;
clear all;
close all;
I=imread('pic1.jpg');
G=rgb2gray(I);
H=histeq(G);
subplot(3,2,1:2);imhist(G)
subplot(3,2,3:4);imhist(H)
subplot(3,2,5);imshow(G)
subplot(3,2,6);imshow(H)
```

3. Local Histogram Processing

```
clc;
clear all;
close all;
I=imread('pic1.jpg');
I=rgb2gray(I);
f=double(I);
[m n]=size(f);
f1 = f;
f2 = zeros(m,n);
f3 = zeros(m,n);
M=mean2(f);
D=std2(f);
k=[0.4 0.02 0.4];
E=4.0;
for i=2:m-1
for j=2:n-1
con=0; s=0;
for i1=i-1:i+1
for j1=j-1:j+1
con=con+1;
```

```
s(con)=f(i1,j1);
end
end
Mloc=mean(s);
f2(i,j)=mean(s);
Dloc = std(s);
f3(i,j)=std(s);
if (Mloc<=k(1)*M) && (Dloc>=k(2)*D) && (Dloc<=k(3)*D)
f1(i,j)=E*f(i,j);
else
f1(i,j)=f(i,j);
end
end
end
figure,imshow(I),title('Original Image');
figure,imshow(uint8(f2)),title('Image formed from local means');
figure,imshow(uint8(f3)),title('Image formed from local standard deviation');
figure,imshow(uint8(f1)),title('Image formed from all multiplication constants'),xlabel('Enhanced Image');
```

4. Add Mathematical Operations on Images

```
clc;
clear all;
close all;
I=imread('pic1.jpg')
I=rgb2gray(I);
J = imnoise(I,'salt & pepper',0.02);
figure;imshow(J);
K = filter2(fspecial('average',8),J)/255;
figure;imshow(K);
```

5. Subtract Mathematical Operations on Images

```
clc;
clear all;
close all;
I=imread('pic2.jpg');
I=imresize(I,[256 256]);
I=rgb2gray(I);
g=imread('pic1.jpg');
g=imresize(g,[256 256]);
g=rgb2gray(g);
F=imsubtract(I,g);
imshow(F)
```

6. Multi(*)Mathematical Operations on Images

```
clc;
clear all;
close all;
I=imread('pic2.jpg');
I=imresize(I,[256 256]);
I=rgb2gray(I);
g=imread('pic1.jpg');
g=imresize(g,[256 256]);
g=rgb2gray(g);
F=g.*I;
imshow(F);
```

7. AND Logical Operations on Images

```
clc;
clear all;
close all;
I=imread('pic2.jpg');
I=imresize(I,[256 256]);
I=rgb2gray(I);
g=imread('pic1.jpg');
g=imresize(g,[256 256]);
g=rgb2gray(g);
C=bitand(I, g);
imshow(C);
```

8. OR Logical Operations on Images

```
clc;
clear all;
close all;
I=imread('pic2.jpg');
I=imresize(I,[256 256]);
I=rgb2gray(I);
g=imread('pic1.jpg');
g=imresize(g,[256 256]);
g=rgb2gray(g);
C=bitor(I, g);
imshow(C);
```

9. Mean(Averaging) Filter

```
clc;
```

```
clear all;
close all;
i=imread('pic1.jpg');
i=im2double(i);
g=rgb2gray(i);
g = imnoise(g,'salt & pepper',0.08);
b=g;
s=size(g);
for x=2:s(1)-1
    for y=2:s(2)-1
        b(x,y)=(g(x+1,y)+g(x-1,y)+g(x+1,y+1)+g(x,y+1)+g(x,y-1)+g(x+1,y-1)+g(x-1,y-1)+g(x,y)+g(x-1,y+1))/9;

end
end
subplot(1,3,1); imshow(i);
subplot(1,3,2); imshow(g);
subplot(1,3,3); imshow(b);
```

10. Median filtering

```
clc;
clear all;
close all;
I=imread('11.jpg');
I=rgb2gray(I);
[r c]=size(I);
I = imnoise(I,'salt & pepper',0.02);
for x=2: r-1
    for y=2: c-1
        w=I(x-1:x+1,y-1:y+1);
        g=sort(w);
        f(x,y)=median(median(g));

    end
end
imshow(I,[]);
figure;imshow(f,[]);
```

11. Second order derivative

```
clc;
close all;
clear all;
```

```
I=imread('pic1.jpg');
I=rgb2gray(I);
I=imresize(I,[256 256]);
I=im2double(I);
[r c]=size(I);
for x=2: r-1
   for y=2: c-1
       G(x,y)=eps((I(x-1,y)+I(x+1,y))-2.*I(x,y));
        M(x,y)=G(x,y)+I(x,y);
   end
end
figure;imshow(I,[]);title('Orignal Image');
figure;imshow(M,[]);title('Sharp Image');
figure;imshow(G,[]);title('After Derivatie');
```

12. % Laplacian program with respect to +ve and -ve

```
clc;
close all;
clear all;
I=imread('11.jpg');
I=rgb2gray(I);
I=imresize(I,[256 256]);
[r c]=size(I);
LP=[-1 -1 -1;
   -1 8 -1;
   -1 -1 -1];       % Laplacian with repect to +ve window
LN=[1 1 1;
   1 -8 1;
   1 1 1];                    % Laplacian with repect to -ve window
for x=2: r-1
   for y=2: c-1
       w=I(x-1:x+1,y-1:y+1);
       gi=double(w)+double(LP);
       gp(x,y)=gi(2,2);       % Laplacian with repect to +ve
       g=imsubtract(double(w),double(LN));
       gn(x,y)=g(2,2);        % Laplacian with repect to +ve

   end
end
figure;imshow(I,[]);title('orignal image');
figure;imshow(gp,[]);title('+ve Laplacian Image');
figure;imshow(gn,[]);title('-ve Laplacian Image');
```

13. %%High Boost Filtering%%

```
I=imread('pic1.jpg');
I=rgb2gray(I);
I=imresize(I,[300 300]);
[r c]=size(I);
LP=[-1 -1 -1; -1 8 -1; -1 -1 -1];
LN=[1 1 1; 1 -8 1; 1 1 1];
for x=2: r-1
   for y=2: c-1
      w=I(x-1:x+1,y-1:y+1);
      gi=double(w)+double(LP);
      gp(x,y)=gi(2,2);
      g=imsubtract(double(w),double(LN));
      gn(x,y)=g(2,2);

   end
end
H=3.*I;
gni=imresize(gn, [240 210]);
gpi=imresize(gp, [240 210]);
HN=imsubtract(H,gni);
HP=double(H)+double(gpi);
figure;imshow(gp,[]);title('Plus Laplacian Image');
figure;imshow(gn,[]);title('Negative Laplacian Image');
figure;imshow(HN,[]);title('HN Image');
figure;imshow(HP,[]);title('HP Image');
```

14. Gradient Operators%

```
clc;
close all;
clear all;
I=imread('pic1.jpg');
I=rgb2gray(I);
I=imresize(I,[256 256]);
[r c]=size(I);
LP=[-1 -1 -1;
   0 0 0;
   -1 -1 -1];       % Gradient with respect to Horizontal
LN=[-1 0 1;
   -1 0 1;
   -1 0 1];                   % Gradient with respect to Vertical
for x=2: r-1
   for y=2: c-1
      w=I(x-1:x+1,y-1:y+1);
      gi=double(w)+double(LP);
      gp(x,y)=gi(2,2);       %Gradient with respect to Horizontal
      g=imsubtract(double(w),double(LN));
```

```
        gn(x,y)=g(2,2);        % Gradient with respect to Vertical

    end
end
figure;imshow(I,[]);title('original image');
figure;imshow(gp,[]);title('horiz grad Image');
figure;imshow(gn,[]);title('ver grad Image');
```

15. Sobal Filter vertical

```
clc;
close all;
clear all;
i= imread('pic1.jpg');
i = rgb2gray(i);
[r c]=size(i);
f=[-1 0 1;-2 0 2;-1 0 1];
for x=2:r-1
    for y=2:c-1
    w=i(x-1:x+1,y-1:y+1);
    m(x,y)=sum(sum(double(w).*f));
    end
end
sub
```

16. Sobal Horizontal

```
clc;
close all;
clear all;
i= imread('pic1.jpg');
i = rgb2gray(i);
[r c]=size(i);
f=[-1 -2 -1;0 0 0;1 2 1];
for x=2:r-1
    for y=2:c-1
    w=i(x-1:x+1,y-1:y+1);
    m(x,y)=sum(sum(double(w).*f));
    end
end
subplot(1,2,1); imshow(i);
subplot(1,2,2); imshow(m);
```

17. Gaussian Filter

```
clc;
clear all;
close all;
I=imread('pic1.jpg');
I = imnoise(I,'salt & pepper',0.02);
PSF = fspecial('gaussian',10,2);
Blurred = imfilter(I,PSF);
x=imsubtract(I,Blurred);
imshow(x);
figure; imshow(Blurred);title('Blurred Image');
```

PRACTICE Four

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	1	Practical title	**Intensity Transformations and Spatial Filtering**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					
	Intensity Transformations and Spatial Filtering Photographic Negative Contrast-Stretching Transformations Logarithmic Transformations Spatial Filtering Predefined Filters Intensity Transformations and Spatial Filtering: Adjust image intensity values or colormap Intensity Transformations and Spatial Filtering: Adjust image intensity values or colormap				

Intensity Transformations and Spatial Filtering

When you are working with gray-scale images, sometimes you want to modify the intensity values. For instance, you may want to reverse black and the white intensities or you may want to make the darks darker and the lights lighter.

An application of intensity transformations is to increase the contrast between certain intensity values so that you can pick out things in an image. For instance, the following two images show an image before and after an intensity transformation.

Generally, making changes in the intensity is done through *Intensity Transformation Functions*.

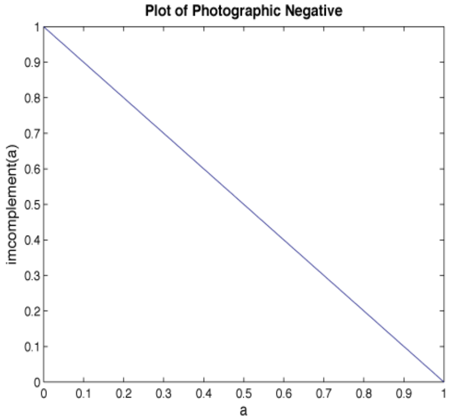
four main intensity transformation functions:

- photographic negative (using imcomplement)
- gamma transformation (using imadjust)
- logarithmic transformations (using c*log(1+f))
- contrast-stretching transformations (using 1./(1+(m./(double(f)+eps)).^E)

Photographic Negative

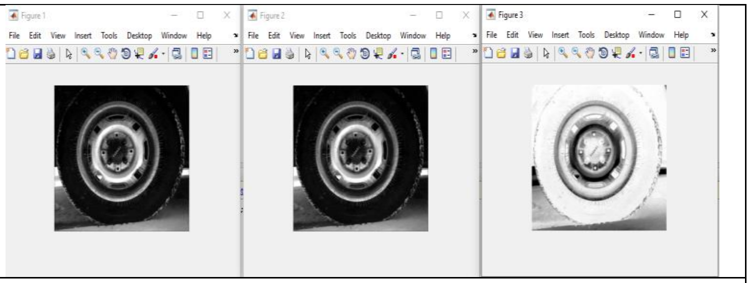
The Photographic Negative is probably the easiest of the intensity transformations to describe.

- Assume that we are working with grayscale double arrays where black is 0 and white is 1. The idea is that 0's become 1's, 1's become 0's, and any gradients in between are also reversed.
- In intensity, this means that the true black becomes true white and vise versa. MATLAB has a function to create photographic negatives--imcomplement(f).
- Given a=0:.01:1, the below shows a graph of the mapping between the original values (x-axis) and the imcomplement function.



Photographic Negative

Matlab Code	Result

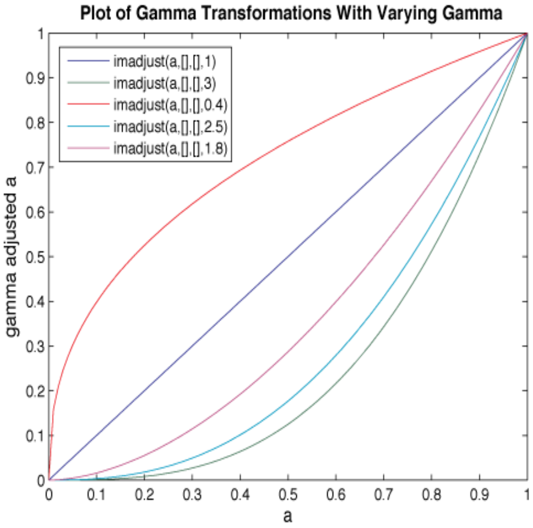
clc Clear all Close all I=imread('tire.tif'); imshow(I) J=imcomplement(I); figure, imshow(J)	 Figure 1 Figure 2 Figure 3 File Edit View Insert Tools Desktop Window Help
Important note about example :	

Gamma Transformations

With Gamma Transformations, you can curve the grayscale components either to brighten the intensity (when gamma is less than one) or darken the intensity (when gamma is greater than one).

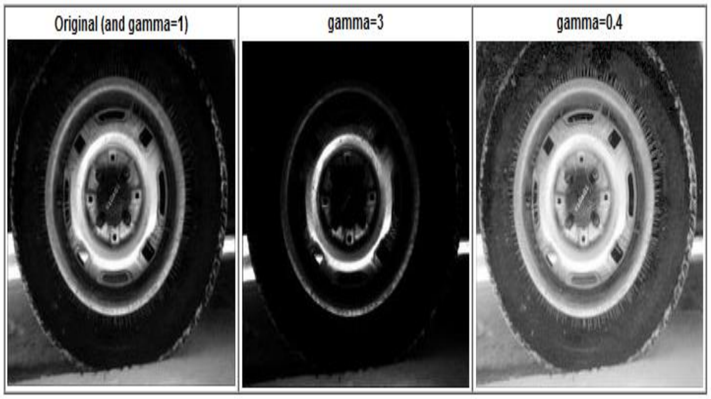
The MATLAB function that creates these gamma transformations is:

- imadjust(f, [low_in high_in], [low_out high_out], gamma)
  f is the input image, gamma controls the curve, and [low_in high_in] and [low_out high_out] are used for clipping.
- Values below low_in are clipped to low_out and values above high_in are clipped to high_out.
- For the purposes of this lab, we use [] for both [low_in high_in] and [low_out high_out].
- This means that the full range of the input is mapped to the full range of the output.
- Given a=0:.01:1, the following plots show the effect of the gamma transformation with varying gamma.
- Notice that the red line has gamma=0.4, which creates an upward curve and will brighten the image.



The following shows the results of three of the gamma transformations shown in the plot above. Notice how the values greater than 1 one create a darker image, whereas values between 0 and 1 create a brighter image with more contrast in dark areas so that you can see the details of the tire.

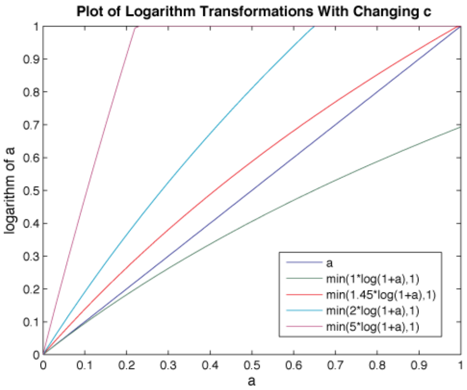
Matlab Code	Result

clc
clear all
close all
I=imread('tire.tif');
J=imadjust(I,[],[],1);
J2=imadjust(I,[],[],3);
J3=imadjust(I,[],[],0.4);
imshow(J);
figure,imshow(J2);
figure,imshow(J3);
	

Important note about example :	

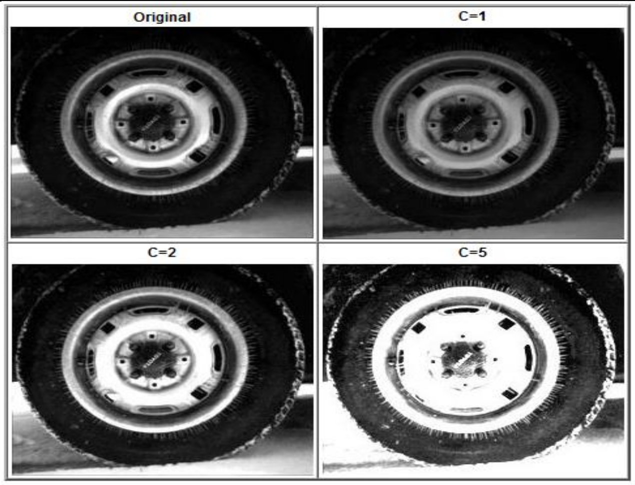
Logarithmic Transformations

Logarithmic Transformations can be used to brighten the intensities of an image (like the Gamma Transformation, where gamma < 1). More often, it is used to increase the detail (or contrast) of lower intensity values. They are especially useful for bringing out detail in Fourier transforms (covered in a later lab). In MATLAB, the equation used to get the Logarithmic transform of image f is:
g = c*log(1 + double(f))
The constant c is usually used to scale the range of the log function to match the input domain. In this case c=255/log(1+255) for a uint8 image, or c=1/log(1+1) (~1.45) for a double image. It can also be used to further increase contrast—the higher the c, the brighter the image will appear. Used this way, the log function can produce values too bright to be displayed. Given a=0:.01:1, the plot below shows the result for various values of c. The y-values are clamped at 1 by the min function for the plot of c=2 and c=5 (teal and purple lines, respectively).



- the original image and the results of applying three of the transformations from above.
- Notice that when c=5, the image is the brightest and you can see the radial lines on the inside of the tire (these lines are barely viewable in the original because there is not enough contrast in the lower intensities).

Matlab Code	Result

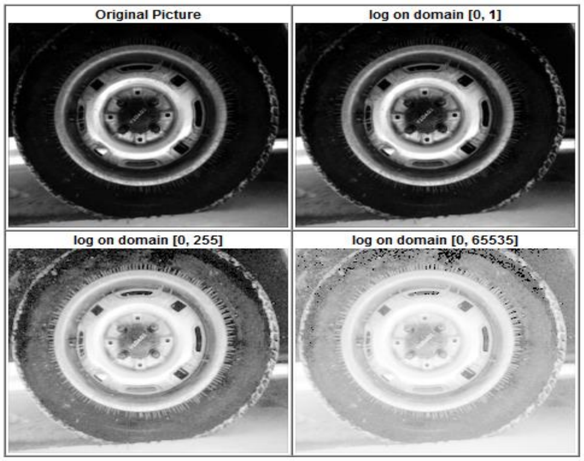
clc
clear all
close all
I=imread('tire.tif');
imshow(I)
I2=im2double(I);
J=1*log(1+I2);
J2=2*log(1+I2);
J3=5*log(1+I2);
figure, imshow(J)
figure, imshow(J2)
figure, imshow(J3)
	

Important note about example :	

Logarithmic Transformations

It is important to be aware of this effect if you plan to use logarithm transformations successfully, so here is the result of scaling an image's values to those ranges before applying the logarithm transform:

Matlab Code	Result

clc
clear all
close all
I=imread('tire.tif');
imshow(I)
I2=im2double(I);
J=1*log(1+I2);
J2=2*log(1+I2);
J3=5*log(1+I2);
figure, imshow(J)
figure, imshow(J2)
figure, imshow(J3)
	 Original Picture
	log on domain [0, 1]
	log on domain [0, 255]
	log on domain [0, 65535]

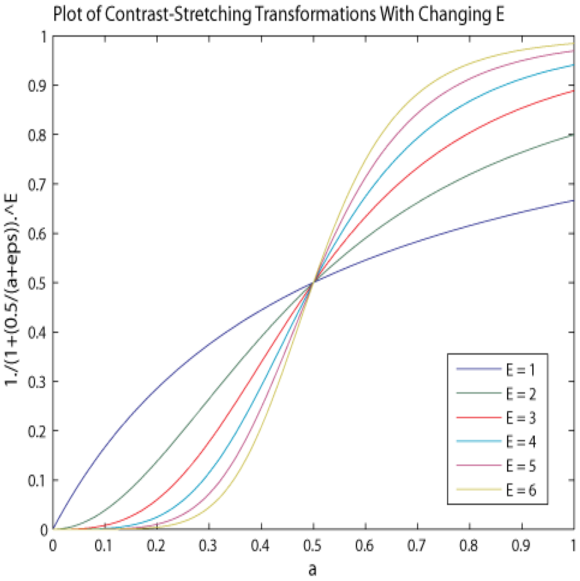
Important note about example :	

Note that for domain [0, 1] the effects of the logarithm transform are barely noticeable, while for domain [0, 65535] the effect is extremely exaggerated. Also note that, unlike with linear scaling and clamping, gross detail is still visible in light areas.

Contrast-Stretching Transformations

- Contrast-stretching transformations increase the contrast between the darks and the lights. In lab 1 we saw a simplified version of the automatic contrast adjustment in section 5.3 of the textbook. That transformation kept everything at relativelt similar intensities and merely stretched the histogram to fill the image's intensity domain. Sometimes you want to stretch the intensity around a certain level. You end up with everything darker darks being a lot darker and everything lighter being a lot lighter, with only a few levels of gray around the level of interest. To create such a contrast-stretching transformation in MATLAB, you can use the following function:

- g=1./(1 + (m./(double(f) + eps)).^E)
- E controls the slope of the function and m is the mid-line where you want to switch from dark values to light values. eps is a MATLAB constant that is the distance between 1.0 and the next largest number that can be represented in double-precision floating point. In this equation it is used to prevent division by zero in the event that the image has any zero valued pixels. There are two plot/diagram sets below to represent the results of changing both m and E. The below plot shows the results for several different values of E given a=0:.01:1 and m=0.5.



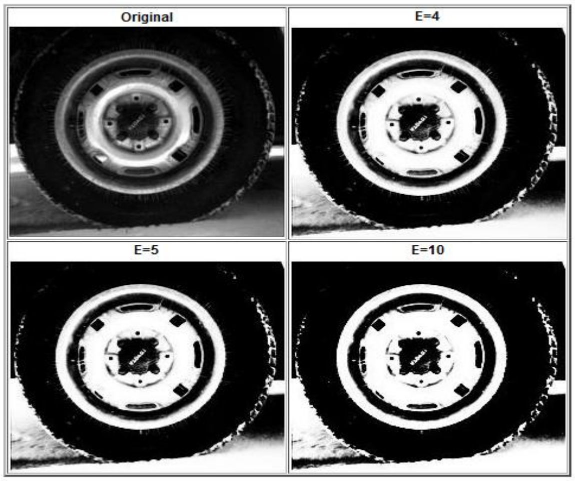
Contrast-Stretching Transformations

The following shows the original image and the results of applying the three transformations from above.

The m value used below is the mean of the image intensities (0.2104).

At very high E values, such as 10, the function becomes more like a thresholding function with threshold m—the resulting image is more black and white than grayscale.

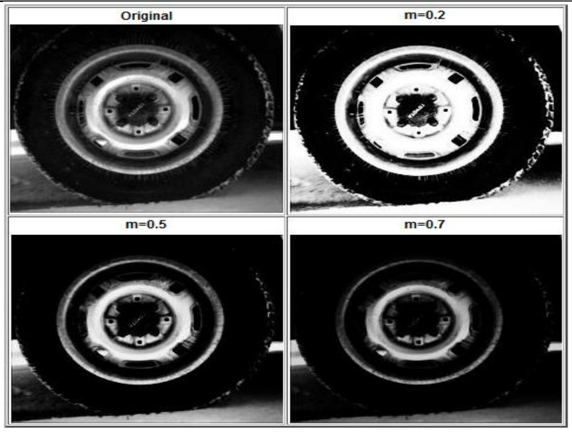
Matlab Code	Result

clc
clear all
close all
I=imread('tire.tif');
I2=im2double(I);
m=mean2(I2)
contrast1=1./(1+(m./(I2+eps)).^4);
contrast2=1./(1+(m./(I2+eps)).^5);
contrast3=1./(1+(m./(I2+eps)).^10);
imshow(I2)
figure,imshow(contrast1)
figure,imshow(contrast2)
figure,imshow(contrast3)
	

Important note about example :	

The following shows the original image and the results of applying the three transformations from above. The m value used below is 0.2, 0.5, and 0.7. Notice that 0.7 produces a darker image with fewer details for this tire image

Matlab Code	Result

clc
clear all
close all
I=imread('tire.tif');
I2=im2double(I);
contrast1=1./(1+(0.2./(I2+eps)).^4);
contrast2=1./(1+(0.5./(I2+eps)).^4);
contrast3=1./(1+(0.7./(I2+eps)).^4);
imshow(I2)
figure,imshow(contrast1)
figure,imshow(contrast2)
figure,imshow(contrast3)
	

Important note about example :	

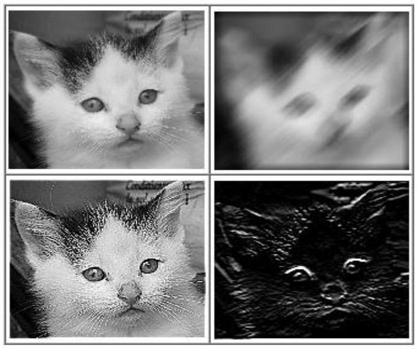
Spatial Filtering

There are two main types of filtering applied to images:

- – spatial domain filtering
- – frequency domain filtering

- In a later lab we will talk about frequency domain filtering, which makes use of the Fourier Transform.

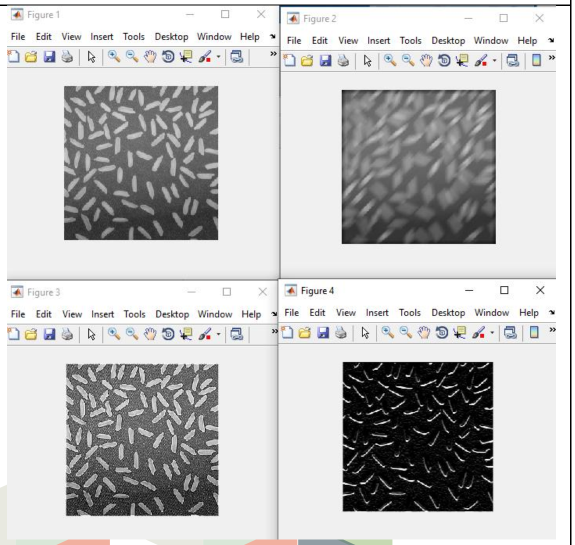
- For spatial domain filtering, we are performing filtering operations directly on the the pixels of an image.
- Spatial filtering is a technique that uses a pixel and its neighbors to select a new value for the pixel.
- The simplest type of spatial filtering is called linear filtering. It attaches a weight to the pixels in the neighborhood of the pixel of interest, and these weights are used to blend those pixels together to provide a new value for the pixel of interest.
- Linear filtering can be uses to smooth, blur, sharpen, or find the edges of an image. The following four images are meant to demonstrate what spatial filtering can do. The original image is shown in the upper left-hand corner.



Sometimes a linear filter is not enough to solve a particular problem. In that case it is possible to use higher order math or full-blown MATLAB functions produce specialized results. Such non-linear filters are useful for smoothing only smooth areas, enhancing only strong edges or removing speckles from images.

Predefined Filters

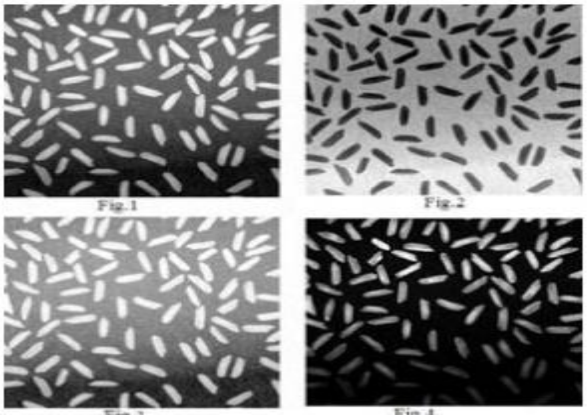
Matlab Code	Result

close all
%original picture
cat=imread('rice.png');
figure, imshow(cat)
 	
%motion blur
h=fspecial('motion', 20, 45);
cat_motion=imfilter(cat,h);
figure, imshow(cat_motion)
 	
%sharpening
%see section 7.6 (esp 7.6.2)
h=fspecial('unsharp');
cat_sharp=imfilter(cat,h);
figure, imshow(cat_sharp)
 	
%horizontal edge-detection
%see section 7.2 and 7.3.1
h=fspecial('sobel');
cat_sobel=imfilter(cat,h);
	 Figure 1
	Figure 2
	Figure 3
	Figure 4
	File Edit View Insert Tools Desktop Window Help

`figure, imshow(cat_sobel)`	
Important note about example :	

Intensity Transformations and Spatial Filtering:
Adjust image intensity values or colormap

Matlab Code	Result

`f= imread('rice.png');`
`g1=imadjust(f,[0 1],[1 0]);`
`g2=imadjust(f,[ ],[ ],.05);`
`g3=imadjust(f,[ ],[ ],5);`
`figure(1);imshow(f,[ ])`
`figure(2);imshow(g1,[ ])`
`figure(3);imshow(g2,[ ])`
`figure(4);imshow(g3,[ ])`
	

Important note about example :	

Spatial Filtering

There are two main types of filtering applied to images:

- spatial domain filtering
- frequency domain filtering
- For spatial domain filtering, we are performing filtering operations directly on the the pixels of an image.

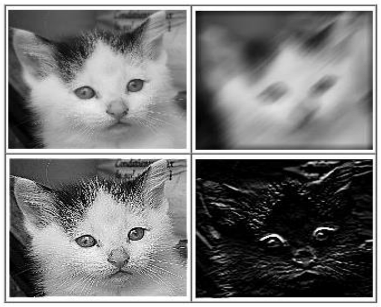
Spatial filtering is a technique that uses a pixel and its neighbors to select a new value for the pixel.

The simplest type of spatial filtering is called linear filtering.

It attaches a weight to the pixels in the neighborhood of the pixel of interest, and these weights are used to blend those pixels together to provide a new value for the pixel of interest.

Linear filtering can be uses to smooth, blur, sharpen, or find the edges of an image.

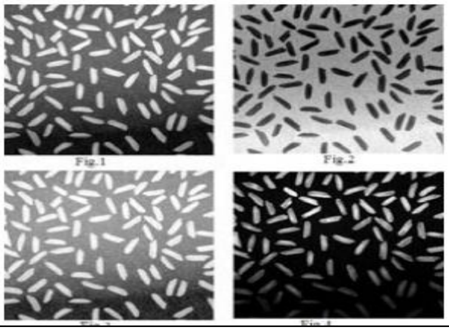
The following four images are meant to demonstrate what spatial filtering can do. The original image is shown in the upper left-hand corner.



Sometimes a linear filter is not enough to solve a particular problem. In that case it is possible to use higher order math or full-blown MATLAB functions produce specialized results. Such non-linear filters are useful for smoothing only smooth areas, enhancing only strong edges or removing speckles from images.

Intensity Transformations and Spatial Filtering: Adjust image intensity values or colormap

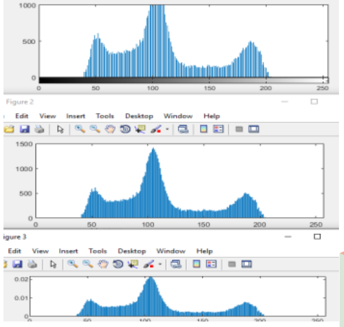
Matlab Code	Result

f= imread('rice.png');
g1=imadjust(f,[0 1],[1 0]);
g2=imadjust(f,[ ],[ ],.05);
g3=imadjust(f,[ ],[ ],5);
figure(1);imshow(f,[ ])
figure(2);imshow(g1,[ ])
figure(3);imshow(g2,[ ])
figure(4);imshow(g3,[ ])
	

Important note about example :	

Type of histogram showing

Matlab Code	Result

close all
clc
clear all
f= imread('rice.png');
h=imhist(f);
figure(1);imhist(f)
figure(2);bar(h)
figure(3);bar(h/numel(f))
	 1000
	500
	0
	0 50 100 150 200 250
	Figure 2
	Edit View Insert Tools Desktop Window Help
	1500
	1000
	500
	0
	0 50 100 150 200 250
	igure 3
	Edit View Insert Tools Desktop Window Help
	0.02
	0.01

Important note about example :	

Mix last two program

Matlab Code	Result

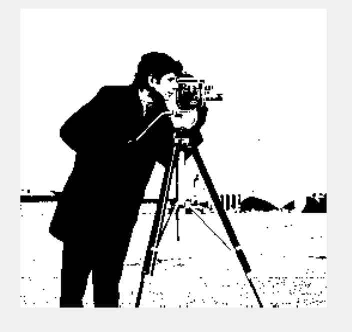
clc
clear all
close all
f= imread('rice.png');
h=imhist(f);
g1=imadjust(f,[0 1],[1 0]);
g2=imadjust(f,[ ],[ ],.05);
g3=imadjust(f,[ ],[ ],5);
figure(1);imshow(f,[ ])
figure(2);imshow(g1,[ ])
figure(3);imshow(g2,[ ])
figure(4);imshow(g3,[ ])
figure(5);imhist(f)
figure(6);imhist(g1)
figure(7);imhist(g2)	

`figure(8);imhist(g3)`	
Important note about example :	

Intensity Transformations and Spatial Filtering

One of the method to find the good threshold is using the OUSTO method which it use in matlab with graythresh .In this method the aim is to find the proper threshold and minise the intraclass variance in the binary image

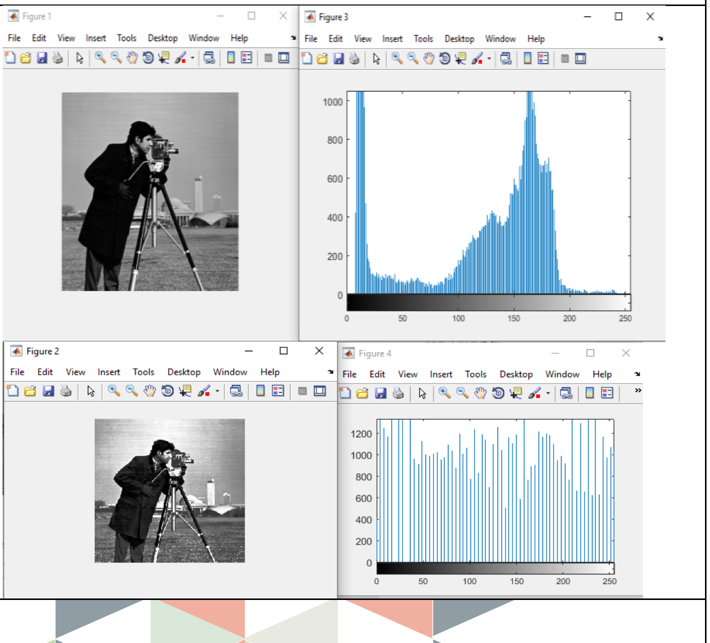
Matlab Code	Result

`close all`
`clc`
`clear all`
`I = imread('cameraman.tif');`
`level = graythresh(I);`
`BW = im2bw(I,level);`
`imshow(BW)`
	

Important note about example :	

Intensity Transformations and Spatial Filtering_Histogram Equalization

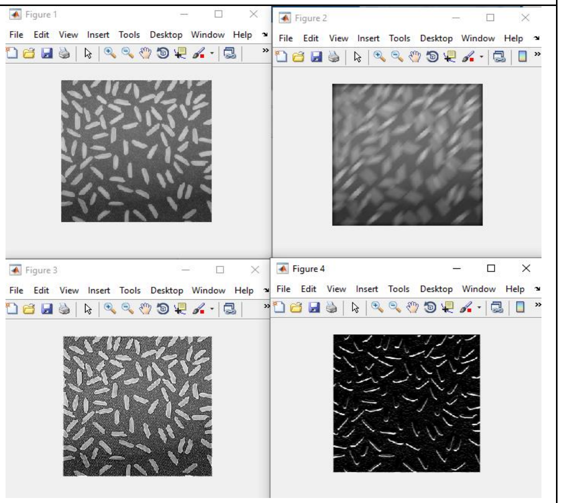
Matlab Code	Result

`close all`
`clc`
`clear all`
`f= imread('cameraman.tif');`
`g=histeq(f);`
`figure(1);imshow(f,[])`
`figure(2);imshow(g,[])`
`figure(3);imhist(f)`
`figure(4);imhist(g)`
	 Figure 1
	File Edit View Insert Tools Desktop Window Help
	Figure 3
	File Edit View Insert Tools Desktop Window Help
	1000
	800
	600
	400
	200
	0
	0 50 100 150 200 250
	Figure 2
	File Edit View Insert Tools Desktop Window Help
	Figure 4
	File Edit View Insert Tools Desktop Window Help
	1200
	1000
	800
	600
	400
	200
	0
	0 50 100 150 200 250

Important note about example :	

Predefined Filters

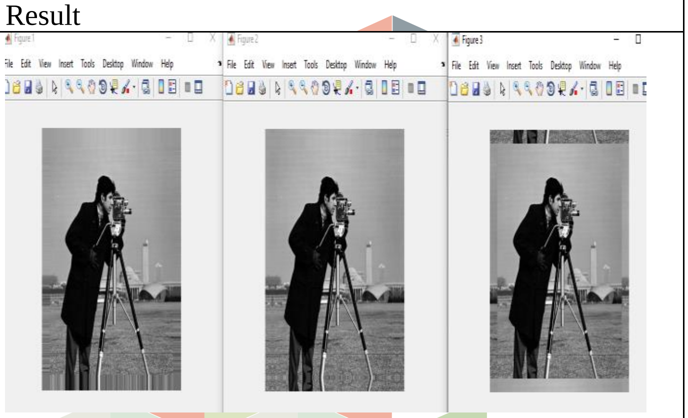
Matlab Code	Result

close all
%original picture
cat=imread('rice.png');
figure, imshow(cat)
 	
%motion blur
h=fspecial('motion', 20, 45);
cat_motion=imfilter(cat,h);
figure, imshow(cat_motion)
 	
%sharpening
h=fspecial('unsharp');
cat_sharp=imfilter(cat,h);
figure, imshow(cat_sharp)
 	
%horizontal edge-detection
h=fspecial('sobel');
cat_sobel=imfilter(cat,h);
figure, imshow(cat_sobel)
	 Figure 1
	Figure 2
	Figure 3
	Figure 4
	File Edit View Insert Tools Desktop Window Help

Important note about example :	

Intensity Transformations and Spatial Filtering_Correlation_

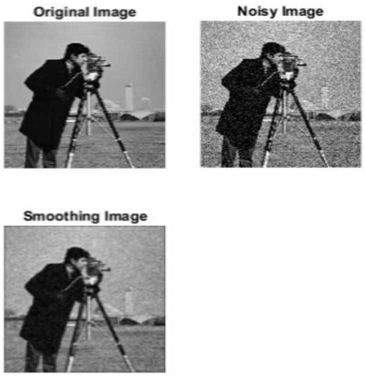
Matlab Code	Result

close all
clc
clear all
f= imread('cameraman.tif');
w=zeros(31);
w(16,16)=1;
g1=imfilter(f,w,'corr','replicate','full');
g2=imfilter(f,w,'corr','symmetric','full');
g3=imfilter(f,w,'corr','circular','full');
figure(1);imshow(g1,[])
figure(2);imshow(g2,[])
figure(3);imshow(g3,[])
	 Figure 1
	Figure 2
	Figure 3
	File Edit View Insert Tools Desktop Window Help

Important note about example :	

Intensity Transformations and Spatial Filtering_gaussian

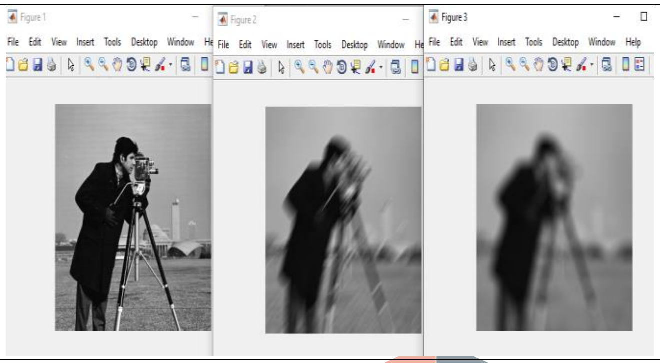
Matlab Code	Result

clc
clear all
close all
f= imread('cameraman.tif');
fn=imnoise(f,'gaussian');
w=ones(3)/9;
g=imfilter(fn,w);
subplot(2,2,1);imshow(f);title('Original Image');
subplot(2,2,2);imshow(fn);title('Noisy Image');
subplot(2,2,3);imshow(g);title('Smoothing Image');
	

Important note about example :	

Create Various Filters and Filter an Image

Matlab Code	Result

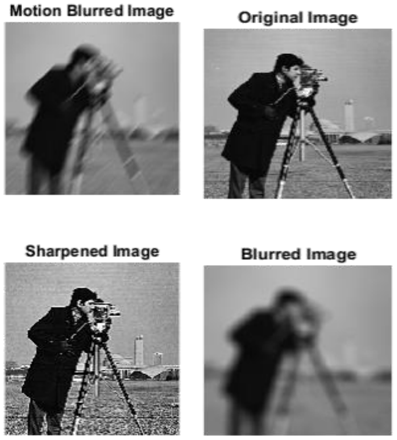
clc
clear all
close all
I = imread('cameraman.tif');
imshow(I);
H = fspecial('motion',20,45);
MotionBlur = imfilter(I,H,'replicate');
figure(); imshow(MotionBlur);
H = fspecial('disk',10);
blurred = imfilter(I,H,'replicate');
figure(); imshow(blurred);
	 Figure 1
	Figure 2
	Figure 3
	File Edit View Insert Tools Desktop Window Help

Important note about example :	

Intensity Transformations and Spatial Filtering

Matlab Code	Result

```
clc
clear all
close all
I = imread('cameraman.tif');
subplot(2,2,1);imshow(I);title('Original Image');
w1 = fspecial('motion',20,45);
MotionBlur = imfilter(I,w1,'replicate');
subplot(2,2,2);imshow(MotionBlur);title('Motion Blurred
Image');
w2 = fspecial('disk',10);
blurred = imfilter(I,w2,'replicate');
subplot(2,2,3);imshow(blurred);title('Blurred Image');
w3 = fspecial('unsharp');
sharpened = imfilter(I,w3,'replicate');
subplot(2,2,4);imshow(sharpened);title('Sharpened Image');
```

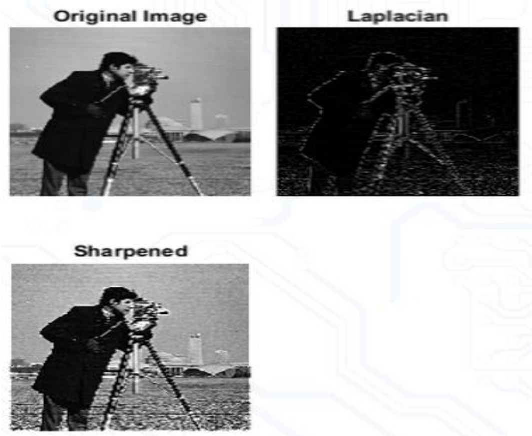


Important note about example :

Intensity Transformations and Spatial Filtering

Matlab Code | Result

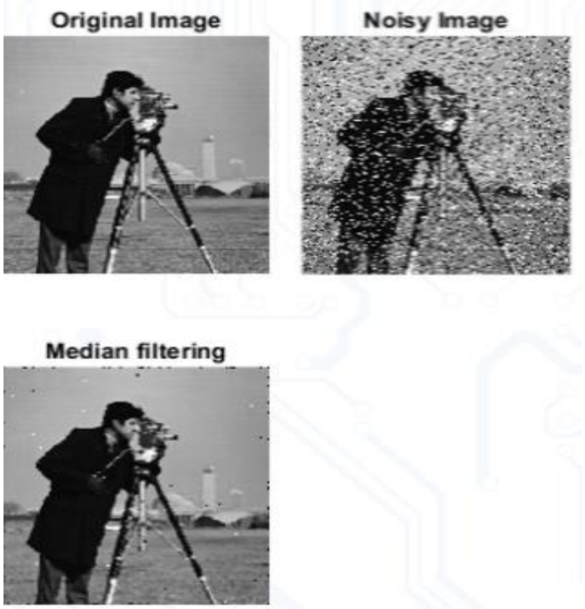
```
 clc
clear all
close all
f= imread('cameraman.tif');
f=im2double(f);
w=fspecial('laplacian',0);
g1=imfilter(f,w);
g2=f-g1;
subplot(2,2,1);imshow(f);title('Original
Image');
subplot(2,2,2);imshow(g1);title('Laplacian');
subplot(2,2,3);imshow(g2);title('Sharpened')
```



Important note about example :

Matlab Code | Result

```
 close all
clc
clear all
f= imread('cameraman.tif');
fn=imnoise(f,'salt & pepper', .2);
g=medfilt2(fn);
subplot(2,2,1);imshow(f);title('Original
Image');
subplot(2,2,2);imshow(fn);title('Noisy Image');
subplot(2,2,3);imshow(g);title('Median
filtering');
```



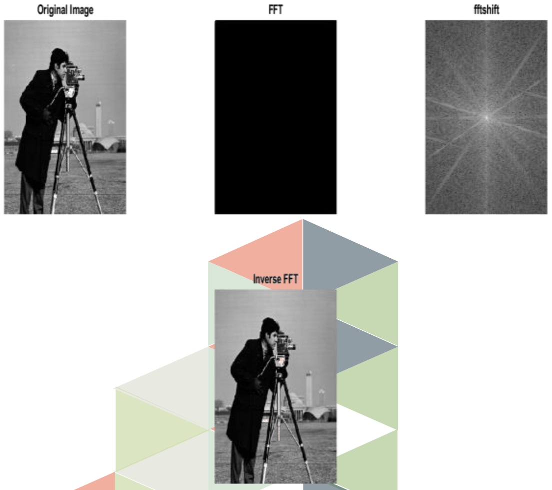
Important note about example :

Frequency Domain Processing

Matlab Code

```
 clc
close all
clear all
f=imread('cameraman.tif');
F=fft2(f);
s=abs(F);
s1=log(1+s);
s2=fftshift(s1)
f1=real(ifft2(F));
subplot(2,3,1);imshow(f,[]);title('Original
Image');
subplot(2,3,2);imshow(s,[]);title('FFT');
subplot(2,3,3);imshow(s1,[]);title('LOG
transform');
subplot(2,3,3);imshow(s2,[]);title('fftshift')
;
subplot(2,3,5);imshow(f1,[]);title('Inverse
FFT');
```

Result

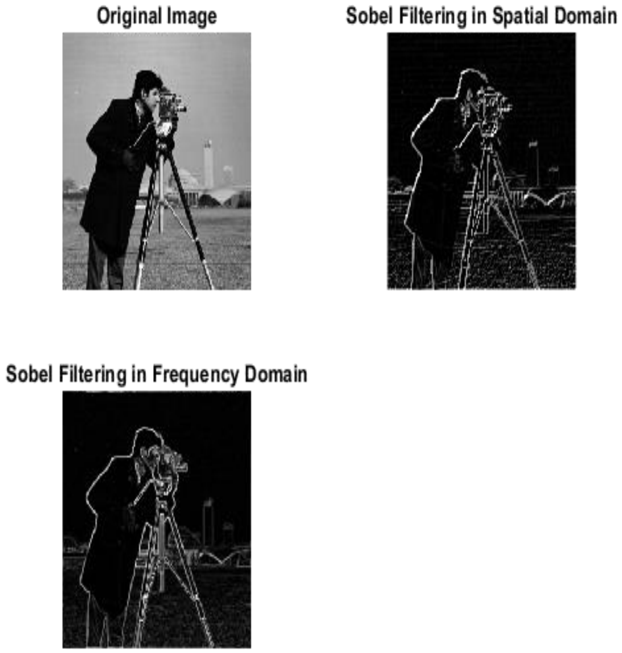


Important note about example :

Matlab Code

```
 clc
close all
clear all
f=imread('cameraman.tif');
h1=fspecial('sobel');
%horizontal mask in spatial domain
h2=h1'; %vertical mask in spatial domain
gs1=imfilter(f,h1); %horizontal edges
gs2=imfilter(f,h2); %vertical edges
gs=imadd(gs1,gs2,'uint16');
[R,C]=size(f);
H1=fftshift(freqz2(h1,R,C)); %horizontal
filter in frequency domain
H2=fftshift(freqz2(h2,R,C)); %vertical
filter in frequency domain
F=fft2(f);
GF1=H1.*F; % Filtering in frequency domain
GF2=H2.*F; % Filtering in frequency domain
gf1=abs(real(ifft2(GF1))); %horizontal
edges
gf2=abs(real(ifft2(GF2))); %vertical edges
gf=imadd(gf1,gf2,'uint16');
subplot(2,2,1);imshow(f,[]);title('Original
Image');
subplot(2,2,2);imshow(gs,[]);title('Sobel
Filtering in Spatial Domain');
subplot(2,2,3);imshow(gf,[]);title('Sobel
Filtering in Frequency Domain');
```

Result

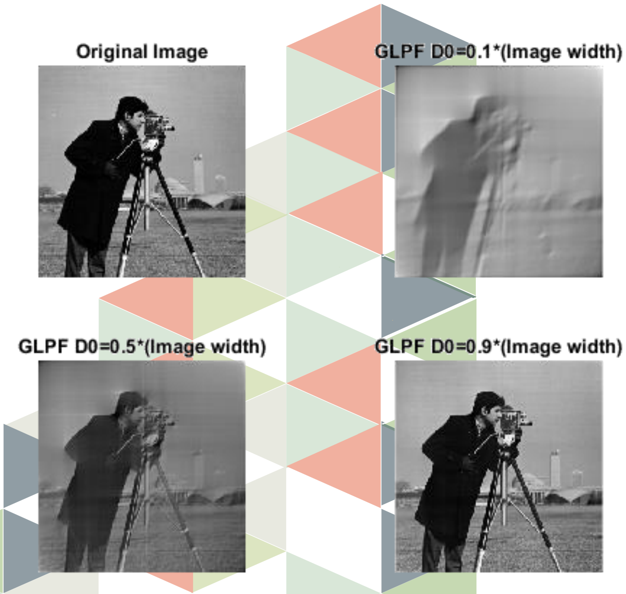

Original Image
Sobel Filtering in Spatial Domain
Sobel Filtering in Frequency Domain

Important note about example :

Matlab Code

```
clc
close all
clear all
f=imread('cameraman.tif');
[R,C]=size(f);
P=2*R-1;Q=2*C-1;
F=fft2(f,P,Q);
u=0:P-1; v=0:Q-1;
[V,U]=meshgrid(v,u);
D0=.1*(Q);
H=exp(-(U.^2+V.^2)/(2*(D0^2)));
g1=real(ifft2(H.*F));
g11=g1(1:R,1:C);
subplot(2,2,1);imshow(f,[]);title('Original
Image');
subplot(2,2,2);imshow(g11,[]);title('GLPF
D0=0.1*(Image width)');
D0=.5*(Q);
H=exp(-(U.^2+V.^2)/(2*(D0^2)));
g2=real(ifft2(H.*F));
g22=g2(1:R,1:C);
subplot(2,2,3);imshow(g22,[]);title('GLPF
D0=0.5*(Image width)');
D0=.9*(Q);
```

Result


Original Image
GLPF D0=0.1*(Image width)
GLPF D0=0.5*(Image width)
GLPF D0=0.9*(Image width)

H=exp(-(U.^2+V.^2)/(2*(D0^2)));
g3=real(ifft2(H.*F));
g33=g3(1:R,1:C);
subplot(2,2,4);imshow(g33,[]);title('GLPF D0=0.9*(Image width)');	

Important note about example :	

Practice Five

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	5	Practical title	**Applications of Convolution in Image Processing with MATLAB**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					

The purpose and requirement (fill in)

Purpose:

•

Requirement:

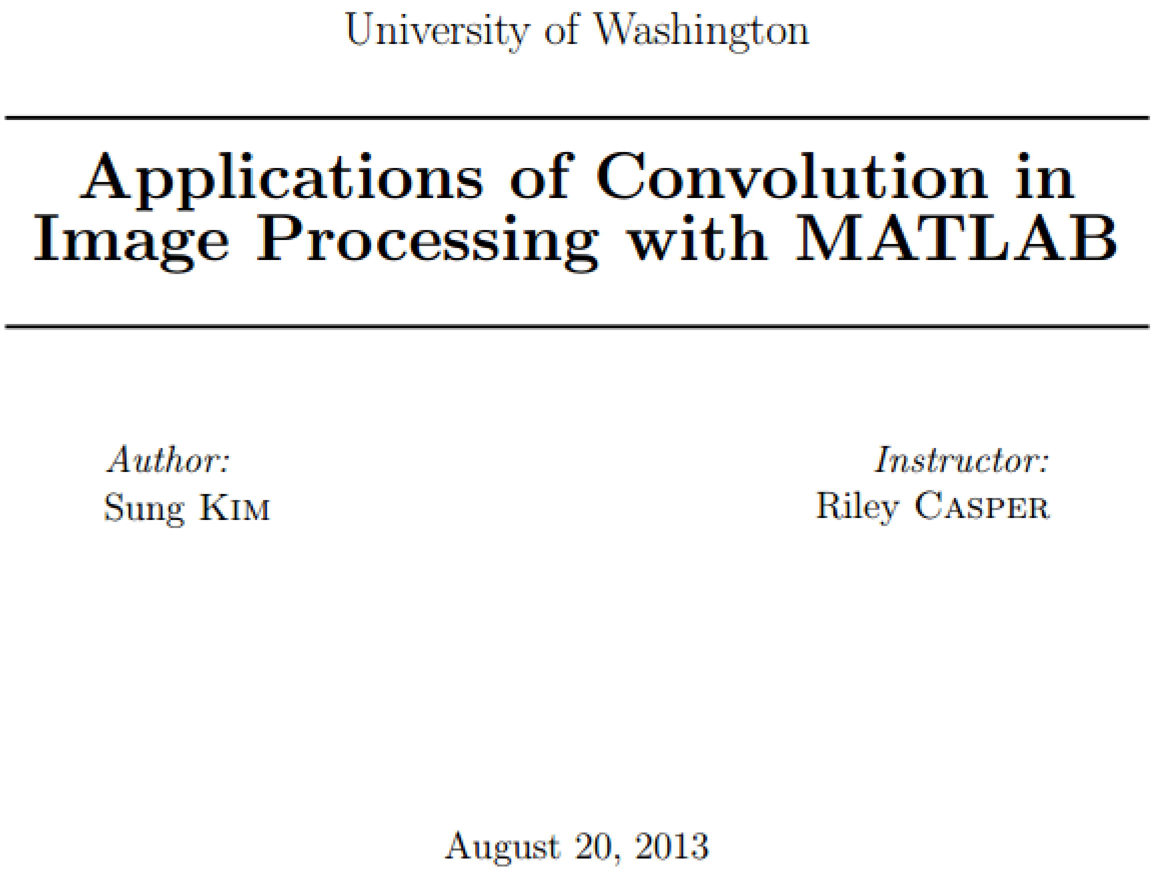
•

Every student have lecture slide and file which I send.

Content:

a)

Follow **the** pdf file about convolution and repeat the coding part



University of Washington

Applications of Convolution in Image Processing with MATLAB

Author:
Sung Kim

Instructor:
Riley Casper

August 20, 2013

Practice Six

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	5	Practical title	**Image Segmentation**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					

The purpose and requirement (fill in)

Purpose:

Requirement:

- Each student must have resources(computer)
- Every student have installed MATLAB2014a version.

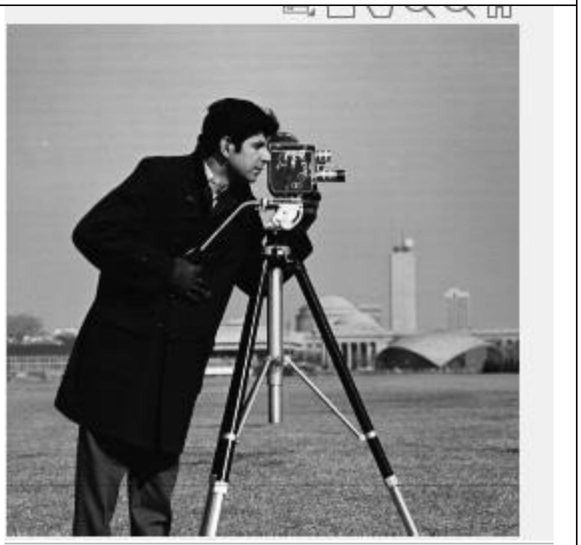
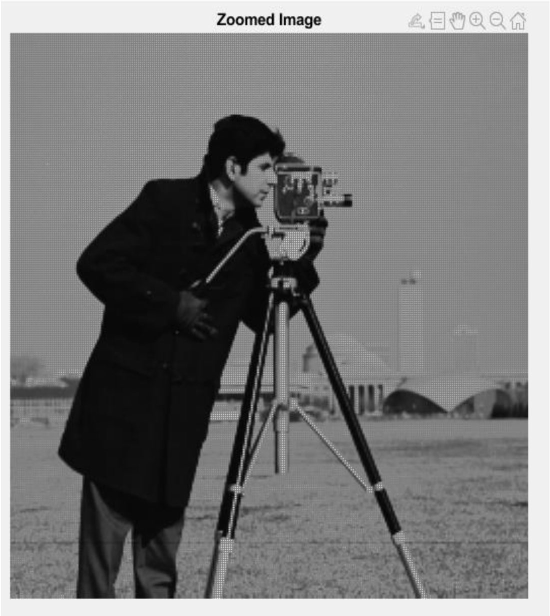
Every student have lecture slide and file which I send.

Content:

b) Program to zoom the image: test and report with different number
c) Program to digital negative the image: test and report with different number
d)

%Program to zoom the image: test and report with different number

Matlab Code	Result

clear all;
close all;
f1 = input('Enter the factor by which the image is to be Zoomed: ');
A = imread('cameraman.tif');
s = size(A);
s2 = s*f1;
k = 1;
l = 1;
for (i=1:f1:s2)
    for(j=1:f1:s2)
        C(i,j) = A(k,l);
        l = l+1;
    end
    l = 1;
    k = k+1;
end
 	
for (i=1:f1:s2)
    for (j=2:f1:s2-1)
        C(i,j) = [C(i,j-1)+ C(i, j+1)]*0.5;
    end
end
 	
for(j=1:f1:s2)
    for(i=2:f1:s2-1)
        C(i,j) = [C(i-1,j)+C(i+1,j)]*0.5;
    end
end
 	
for (i=2:f1:s2-1)
    for (j=2:f1:s2-1)
        C(i,j) = [C(i,j-1)+ C(i, j+1)]*0.5;
    end
end
figure,imshow(C);
title('Zoomed Image');
	
	

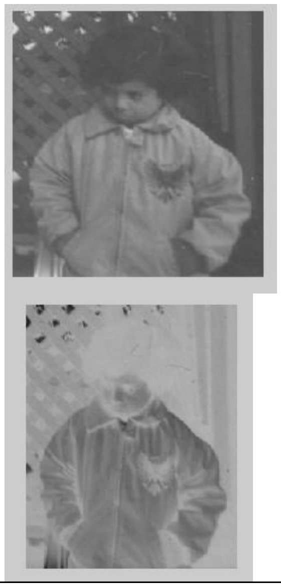
Important note about example :	

% Program to digital negative the image: test and report with different number

Matlab Code

Result

```
  clear all;
close all;
%Negative
clc;
clear all;
a = imread('pout.tif');
%a=zeros(10,10);
b=255-a;

subplot(2,2,1);
imshow(a);
subplot(2,2,2);
imshow(b);
c=imcrop(a(x,y),100,100);
subplot(2,2,3);
imshow(c);
```



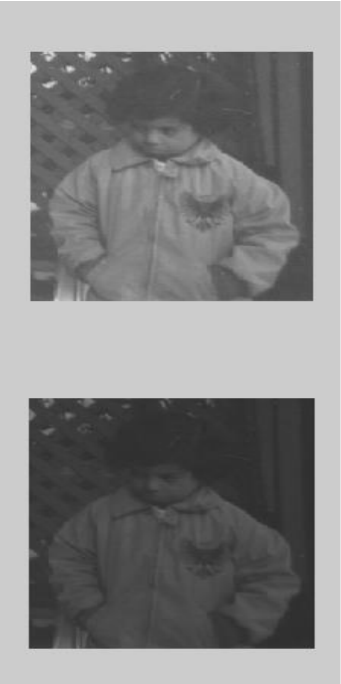
Important note about example :

% Program to contrast stretching the image: test and report with different number

Matlab Code

Result

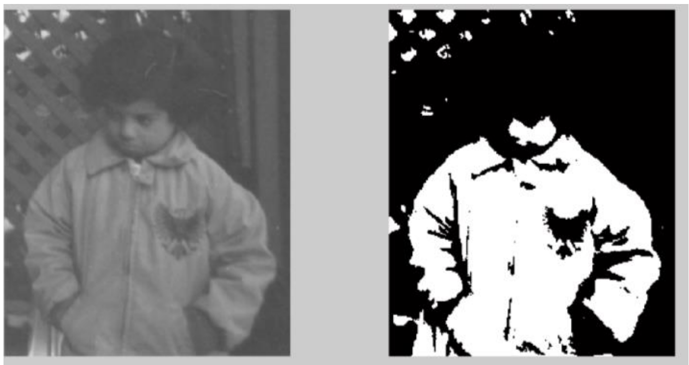
```
   %Contrast stretching
%lt=5 and ut=10
close all;
close all;
clc;
a=imread('pout.tif');
subplot(2,1,1);
imshow(a);
[m,n]=size(a);
b=a;
t1=input('enter threshold value1');
t2=input('enter threshold value2');
for i=1:1:m
    for j=1:1:n
        if(a(i,j)<t1)
            b(i,j)=0.5*t1;
        else if(a(i,j)<t2 & a(i,j)>t1)
                b(i,j)=2*(a(i,j)-t1) + 0.5*t1;
            else
                b(i,j)=0.5*(a(i,j)-t2) + 2*(t2-t1) + 0.5*t1;
            end
        end
    end
end
subplot(2,1,2)
imshow(b);
```



Important note about example :

% Program to image thresholding the image:test and report with different number

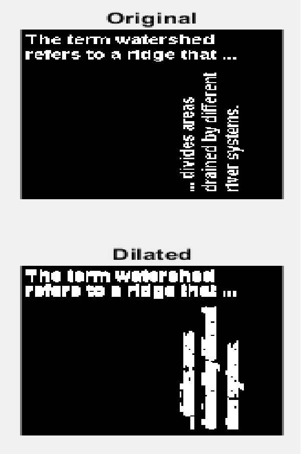
Matlab Code	Result

clc
clear all
close all
 %thresholding
%threshold=120
clear all;
a=imread('pout.tif');
[m n]=size(a);
 	
t=input('enter the threshold value');
for i=1:1:m
 for j=1:1:n
    if (a(i,j)<t)
        b(i,j)=0;
    else
    b(i,j)=255;
    end
 end
end
subplot(1,2,1)
imshow(a);
subplot(1,2,2)
imshow(b);
	

Important note about example :	

%% Program to Morphology Dilation of the image

Matlab Code	Result

%Dilation
A=imread('text.png');
A=im2bw(A);
%Structuring element
B2=getnhood(strel('line',7,90));
m=floor(size(B2,1)/2);
n=floor(size(B2,2)/2);
%Pad array on all the sides
C=padarray(A,[m n]);
D=false(size(A));
for i=1:size(C,1)-(2*m)
    for j=1:size(C,2)-(2*n)
        Temp=C(i:i+(2*m),j:j+(2*n));
        D(i,j)=max(max(Temp&B2));
    end
end
subplot(2,1,1),
imshow(A), title('Original')
subplot(2,1,2),
imshow(D), title('Dilated')
	 Original
	The term watershed refers to a ridge that ...
	... divides areas drained by different river systems.
	Dilated
	The term watershed refers to a ridge that ...

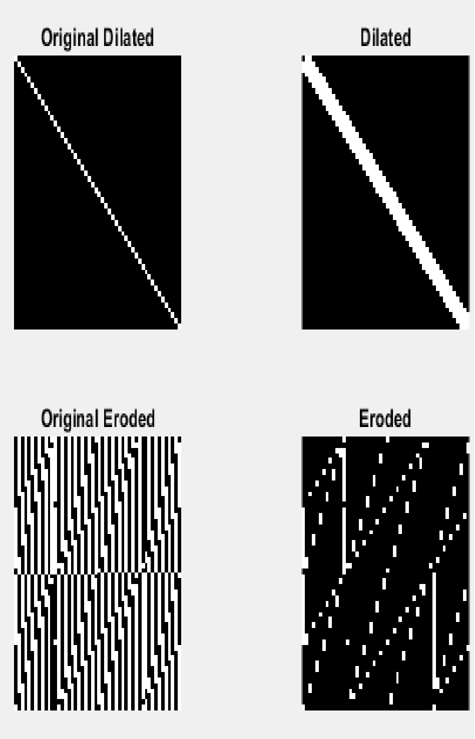
Important note about example :	

%% Program to morphology Dilation Erosion the image

Matlab Code

```
 clc
clear all
close all
%Dilation Erosion
A=eye(50);
Acom= magic(50);
for i=1:50
    for j=1:50
        Acom(i,j)=rem(Acom(i,j),2);
    end
end
%Structuring element
B=[1 0 1; 0 0 1; 1 1 0];
%Pad zeros on all the sides
C=padarray(A,[1 1]);
%Intialize a matrix of matrix size A with zeros
D=false(size(A));
for i=1:size(C,1)-2
    for j=1:size(C,2)-2
        %Perform logical AND operation
        D(i,j)=sum(sum(B&C(i:i+2,j:j+2)));
    end
end
%Structuring element
B1=[1 1 0];
%Pad array with ones on both sides
E=padarray(Acom,[0 1],1);
%Intialize the matrix D of size A with zeros
F=false(size(Acom));
for i=1:size(E,1)
    for j=1:size(E,2)-2
        In=E(i,j:j+2);
        %Find the position of ones in the structuring element
        In1=find(B1==1);
        %Check whether the elements in the window have the value one in the
        %same positions of the structuring element
        if(In(In1)==1)
        F(i,j)=1;
        end
    end
end
subplot(2,2,1),
imshow(A), title('Original Dilated');
subplot(2,2,2),
imshow(D), title('Dilated');
subplot(2,2,3),
imshow(Acom), title('Original Eroded');
subplot(2,2,4),
imshow(F), title('Eroded');
```

Result



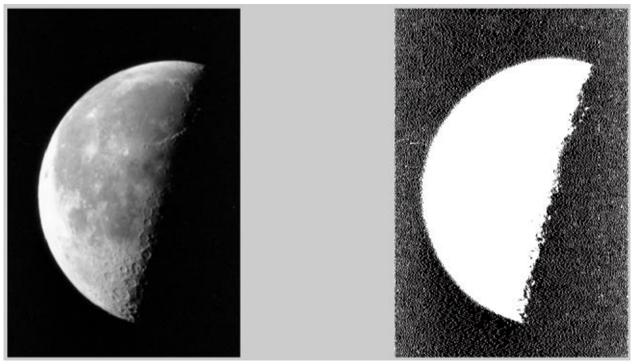
Important note about example :

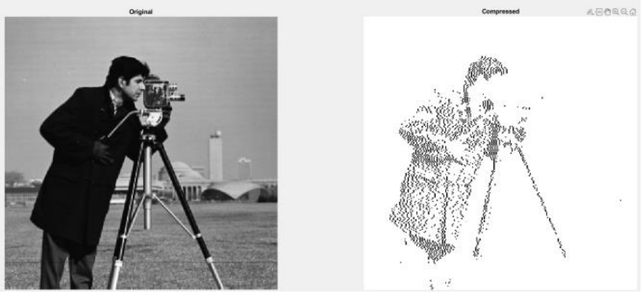
%% Program to Improved Grayscale Quantization the image:test with different image and report

Matlab Code

```
%IGS
clc;
clear all;
a=imread('cameraman.tif');
a=double(a);
%a=[12 12 13 13 57 54];
[row col]=size(a)
temp=dec2bin(0,8);
for x=1:1:row-1
    for y=1:1:col-1
        q=dec2bin(a(x,y),8);
        q1=[q(1) q(2) q(3) q(4)];
        a1(x,y)=bin2dec(q1);
        if a(x,y)>=240
            temp2=0;
        else
            temp1=[temp(5) temp(6) temp(7) temp(8)];
            temp2=bin2dec(temp1);
        end
        c=a(x+1,y+1)+temp2;
        c1=dec2bin(c,8);
        temp=c1;
        igs1=[c1(1) c1(2) c1(3) c1(4)];
        code(x,y)=bin2dec(igs1);
    end
end

subplot(1,2,2)
imshow(code);
title('Compressed');
subplot(1,2,1)
imshow(uint8(a));
title('Original');
```

Result





Important note about example :

Practice Seven

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	5	Practical title	**Image Segmentation**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					

The purpose and requirement (fill in)

Purpose:

Requirement:

- Each student must have resources(computer)
- Every student have installed MATLAB2014a version.

Every student have lecture slide and file which I send.

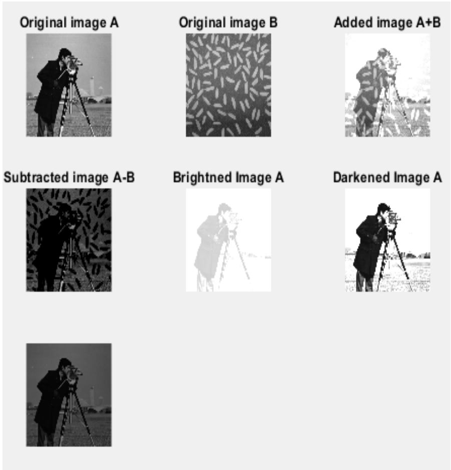
Content:

e)

%% Program to Discrete Fourier arithmetic operation in the image: test with different image and report

Matlab Code | Result

```
%Arithmetic
clc;
clear all;
a = imread('cameraman.tif'); %Read Images
b = imread('rice.png');
I = imadd(a,b);             % Use inbuilt MatLab functions to Add and Subtract A & B.
Y = imsubtract(a,b);
Z = mean2(a); %Mean of all pixel values of Image A.
disp(Z);
p = a+200;
disp(p);
V = a.*2; %Change the mean value in order to change the brightness.
B = a./2;
subplot(3,3,1);imshow(a);Title('Original image A');
subplot(3,3,2);imshow(b);Title('Original image B');
subplot(3,3,3);imshow(I);Title('Added image A+B');
subplot(3,3,4);imshow(Y);Title('Subtracted image A-B');
subplot(3,3,5);imshow(p);Title('Brightned Image A');
subplot(3,3,6);imshow(V);Title('Darkened Image A');
subplot(3,3,7);imshow(B);

%mean value obtained=118.7245
```

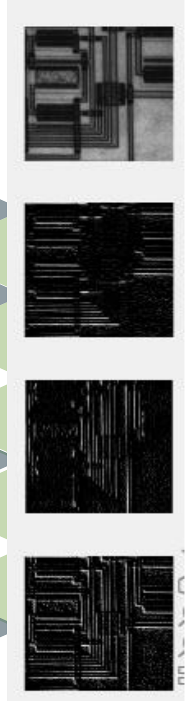


Important note about example:

%% Program to Edge Detection with prewitt in the image: test with different image and report

Matlab Code | Result

```
%Prewitt
clc;
clear all;
a=imread('circuit.tif');
[x y]=size(a);
w=[-1 0 1; -1 0 1; -1 0 1];
w1=[-1 -1 -1; 0 0 0; 1 1 1];
%BW2 = edge(a,'Prewitt');
a=double(a);
for m=2:1:x-1
    for n=2:1:y-1
        r(m,n)=w(1)*a(m-1,n-1)+ w(2)*a(m,n-1)+w(3)*a(m+1,n-1)+w(4)*a(m-1,n)+w(5)*a(m,n)+w(6)*a(m+1,n)+w(7)*a(m-1,n+1)+w(8)*a(m,n+1)+w(9)*a(m+1,n+1);

       r1(m,n)=w1(1)*a(m-1,n-1)+ w1(2)*a(m,n-1)+w1(3)*a(m+1,n-1)+w1(4)*a(m-1,n)+w1(5)*a(m,n)+w1(6)*a(m+1,n)+w1(7)*a(m-1,n+1)+w1(8)*a(m,n+1)+w1(9)*a(m+1,n+1);
    end
end
z=r+r1;
%for m=1:1:x
 %   for n=1:1:y
subplot(1,4,1);imshow(uint8(a));
subplot(1,4,2);imshow(uint8(r));
subplot(1,4,3);imshow(uint8(r1));
subplot(1,4,4);imshow(uint8(z));
```



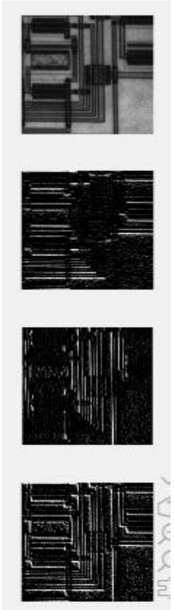
Important note about example:

%% Program to Edge Detection with sobel in the image:test with different image and report

Matlab Code

Result

```
 %Sobel
clc;
clear all;
a=imread('circuit.tif');
[x y]=size(a);
w=[-1 0 1; -2 0 2; -1 0 1];
w1=[-1 -2 -1; 0 0 0; 1 2 1];
%BW2 = edge(a,'Prewitt');
a=double(a);
for m=2:1:x-1
   for n=2:1:y-1
       r(m,n)=w(1)*a(m-1,n-1)+ w(2)*a(m,n-1)+w(3)*a(m+1,n-1)+w(4)*a(m-1,n)+w(5)*a(m,n)+w(6)*a(m+1,n)+w(7)*a(m-1,n+1)+w(8)*a(m,n+1)+w(9)*a(m+1,n+1);
       r1(m,n)=w1(1)*a(m-1,n-1)+ w1(2)*a(m,n-1)+w1(3)*a(m+1,n-1)+w1(4)*a(m-1,n)+w1(5)*a(m,n)+w1(6)*a(m+1,n)+w1(7)*a(m-1,n+1)+w1(8)*a(m,n+1)+w1(9)*a(m+1,n+1);
   end
end
z=r+r1;
%for m=1:1:x
 %   for n=1:1:y
subplot(1,4,1);imshow(uint8(a));
subplot(1,4,2);imshow(uint8(r));
subplot(1,4,3);imshow(uint8(r1));
subplot(1,4,4);imshow(uint8(z));
```



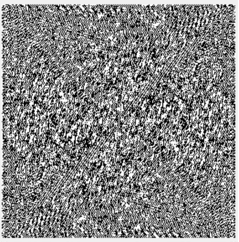
Important note about example:

% Program to Discrete Fourier Transform in the image: test with different image and report

Matlab Code

Result

```
%Discrete Fourier Transform
clc;
a=imread('cameraman.tif');
figure(1)
imshow(a)
a=double(a);
[row col]=size(a);
const=sqrt(row*col);
for n=0:1:row-1
   for k=0:1:col-1
       W(n+1,k+1)=exp(-i*2*pi*n*k/const);
   end
end
X=W*a*W.';
figure(2)
imshow(X)
ff=fft2(a)
aimg=ifft2(ff);
figure(3)
imshow(abs(mat2gray(aimg)))
```



Important note about example:

Practice eight

The lab number	M601	实验室名称	本院实验中心		
Course number		Subject title	**Digital Image Processing (MATLAB Programming)**		
The experiment item no	5	Practical title	**Image Segmentation**		
(To guide the file name)	(write)	(The experimental requirements)	(Will do)	(The experimental type)	(validation)
(period)					
(For professional)					

The purpose and requirement (fill in)

Purpose:

- Basic idea behind the practice is to learn practically about the image transformation because it's a common step mostly in every application (using image processing technique).
- Learn how to deal with the pixels, performing different operation.
- Understand how we can change intensity of an image.
- To get knowledge about the detail in an image (different slice contain).

Requirement:

- Each student must have resources(computer)
- Every student have installed MATLAB2014a version.

Every student have lecture slide and file which I send.

Content:

f) Hand region segmentation
g) Brain tumor segmentation
h) Face Recognition Project

%% Program to segment the brain tumor from MRI image

<table>
<tr><th>Matlab Code</th><th>Result</th></tr>
<tr><td>

```
  clear all;
close all;
% jpgFiles=dir('*.JPG');
%if wou want to get more picture at the same time you delete the comment '%'
% % for k=1:length(jpgFiles)
 %if wou want to get more picture at the same time you delete the comment '%'
%     Wajiha=k;
%if wou want to get more picture at the same time you delete the comment '%'
%     filename=jpgFiles(k).name;
%if wou want to get more picture at the same time you delete the comment '%'
%     I=imread(filename);
%if wou want to get more picture at the same time you delete the comment '%'
I=imread('12.jpg');
%if wou want to get more picture at the same time you comment it like'%'
I=imresize(I,[256 256]);
figure;imshow(I);
I=im2double(I);
[nrow ncol dim] = size(I);

if dim==3
    I = rgb2gray(I);
end
[r c]=size(I);
for x=1:r
    for y=1:c
        if I(x,y)>=0.7;
            M(x,y)=I(x,y)^(0.6);
        else
            M(x,y)=I(x,y)^2;
        end
    end
end
figure;imshow(M);
se = strel(ones(5,5));
e1 = imerode(M, se);
figure;imshow(e1);
nCluster =3;

[IDX,C,sumd,D] = kmeans(e1,nCluster);
%  sums of point-to-centroid distances in the 1-by-nCluster vector
center=sort(sumd);
Sc=size(sumd,1);

for x=1:Sc
if x<Sc
threshvalue(x) = (center(x)+center(x+1))/2;
end
end
if (Sc>2) & (Sc<4)
M = median(center);
L1=M*ones(nrow,ncol);
for irow=1:nrow
for icol=1:ncol
    for iCluster = 1:nCluster
        if ( e1(irow,icol) < threshvalue(Sc-2)/255 )
            L1(irow,icol)=center(Sc);
        end
        if ( e1(irow,icol) < threshvalue(Sc-1)/255 )
```

</td><td></td></tr>
</table>

```
            L1(irow,icol)=center(Sc-2);
        end
    end
end
end

else if Sc==2
        L1=center(Sc)*ones(nrow,ncol);
for irow=1:nrow
for icol=1:ncol
    for iCluster = 1:nCluster
        if ( e1(irow,icol) < threshvalue(Sc-1)/255 )
            L1(irow,icol)=center(Sc-1);
        end
    end
end
end
    end
end

 figure, imshow(L1,[]);

% end  %if wou want to get more picture at the same time you delete the comment '%'
```

Important note about example :

%% Program to segment the hand region from the image

Matlab Code	Result

```
clear all;
close all;
% jpgFiles=dir('*.JPG');
%if wou want to get more picture at the
same time you delete the comment '%'

% for k=1:length(jpgFiles)
%if wou want to get more picture at the
same time you delete the comment '%'

%       pic=k;
%if wou want to get more picture at the
same time you delete the comment '%'

%    filename=jpgFiles(k).name;
 %if wou want to get more picture at the
same time you delete the comment '%'

%
%  I=imread(filename);
```

```
%if wou want to get more picture at the
same time you delete the comment '%'

I=imread('65.jpg'); %if wou want to get more
picture at the same time you comment it
like'%'
I=imresize(I,[256 256]);
 figure;imshow(I);
[nrow ncol dim] = size(I);
cform = makecform('srgb2lab');
J = applycform(I,cform);
figure;imshow(J);
K=J(:,:,3);
figure;imshow(K);
L=graythresh(J(:,:,3));
BW1=im2bw(J(:,:,3),L);
figure;imshow(BW1);
[r c]=size(BW1);
figure;imshow(BW1);
for i=1:r
for j=1:c
    if BW1(i,j)>0
        M(i,j)=I(i,j);
    end
end
end
 figure;imshow(M);
 M=im2bw(M);
SE=[0 0 1 0 0;
    0 1 1 1 0;
    1 1 1 1 1;
    0 1 1 1 0;
    0 0 1 0 0];

IM2=imerode(M,SE);
SE1=[1 1 1 1 1 1;
    1 1 1 1 1 1;
    1 1 1 1 1 1;
    1 1 1 1 1 1;
    1 1 1 1 1 1;
    1 1 1 1 1 1];
IM2 = imdilate(IM2,SE1);

figure;imshow(IM2);
 IM2=im2bw(IM2);
 figure;imshow(IM2);
IM3=edge(IM2,'canny');

 IM3=imresize(IM3,[256 256]);
figure;imshow(IM3);
```

%end **%if you want to get more picture at the same time you delete the comment '%'**	
Important note about example :	

List of main function used in lecture 2021

1) main function used in Matlab

The image part with relationship ID rId56 was not found in the file.

2) Main Function used in Digital Image processing

The image part with relationship ID rId56 was not found in the file.

3) List of main function used in Matlab Digital image processing

List of main function used in DIP lecture 2020

No	Function Name	Function Aim	Function Parameter
1	imread	Read image from graphics file	Filename, format, value
2	imwrite	Write image to graphics file	Raw image, filename, mapping, format
3	imshow	Display image	Out of imread
4	clc	Clear command window	N/A
5	Close	Remove specified figure	Name, all, force, hidden
6	figure	Create figure window	Name, value
7	grayslice	Convert grayscale image to indexed image using multilevel thresholding	Image read from imread, threshold value
8	imhist	Histogram of image data	Image, mapping
9	im2bw	Convert image to binary image	Image
10	imsharpen	Sharpen image using unsharp masking	Image, name, value
11	rgb2gray	Converts RGB color spaced image to grayscale	Image
12	imadd	Add two images	Image1, image2
13	imabsdiff	Absolute difference between 2 images	Image1, image2
14	immultiply	Multiply 2 images	Image1, image2
15	imdivide	Divide one image by another	Image1, image2
16	Iminfo	Show info of image	image
17	nlfilter	General sliding-neighborhood operations	Grayscale image, [mxn] value, function
18	subplot	Create axes in tiled positions	M,n,p where mxn = grid size and p is position by which axes are created
19	im2double	Convert image to double precision	Image
20	imadjust	Adjust image intensity values or colormap	Image
21	graythresh	Global image threshold using Otsu's method	Image, level

22	imopen	Morphologically open image	Image, Strel()
23	Imsubtract	Subtract image from another	Image1, image2
24	imcomplement	Invert colors/complement the image	image
25	Bitand	Bitwise AND logic of image	Image1,image2
26	Bitor	Bitwise OR logic of image	Image1, image2
27	Bitxor	Bitwise XOR logic of image	Image1, image2
28	Bitcmp	Bitwise complement of image	Image1,image2
29	imhisteq	Enhance contrast using histogram equalization	Image, value
30	Imnoise	Add noise to image	Image, noise type, level
31	Imfilter	N-D filtering of multidimensional images	Image, filer mean value [matrix]
32	medfilt2	2-D median filtering	Image, filer mean value [matrix]
33	fspecial	Gaussian lowpass filter of size hsize with standard deviation sigma	Image, sigma
34	ordfilt2	replaces each element in A by the order the element in the sorted set of neighbours specified by the nonzero elements in domain .	Image, filer mean value [matrix]
35	tform	TFORM struct T for a two-dimensional affine transformation	Image , 2D, Axis
36	imadjust	Adjust image intensity values or colormap	Image
37	graythresh	Finding the threshold value	Image, level
38	sobel	detects edges in image	Image , levels
39	meshgrid	2-D grid coordinates based on the coordinates	Image , levels , variables
40	img_pow	Using matlab functions on the image matrix	Matrix, Axis
41	imdilate	Dilates the grayscale, binary, or packed binary image	Image , value
42	bwperim	Find perimeter of objects in binary image	Image , value
43	imerode	The imerode function determines the center element of the neighborhood	Image , text
44	imagesc	Display image with scaled colors	Image

45	bwmorph	Morphological operations on binary images	Image1, Image2
46	bwskel	This MATLAB function reduces all objects in the 2-D binary image A to 1-pixel wide curved lines	Image, Skeletonize Binary Image
47	imtophat	performs morphological top-hat filtering	Image , binary image, filtered image
48	imadjust	Adjust image intensity values or colormap	Image1, Image2
49	imcrop	Crop Image tool associated with the grayscale	Image
50	bwhitmiss	performs the hit-miss operation defined in terms of a single array	Array , Image
51	imgaussfilt	This MATLAB function filters image A with a 2-D Gaussian smoothing kernel	Value , Image1 ,Image2, Frequency
52	imfftlog	frequency usually comes out in linear scale from Discrete Fourier Transform.	Text, Image, Level
53	fft2	frequency usually comes out in linear scale from Discrete Fourier Transform.	Text, Image, Level
54	fftshift	This MATLAB function rearranges a Fourier transform X by shifting the zero-frequency	Text, Image, Level
55	applycform	This MATLAB function converts the colour values in A to the colour space specified in the colour transformation	Image1 , Image2
56	imLab	graphical application for Scientific Image	Image , scientific image
57	imRGB	RGB image to grayscale	Binary Image, Image
58	FDetect	Face detection	Image
59	BBsize	Bounding Box values based on number of objects	Image detect
60	IEzc	Finding the threshold value	Image
61	Imaqhwinfo	Know about device info	Image, level
62	fscanf	Read formatted data from a file.	Level, Text, Matrix
63	fprintf	Performs formatted writes to screen or file.	Level, Text, Matrix
64	findstr	Finds occurrences of a string.	Image, mapping

65	strcmp	Compares strings.	Image, mapping
66	ezplot	Generates a plot of a symbolic expression.	Image, mapping
67	imhist	Display a histrogram	Image, mapping
68	histeq	Equalize image	Image, mapping
69	graythresh	Finding the threshold value	Image, level
70	Imsharpen	Image sharpening	Image, name, value
71	clc	Clear command window	N/A
72	Close	Remove specified figure	Name, all, force, hidden