江西理工大學信息工程学院

JIANGXI UNIVERSITY OF SCIENCE AND TECHNOLOGY SCHOOL OF INFORMATION ENGINEERING

Mohammad ASIM , 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

```
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
```

2. MATLAB VARIABLES

```
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
```

3. MATLAB - MATRIX

```
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)
```

4. MATLAB – MATRIX: create a sub-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];
sa = a(2:3,2:4)
```

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(愠:\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

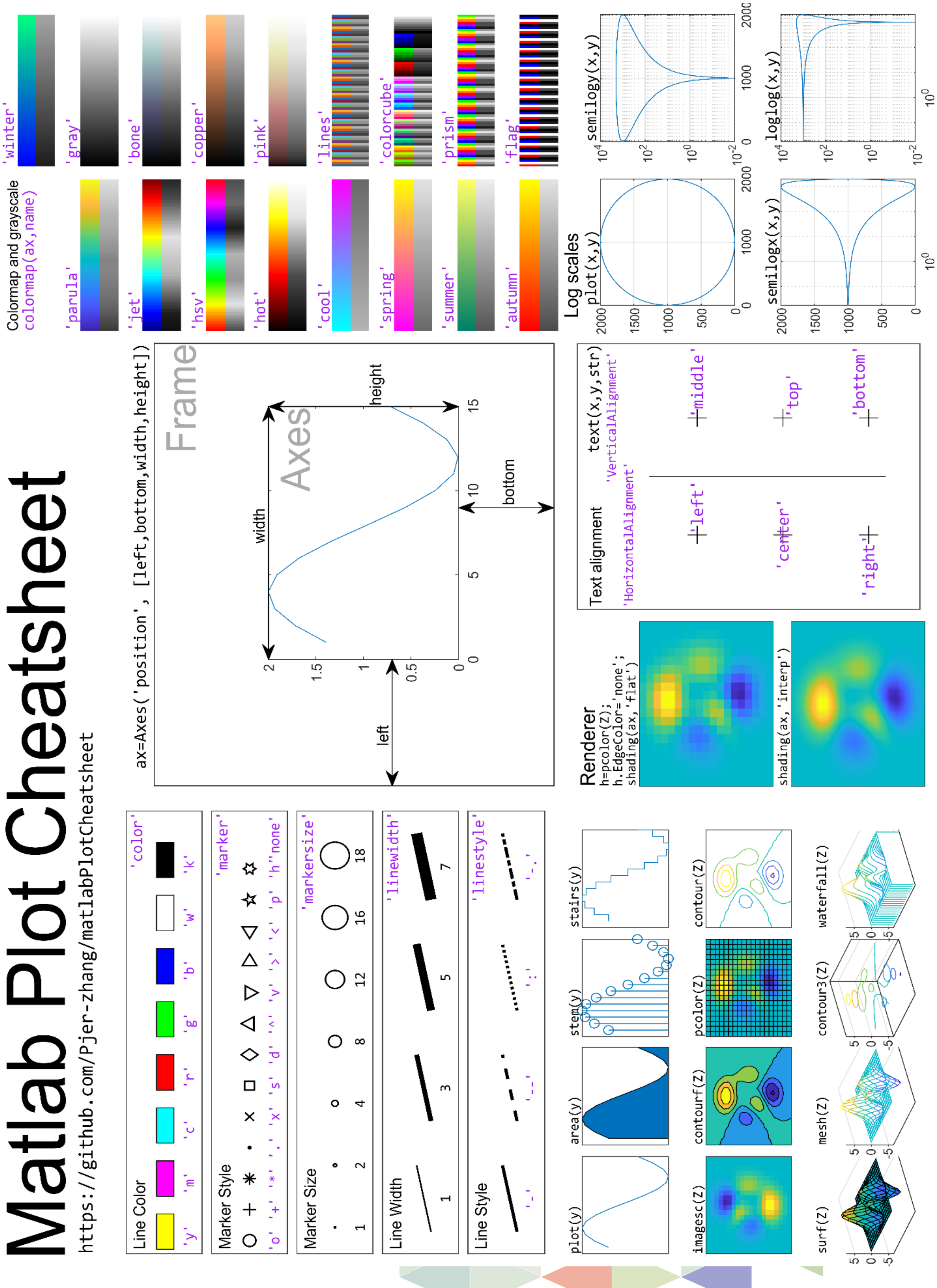
```
ax_txt_posi = axes('position',[0.52,0.015,0.21,0.36],'visible',vis_ax);
text(1,0.98,'text(x,y,str)','HorizontalAlignment','right','VerticalAlignment','top','FontSize',ftsz*12,'FontName','consolas','color','k')
text(0.01,0.98,'Text alignment','VerticalAlignment','top','FontSize',ftsz*12,'color','k')
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

```
ax=Axes('position', [left,bottom,width,height])
colormap(ax,name)
text(x,y,str)
h=pcolor(Z);
h.EdgeColor='none';
shading(ax,'flat')
shading(ax,'interp')
```

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	4	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:

a) Hand region segmentation
b) Brain tumor segmentation
c) Face Recognition Project

1. %% Program to segment the brain tumor from MRI image

```
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 )
         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 '%'
```

2. %% Program to segment the hand region from the image

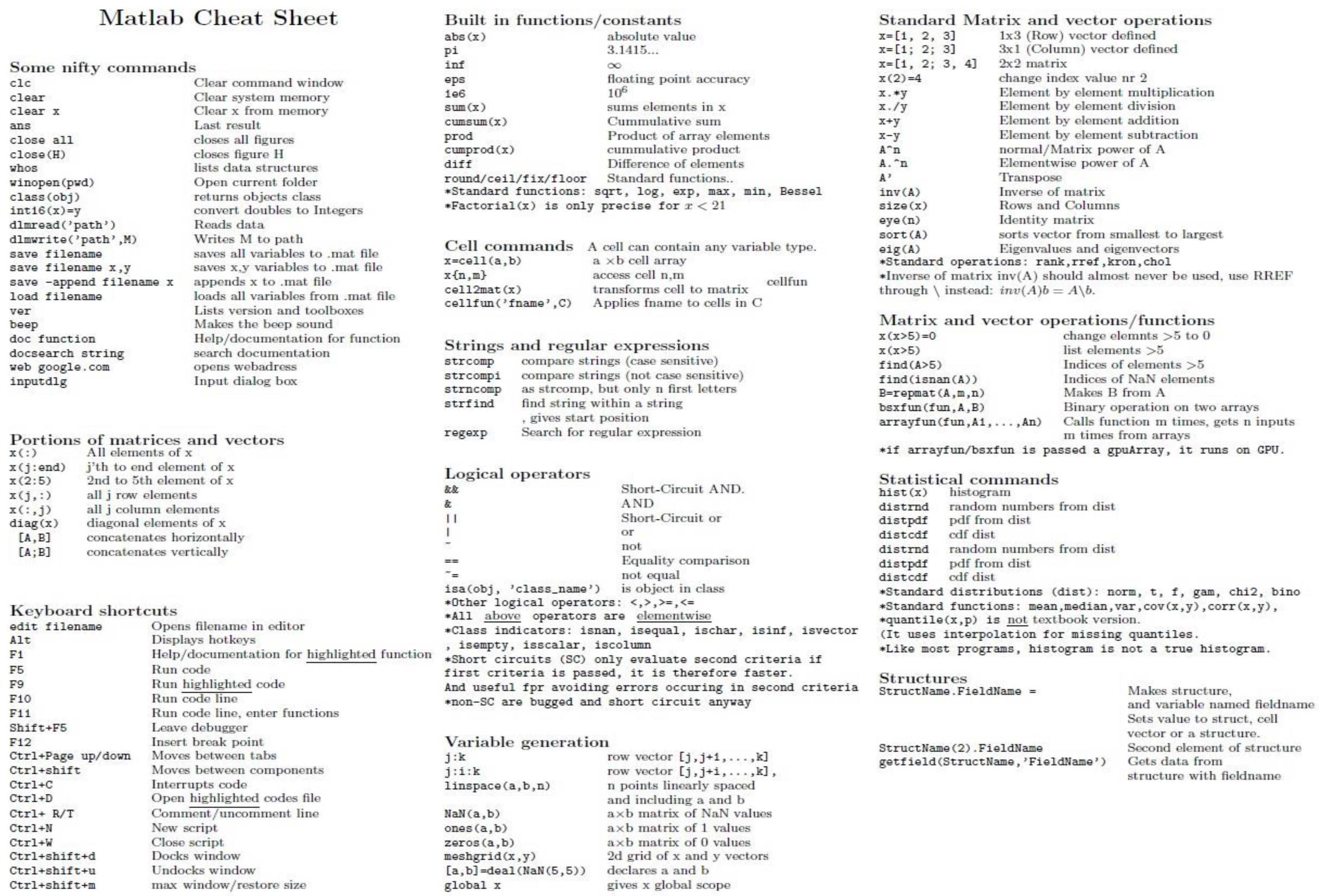
```
clear all;
close all;
% jpgFiles=dir('*.JPG');
%if wou want to get more picture at the same time you delete the comment '%'

% 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 '%'**

List of main function used in lecture 2021

1) main function used in Matlab



Matlab Cheat Sheet

Some nifty commands

`clc`	Clear command window
`clear`	Clear system memory
`clear x`	Clear x from memory
`ans`	Last result
`close all`	closes all figures
`close(H)`	closes figure H
`whos`	lists data structures
`winopen(pwd)`	Open current folder
`class(obj)`	returns objects class
`int16(x)=y`	convert doubles to Integers
`dlmread('path')`	Reads data
`dlmwrite('path',M)`	Writes M to path
`save filename`	saves all variables to .mat file
`save filename x,y`	saves x,y variables to .mat file
`save -append filename x`	appends x to .mat file
`load filename`	loads all variables from .mat file
`ver`	Lists version and toolboxes
`beep`	Makes the beep sound
`doc function`	Help/documentation for function
`docsearch string`	search documentation
`web google.com`	opens webadress
`inputdlg`	Input dialog box

Portions of matrices and vectors

`x(:)`	All elements of x
`x(j:end)`	j'th to end element of x
`x(2:5)`	2nd to 5th element of x
`x(j,:)`	all j row elements
`x(:,j)`	all j column elements
`diag(x)`	diagonal elements of x
`[A,B]`	concatenates horizontally
`[A;B]`	concatenates vertically

Keyboard shortcuts

`edit filename`	Opens filename in editor
`Alt`	Displays hotkeys
`F1`	Help/documentation for highlighted function
`F5`	Run code
`F9`	Run highlighted code
`F10`	Run code line
`F11`	Run code line, enter functions
`Shift+F5`	Leave debugger
`F12`	Insert break point
`Ctrl+Page up/down`	Moves between tabs
`Ctrl+shift`	Moves between components
`Ctrl+C`	Interrupts code
`Ctrl+D`	Open highlighted codes file
`Ctrl+ R/T`	Comment/uncomment line
`Ctrl+N`	New script
`Ctrl+W`	Close script
`Ctrl+shift+d`	Docks window
`Ctrl+shift+u`	Undocks window
`Ctrl+shift+m`	max window/restore size

Built in functions/constants

`abs(x)`	absolute value
`pi`	3.1415...
`inf`	∞
`eps`	floating point accuracy
`1e6`	10^6
`sum(x)`	sums elements in x
`cumsum(x)`	Cummulative sum
`prod`	Product of array elements
`cumprod(x)`	cummulative product
`diff`	Difference of elements
`round/ceil/fix/floor`	Standard functions..

*Standard functions: `sqrt, log, exp, max, min, Bessel`
*Factorial(x) is only precise for $x < 21$

Cell commands A cell can contain any variable type.

`x=cell(a,b)`	a ×b cell array	
`x{n,m}`	access cell n,m	cellfun
`cell2mat(x)`	transforms cell to matrix	
`cellfun('fname',C)`	Applies fname to cells in C	

Strings and regular expressions

`strcomp`	compare strings (case sensitive)
`strcompi`	compare strings (not case sensitive)
`strncomp`	as strcomp, but only n first letters
`strfind`	find string within a string , gives start position
`regexp`	Search for regular expression

Logical operators

`&&`	Short-Circuit AND.
`&`	AND
`||`	Short-Circuit or
`|`	or
`~`	not
`==`	Equality comparison
`~=`	not equal
`isa(obj, 'class_name')`	is object in class

*Other logical operators: `<,>,>=,<=`
*All above operators are elementwise
*Class indicators: `isnan, isequal, ischar, isinf, isvector, isempty, isscalar, iscolumn`
*Short circuits (SC) only evaluate second criteria if first criteria is passed, it is therefore faster. And useful fpr avoiding errors occuring in second criteria
*non-SC are bugged and short circuit anyway

Variable generation

`j:k`	row vector [j,j+1,...,k]
`j:i:k`	row vector [j,j+i,...,k],
`linspace(a,b,n)`	n points linearly spaced and including a and b
`NaN(a,b)`	a×b matrix of NaN values
`ones(a,b)`	a×b matrix of 1 values
`zeros(a,b)`	a×b matrix of 0 values
`meshgrid(x,y)`	2d grid of x and y vectors
`[a,b]=deal(NaN(5,5))`	declares a and b
`global x`	gives x global scope

Standard Matrix and vector operations

`x=[1, 2, 3]`	1x3 (Row) vector defined
`x=[1; 2; 3]`	3x1 (Column) vector defined
`x=[1, 2; 3, 4]`	2x2 matrix
`x(2)=4`	change index value nr 2
`x.*y`	Element by element multiplication
`x./y`	Element by element division
`x+y`	Element by element addition
`x-y`	Element by element subtraction
`A^n`	normal/Matrix power of A
`A.^n`	Elementwise power of A
`A'`	Transpose
`inv(A)`	Inverse of matrix
`size(x)`	Rows and Columns
`eye(n)`	Identity matrix
`sort(A)`	sorts vector from smallest to largest
`eig(A)`	Eigenvalues and eigenvectors

*Standard operations: `rank,rref,kron,chol`
*Inverse of matrix inv(A) should almost never be used, use RREF through \ instead: $inv(A)b = A\backslash b$.

Matrix and vector operations/functions

`x(x>5)=0`	change elemnts >5 to 0
`x(x>5)`	list elements >5
`find(A>5)`	Indices of elements >5
`find(isnan(A))`	Indices of NaN elements
`B=repmat(A,m,n)`	Makes B from A
`bsxfun(fun,A,B)`	Binary operation on two arrays
`arrayfun(fun,A1,...,An)`	Calls function m times, gets n inputs m times from arrays

*if arrayfun/bsxfun is passed a gpuArray, it runs on GPU.

Statistical commands

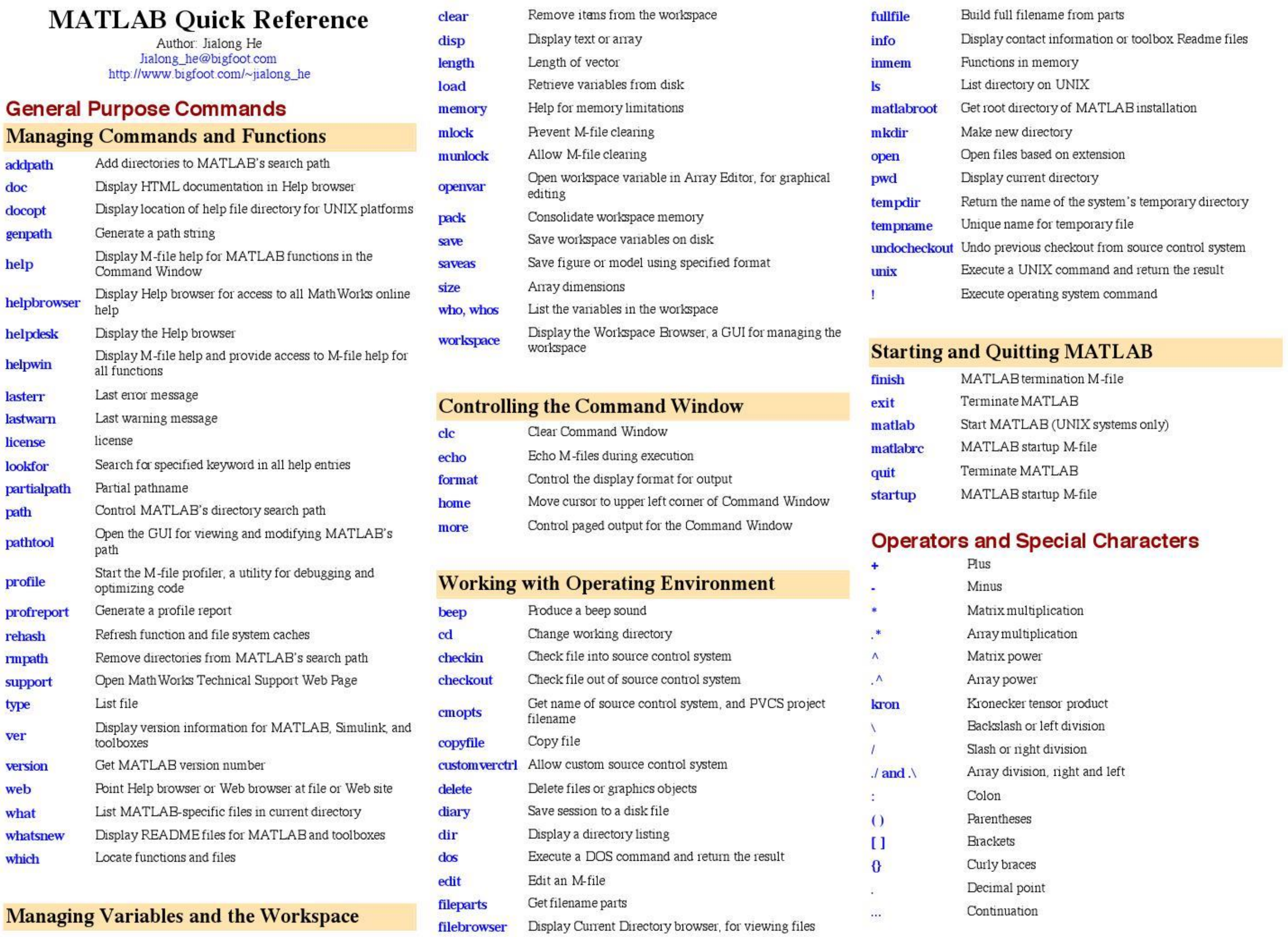
`hist(x)`	histogram
`distrnd`	random numbers from dist
`distpdf`	pdf from dist
`distcdf`	cdf dist
`distrnd`	random numbers from dist
`distpdf`	pdf from dist
`distcdf`	cdf dist

*Standard distributions (dist): `norm, t, f, gam, chi2, bino`
*Standard functions: `mean,median,var,cov(x,y),corr(x,y),`
*quantile(x,p) is not textbook version.
(It uses interpolation for missing quantiles.
*Like most programs, histogram is not a true histogram.

Structures

`StructName.FieldName =`	Makes structure, and variable named fieldname Sets value to struct, cell vector or a structure.
`StructName(2).FieldName`	Second element of structure
`getfield(StructName,'FieldName')`	Gets data from structure with fieldname

2) Main Function used in Digital Image processing



MATLAB Quick Reference

Author: Jialong He
Jialong_he@bigfoot.com
http://www.bigfoot.com/~jialong_he

General Purpose Commands

Managing Commands and Functions

addpath	Add directories to MATLAB's search path
doc	Display HTML documentation in Help browser
docopt	Display location of help file directory for UNIX platforms
genpath	Generate a path string
help	Display M-file help for MATLAB functions in the Command Window
helpbrowser	Display Help browser for access to all MathWorks online help
helpdesk	Display the Help browser
helpwin	Display M-file help and provide access to M-file help for all functions
lasterr	Last error message
lastwarn	Last warning message
license	license
lookfor	Search for specified keyword in all help entries
partialpath	Partial pathname
path	Control MATLAB's directory search path
pathtool	Open the GUI for viewing and modifying MATLAB's path
profile	Start the M-file profiler, a utility for debugging and optimizing code
profreport	Generate a profile report
rehash	Refresh function and file system caches
rmpath	Remove directories from MATLAB's search path
support	Open MathWorks Technical Support Web Page
type	List file
ver	Display version information for MATLAB, Simulink, and toolboxes
version	Get MATLAB version number
web	Point Help browser or Web browser at file or Web site
what	List MATLAB-specific files in current directory
whatsnew	Display README files for MATLAB and toolboxes
which	Locate functions and files

Managing Variables and the Workspace

clear	Remove items from the workspace
disp	Display text or array
length	Length of vector
load	Retrieve variables from disk
memory	Help for memory limitations
mlock	Prevent M-file clearing
munlock	Allow M-file clearing
openvar	Open workspace variable in Array Editor, for graphical editing
pack	Consolidate workspace memory
save	Save workspace variables on disk
saveas	Save figure or model using specified format
size	Array dimensions
who, whos	List the variables in the workspace
workspace	Display the Workspace Browser, a GUI for managing the workspace

Controlling the Command Window

clc	Clear Command Window
echo	Echo M-files during execution
format	Control the display format for output
home	Move cursor to upper left corner of Command Window
more	Control paged output for the Command Window

Working with Operating Environment

beep	Produce a beep sound
cd	Change working directory
checkin	Check file into source control system
checkout	Check file out of source control system
cmopts	Get name of source control system, and PVCS project filename
copyfile	Copy file
customverctrl	Allow custom source control system
delete	Delete files or graphics objects
diary	Save session to a disk file
dir	Display a directory listing
dos	Execute a DOS command and return the result
edit	Edit an M-file
fileparts	Get filename parts
filebrowser	Display Current Directory browser, for viewing files
fullfile	Build full filename from parts
info	Display contact information or toolbox Readme files
inmem	Functions in memory
ls	List directory on UNIX
matlabroot	Get root directory of MATLAB installation
mkdir	Make new directory
open	Open files based on extension
pwd	Display current directory
tempdir	Return the name of the system's temporary directory
tempname	Unique name for temporary file
undocheckout	Undo previous checkout from source control system
unix	Execute a UNIX command and return the result
!	Execute operating system command

Starting and Quitting MATLAB

finish	MATLAB termination M-file
exit	Terminate MATLAB
matlab	Start MATLAB (UNIX systems only)
matlabrc	MATLAB startup M-file
quit	Terminate MATLAB
startup	MATLAB startup M-file

Operators and Special Characters

+	Plus
-	Minus
*	Matrix multiplication
.*	Array multiplication
^	Matrix power
.^	Array power
kron	Kronecker tensor product
\	Backslash or left division
/	Slash or right division
./ and .\	Array division, right and left
:	Colon
( )	Parentheses
[ ]	Brackets
{}	Curly braces
.	Decimal point
...	Continuation

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