Lab EXE 2

1. Some Matlab basic coding

1.1 Adding each column vector of a matrix $\boldsymbol{A}$ with a vector $\boldsymbol{b}$ (Of course, the numbera of rows in $\boldsymbol{A}$ and $\boldsymbol{b}$ should be equal)

In Matlab, you can use

$$\boldsymbol{AA} = \boldsymbol{A} + \boldsymbol{b}$$

Example code:

```
>> A=[1 2 3 4 5 6;2 3 4 5 6 7]

A =

     1     2     3     4     5     6
     2     3     4     5     6     7

>> b=[0.5 0.5]'

b =

    0.5000
    0.5000

>> AA=A+b

AA =

    1.5000    2.5000    3.5000    4.5000    5.5000    6.5000
    2.5000    3.5000    4.5000    5.5000    6.5000    7.5000
```

1.2 Multiply each element of a matrix $\boldsymbol{A}$ with a constant scalar c

In Matlab, you can use

$$\boldsymbol{AA} = c * \boldsymbol{A}$$

Example code:

```
>> A=[1 2 3 4 5 6;2 3 4 5 6 7]

A =

     1     2     3     4     5     6
     2     3     4     5     6     7

>> c=0.1

c =

    0.1000

>> AA=c*A

AA =

    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000
    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000
```

1.3 Multiply each column vector of a matrix ***A*** with another matrix ***T*** (of course the number of columns of ***A*** should be equal to the number of rows of ***T*).**

In Matlab, you can use

$$\boldsymbol{AA = T * A}$$

Example code:

```
>> A=[1 2 3 4 5 6;2 3 4 5 6 7]

A =

     1     2     3     4     5     6
     2     3     4     5     6     7

>> T=[0 1;1 0]

T =

     0     1
     1     0

>> AA=T*A

AA =

     2     3     4     5     6     7
     1     2     3     4     5     6
```

1.4

Define a rotation matrix

```
theta=2*pi/2;
R=[cos(theta), -sin(theta);sin(theta) cos(theta)];
```

1.5 Define array for with fixed integer increments

```
N =

   200

>> ar=[0:1:N-1];
>> ar(1:5)

ans =

     0     1     2     3     4
```

Get the i-th element of ar

```
>> i=2

i =

     2

>> ar(i)

ans =

     1
```

```
>> i=4

i =

     4

>> ar(i)

ans =

     3
```

1.6 Define an array for angles from 0 to 2π

```
>> N=200;
>> angle=2*pi*[0:1:N-1]/N;
>> angle(1:8)

ans =

         0    0.0314    0.0628    0.0942    0.1257    0.1571    0.1885    0.2199
```

1.6 Matlab for loop. `for` loop to repeat specified number of times

Syntax

```
for index = values
   statements
end
```

Example

```
>> sum=[0 0]'

sum =

     0
     0

>> for i=1:2
b=[1*i 2*i]'
sum=sum+b
end

b =

     1
     2


sum =

     1
     2


b =

     2
     4


sum =

     3
     6
```

2. Introduction

The aim of this laboratory is to let you to understand the basic concept of geometric transform. First at all, you should review your notes about vector displacement (chapter 2) and rotation matrix (chapter 3). We will use a simple matlab code to visualize our result.

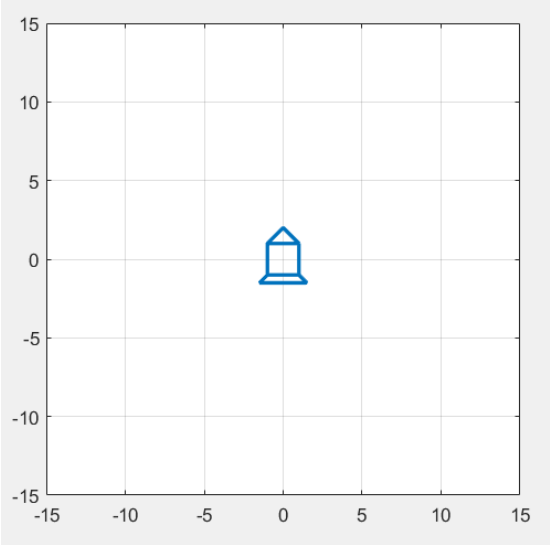


Figure 1 Display an object in the MatLab figure window.

In computer graphics, there are two concepts to represent images. They are raster graphics and vector graphics. The raster graphics concept directly stores the pixel values of the image. For example, if the image resolution is 256×256 , the raster graphics concept stores all the 65,536 pixels.

The vector graphics concept does not directly store an image. It uses a number of control points to represent an object. For example, the object, shown in Figure 1, can be represented by a 2×11 matrix (11 2D control points), given by

data matrix =

```
[1    1  -1  -1  1  0 -1 -1 -1.5   1.5  1;
 1   -1  -1   1  1  2  1 -1 -1.5  -1.5 -1];
```

Each column vector of the matrix defines a control point. To draw the object, the machine first gets the initial position by picking up the first control point. It then draws a line from the first control point to the second point, a line from the second control point to the third control point, and so on.

Question 1
Discuss main differences between raster graphics and vector graphics. (around than 100 -150 words) Hint: use google to study but use your own words.

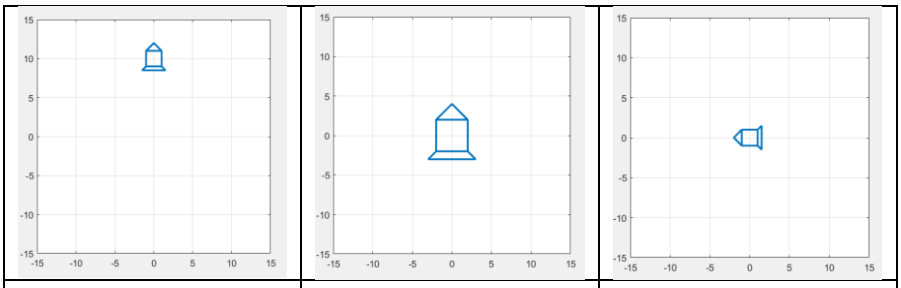
Table 1 shows a basic MatLab program to draw an object in the Matlab figure window. The program has sufficient comments to describe the way to program.

Table 1. Sample code

```
% define control points of the object
x=[1    1  -1  -1  1  0 -1 -1 -1.5   1.5  1;
   1   -1  -1   1  1  2  1 -1 -1.5  -1.5 -1];
N=200
for i=1:N
% you can define some operation to modify the object, such
as
% scaling the object, rotation an object, tanslate an
object.
% x: each column vetor is an originsl position vector
% xx: each column vector is a resultant vector
% if you want to produce an animation, you need to modify
% some values at each iteration.
% Modify the code (rather than using xx=x)

    xx=x;
% draw the object
% Plot is to draw the object with line width equal to 2
% axis is to define the display range
% daspect is to define aspect ratio to 1:1
% grid on is to grid lines in the display window
% draw now is to tell draw the object now and delay the a
while
plot(xx(1,:),xx(2,:),'Linewidth',2);
axis([-15, 15, -15, 15]);
daspect([1 1 1])
grid on
drawnow
end
```

3. The experiment steps:



(a) (b) (c)

1. Design an algebra operation to move the object from $\begin{pmatrix}0\\0\end{pmatrix}$ to $\begin{pmatrix}0\\10\end{pmatrix}$. (Figure 2(a).)

2. Write a matlab program to produce an animation for the movement (see video1 in canvas).

3. Design an algebra operation to magnify the object (Figure 2(b).)

4. Write a matlab program to produce an animation for magnification (see video2 in canvas).

5. Design an algebra operation to rotate the object (Figure 2(c).)

6. Write a matlab program to produce an animation for the rotation (see video3 in canvas).

7. Write a matlab program to produce an animation, in which the object moves along an circular orbit with radius equal to 10. (see video4 in canvas)

   We can generate a set of sample points on 2D on the circular orbit.

```
N=200;
angle=2*pi*[0:1:N-1]/N;
x1=10*cos(angle)
x2=10*sin(angle)
```

Question 2

In computer graphics, we usually define the coordinates in a homogenous form.
For the 2D case, the homogenous coordinates is a 3-vector given by

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}.$$

With this formulation, all the geometric transforms can be written as

$$\boldsymbol{x}' = \boldsymbol{T}\boldsymbol{x},$$

where $\boldsymbol{T}$ is a 3×3 matrix. For different geometric transforms, we have different $\boldsymbol{T}$.

Use Internet to study the various homogenous geometric transforms, including translation, scaling, rotation, and reflection. Use one page to present your findings.

3. Report

1. Answer all the questions. You should use your own words rather than directly copying the text from Internet.
2. Perform all the steps in Section 2. Present and explain the algebra operations.
3. Include the Matlab Code.