EENG 428 Introduction to Robotics Laboratory

EXPERIMENT 1

Introduction to MATLAB

Objectives:

This experiment aims on familiarizing students with the basic usages of MATLAB, including; m-files, basic MATLAB commands, MATLAB variables and functions as well as plotting and labeling functions via MATLAB.

1. Introduction

MATLAB is a high-level programming language and leading problem-solving environment. Basically, it can be efficiently used to solve diverse engineering and mathematical problems. As the name suggests, MATLAB is based on dealing with problems in the matrix form.

The MATLAB environment contains three window types to work with. These are the command window, the Figure window and the Editor window. The Figure window only pops up whenever you plot something. The Editor window is used for writing and editing MATLAB programs (called M-files) and can be invoked in Windows from the pull down menu after selecting **File → New → M-file.**

The command window is the main window in which you communicate with the MATLAB interpreter. The MATLAB interpreter displays a command (>>) indicating that it is ready to accept commands from you.

2. Arithmetic Operations

There are four basic arithmetic operators:

+ Addition

− Subtraction

* Multiplication

/ Division (for matrices it also means inversion)

For arrays (or vectors), there are also three other operators that operate on an element-by-element basis:

.* Multiplication of two vectors, element by element.

./ Division of two vectors, element-wise.

.^ Raise all the elements of a vector to a power.

<u>Exampe 1:</u>

```
>> X=[1,3,4]

>> Y=[4,5,6]

>> X+Y

ans= 5 8 10
```

For the vectors X and Y the operator (+) adds the elements of the vectors, one by one, assuming that the two vectors have the same dimension. In the above example, both vectors had the dimension 1×3.

As an example, to compute the inner (dot) product of two vectors, you can use the multiplication operator *. For the above example, it is:

```
>> X*Y'

ans = 43
```

Note the single quote after Y. The single quote denotes the transpose of a matrix or a vector. To compute an element by element multiplication of two vectors (or two arrays), you can use the **.*** operator:

```
>> X.*Y

ans = 4 15 24
```

That is, X.*Y means [1×4, 3×5, 4×6] = [4 15 24]. The '.*' operator is used very often (and is highly recommended) because it is executed much faster compared to the code that uses for loops.

3. For Loops

The loop concept is developed to handle repetitive procedures, i.e., it is used with processes that are to be repeated for a specific number of times. In each loop, there will be a change in the inputs or the outputs of the process.

Syntax

```
Loop counter incremented by one:
for i = startValue:endValue
x = ...
y = ...
.
.
.
end
```

i is the loop counter

<u>Example2:</u> Compute the sum of the first 10 integers.

```
n = 10;

s = 0;

for i=1:n

s = s + i;
```

3. Platting and Labeling Functions

MATLAB can be used to plot array values versus the index vector, or time functions. The command *plot(X,Y)* plots vector Y versus vector X. Various line types, plot symbols and colors may be obtained using *plot(X,Y,S)* where *S* is a character string indicating the color of the line, and the type of line (e.g., dashed, solid, dotted, etc.). Examples for the string *S* include:

r	Red	+	Plus	--	Dashed
g	Green	*	Star		
b	Blue	s	Square		

To plot a time function, one needs to firstly define the time interval along which the function is defined, as a vector. Next, function values at these time values are evaluated and arranged as another array. After that, the ***plot*** command is used to plot the function vector over the time vector.

To label the x-axis, we use the command **xlabel(“”),** writing the desired text inside the quotations. The same thing holds for labeling the y-axis and also putting a title for the graph, with the commands ***ylabel(“”)*** and ***title(“”),*** respectively.

4. MATLAB Functions:

A function can be defined is an m-file according to the following syntax:

function [out1, out2, ...] = funname(in1, in2, ...)

, where:

out1, *out2*, ..., are the function outputs, *in1, in2*, ... are its inputs and *funname* is the function name.

, then, the function can be called in the command window or in other m-files.

<u>Example 3:</u>

Define a function (*myfunction*) that accepts an array of numbers *x*, and calculates the sum and mean of the array numbers.

```
function [sum, mean]=myfunction(x)
size=length(x);
sum=0;
```

```
    for i=1:size
    sum=sum+x(i);
    end
mean=sum/size;
```

Example 4:

Plot the function $y = 50\sin(20\pi t + 10)$, with a green color, over the interval $t\epsilon[0:10]$, with a step of 0.1. Besides, label the x-axis and they-axis as (Time) and (Value), respectively. Also, title the graph as (Sin Function)

```
t=1:0.01:11;
y=50*sin(2*pi*(t-1)+10);
plot(t,y)
title('Sin Function')
xlabel('Time')
ylabel('Value')
grid
```

5. Homework

Considering the following questions, provide your answers in a report including your codes and associated outputs and/or plots by copying them and pasting on the report pages. To be submitted as a hard copy on next lab session. You should work **individually** and group work will be penalized.

1. Write a MATLAB m-file that accepts the two arrays:
   $$x = [1\ 2\ 3\ 4\ 5\ 6]$$
   $$y = [2\ 4\ 6\ 8\ 10\ 12]$$
   Then calls a function (calculate), that accepts the two arrays as inputs, and then calculates their inner product and returns it as an output.
2. Write a MATLAB program that defines and plots :
   $y = 50\sin(20\pi t + 10) + 10\cos(1020\pi t)$, with a red color, and a star-type line over the interval $t\epsilon[0:10]$, with a step of 0.1. Besides, label the x-axis and the y-axis as (Time) and (Value), respectively. Also, title the graph as (Sine Function).
3. Define a MATLAB function that evaluates the sum of the first n integers.