EENG 428 Laboratory --- Lab Session 2

Description:

This session is divided into two sections:

1- The usage of symbolic variables, and solving systems of equations.

2- Exploring the usage of symbolic Matlab to simplify robotic analysis.

Prerequisites:

Attending students are expected to know:

- Different forms of transformation matrices (Rotation about / Translation along an axis)
- Basic Matlab commands.

Contents:

1- Introduction to symbolic variables, expressions and symbolic matrices.

2- Introduction to solving systems of equations using Matlab.

3- Basic transformation matrices using Matlab.

4- Introduction to position analysis.

1- Introduction to symbolic variables and symbolic matrices:

Usually, mathematical formulas are derived using symbols. It would be very useful if a computer helps in the derivation and/or the simplification of symbolic formulas rather than only calculating numbers.

In the following, we present how to define symbolic variables, use them in a mathematical derivation, simplify our answers, get a geometrical view for our results and moreover, substitute numerical values when we need to.

1.1- Symbolic variables and expressions:

- Defining the Symbolics: Using Matlab, the function *syms()* allows us to define symbolic variables, and the function *sym()* allows us to define symbolic expressions.

Example:

```
syms l m n o p %if we want to use symbols in different
expressions
f= sym('a*x^2+b*x+c') % directly writing a symbolic expression
l=1 % we can give a value of any symbolic variable at any time
p=m*n % we can combine predefined symbolics into a new
expression
```

- Substituting Numbers: After defining an expression, we can substitute values instead of some symbols using the command *subs()*

Example:

```
syms f a b c x
f= a*x^2+b*x+c;
subs(f,[a b],[1 2])
f
```

You should be careful to save your result, notice that *f* have not been modified yet.

```
syms f a b c x
f= a*x^2+b*x+c;
f=subs(f,[a b],[1 2])
```

This time, f has been modified (*a* was substituted by 1 and *b* was substituted by 2).

- Plotting Symbolic Formulas: Usually, a functions can be plotted verses its independent variable within some range by hand.
  Matlab symbolic toolbox offers a similar service using the function *ezplot()*.

Example:

```
syms x y a b c
y=a*x^2+b*x+c;
y=subs(y,[a b c],[1 2 3]);
ezplot(y)
```

- Making the function to be more similar to human handwriting: Usually, humans do not express formulas as computers do. The function *pretty()* helps us to see the formulas as we are used to on paper.

Example:

```
syms x y a b c
y=[(a*x^2+b*x+c)/(b^2+c^2)]^2;
pretty(y)
```

- Symbolic Matrices: In this course, we are highly interested in working with symbolic matrices.

  There are two ways to define a symbolic matrix:

  1- Define each symbol independently, them merge them into a matrix:

Example:

```
syms a b c d e f g h i j
A=[a b c; d e f; g h i]
```

  2- Define the matrix directly using its size, and a symbol to be used in all entries combined with their location.

Example:

```
A=sym('a',[2 3])
B=sym('c',[3 2])
```

Exercise 1 (Homework): Write a Matlab code that:

- Defines two symbolic 4x5 matrices A and B.
- Then adds the first row of A to the first column of B (store the result in B),
- Finally finds 3A+7B.

2- Introduction to solving systems of equations using Matlab:

Matlab can help solving a single or a set of equations, the function *solve()* with the correct input parameters returns the set of structured solution.

(type *help solve* in Matlab command window for all details).

1- Solving a single equation:

If we have one equation with one variable, we only have to let the equation to be in the form (*equation = 0*), then we can give the *equation* part as input for the function *solve()*.

Example:

Find *x* in terms of the other constants for the common quadratic equation:

$$ax^2 + bx + c = 0$$

```
solve('a*x^2+b*x+c')
```

Notice that, the function automatically recognize *x* as the variable.

In case the variable name is not x or y , we have to specify it as an input variable.

Example:

Find a in terms of the other constants in the equation:

$$ax^2 + bx + c = 0$$

```
solve('a*x^2+b*x+c','a')
```

Exercise 2 (Homework):

1- Write a Matlab code that gives the solution for the equation:

$$xln(x) + e^{x^2} = 25$$

2- Verify using one of the iterative formulas (MATH252), by writing the code for the iterative scheme with excepted error < 10^(-6)

2- Solving a set of algebraic equations:

Solve the following set of equations:

$$\begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 6 \\ 2 & -1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 4 \\ 18 \\ 3 \end{bmatrix}$$

```
f =solve('x+y+z=4','2*x+4*y+6*z=18','2*x-y+z=3')
```

Notice that the answer is a structure, and the result is not shown yet.

Simply, follow the following command to find the final vector of solutions.

```
solution=[f.x;f.y;f.z]
```

Exercise 3 (Homework):

1- Find all the sets of solutions for the following set of non-linear equations:

$$n_1{}^2 + n_3{}^2 + 0.5 = 1$$
$$o_2{}^2 + o_3{}^2 = 1$$
$$a_1{}^2 + a_2{}^2 = 1$$
$$(1/\sqrt{2})\, o\_2 + n\_3\, o\_3 = 0$$
$$(1/\sqrt{2})\, a\_2 + a\_1\, n\_1 = 0$$
$$a_2 o_2 = 0$$

2- Form a matrix in which each of its columns contains one set of solutions.

3- Basic transformation matrices using Matlab:

Matlab robotic toolbox offers some helpful transformation matrices that can facilitate different applications of robotic analysis.

The basic transformation matrices are:

1- Rotation about x-axis. (Matlab function *trotx()*)
2- Rotation about y-axis. (Matlab function *troty()*)
3- Rotation about z-axis. (Matlab function *trotz()*)
4- Translation about x,y and z axes. (Matlab function *trnasl()*)

Examples:

```
syms t
trotx(t)% rotate about x-axis with angle t
troty(t)% rotate about y-axis with angle t
trotz(t)% rotate about z-axis with angle t
syms px py pz
transl(px,py,pz)% rotate along x,y and z axes with
distance px, py, and pz respectively
```

4- Introduction to position analysis:

Given two coordinate frames *A* and *B* and the transformation matrix between these coordinates. The problem is to find the new relation between these coordinates after a sequence of transformations w.r.t the frame *A* and/or *B*.

Illustration:

If *A* and *B* are two coordinate frames, and ${}^{A}T_{B}$ is the location and orientation of *B* w.r.t *A*:

- If a translation/rotation happens w.r.t the frame *A* , the new location and orientation $({}^{A}T_{B})^{*}$ is given by the multiplication of the matrix representing the transformation by ${}^{A}T_{B}$ (Pre-multiplication)
- If a translation/rotation happens w.r.t the frame *B* , the new location and orientation $({}^{A}T_{B})^{*}$ is given by the multiplication of ${}^{A}T_{B}$ by the matrix representing the transformation. (Post-multiplication)

Example: (2.10 from the book)

A point in space is defined as ${}^{B}P = [2,3,5]^{T}$ relative to frame *B*, which is attached to the origin of the reference frame *A* and is parallel to it. Apply the following transformations to frame *B* and find ${}^{A}P$.

- Rotate 90° about the *x*-axis.
- Then, rotate 90° about the local *a*-axis.

- Then translate 3 units about the y-axis, 6 units about the z-axis and 5 units about the x-axis.

  (Common notation: *x*,*y* and *z* are the coordinates of the reference frame *A*, and *n*,*o* and *a* are the coordinates of the non-reference frame *B*)

Solution:

Initially, the frames A and B are parallel, and they share the same origin.

Then:

$$^{A}T_{B} = I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- The first transformation results in:

$^{A}T_{B}(new) = Rot(x,90^{\circ})* {}^{A}T_{B}(old) = Rot(x,90^{\circ})*I = Rot(x,90^{\circ})$

- The second transformation results in:

$^{A}T_{B}(new) = {}^{A}T_{B}(old)* Rot(z,90^{\circ}) = Rot(x,90^{\circ})* Rot(z,90^{\circ})$

- The third transformation results in:

$^{A}T_{B}(new) = Translate(5,3,6)*{}^{A}T_{B}(old) = Translate(5,3,6)*Rot(x,90^{\circ})* Rot(z,90^{\circ})$

- As *P* is attached to the coordinate frame *B*, the location of *P* did not change w.r.t *B*, while the new location of *P* w.r.t *A* is given by:

  $^{A}P = {}^{A}T_{B} * {}^{B}P$

Solution with Matlab:

```
>> transl(5,3,6)*trotx(pi/2)*trotz(pi/2)*[2;3;5;1]
ans =
     2
    -2
     8
     1
>>
```

Exercise 4 (Homework):

1- Verify all of the book examples (2.6, 2.7 , 2.8, 2.9, 2.10, 2.11 and 2.12) with Matlab.

2- Solve the book problems (2.7, 2.8, 2.9 and 2.12).

   Write only the Matlab commands for each problem, and give your own comments.

Only Email submissions are accepted (you have exactly 6 days to submit)

Paper solutions are not accepted under any circumstances

- Take clear photos of your paper solutions and combine them all in a single pdf file.
- Send your Matlab codes in a separate file (m-file or word document).

  Submit to the Email: lab.eeng428@gmail.com
- In case of emergency, contact me on the Email: (Don't visit me in the office before writing an Email explaining your problem)

Mohamad.Harastani@emu.edu.tr