EENG 428 Introduction to Robotics Laboratory

Lab Session 2

Objective

This Session aims to explore the usage of symbolic Matlab in order to simplify the robotic analysis.

1- Introduction to symbolic variables, expressions and function

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.

Defining the Symbolic: Using Matlab, the Matlab expression ***syms*** (***sym*** for single variable) allows us to define symbolic variables, and the function ***sym ()*** allows us to define symbolic expressions.

Example 1.1

Write the following expression in matlab using symbolic variables

$$G = ax^4 + bx^3 + e^x + c$$

```
syms G x a b c
G= a*x^4 + b*x^3 + exp(x)+ c
```

- We can directly write G as a symbolic expression

```
G=sym('a*x^4 + b*x^3 + exp(x)+ c')
```

- We also can express G as a symbolic function as follow

```
syms  G(x) a b c
G(x)=a*x^4 + b*x^3 + exp(x)+ c
G(3)
```

Prepared by: Eyad Almasri
eyad.almasri@emu.edu.tr

Substituting Numbers: After defining an expression, we can substitute values instead of some symbols using the command ***subs (S, OLD, NEW)***. This function replaces OLD with NEW in the symbolic expression S.

Example 1.2

Write the following expression in matlab using symbolic variables

$$G = e^x + \log(y)$$

And find the value of G by replacing x and y by 1 and 2 respectively.

```
syms x y
G = exp(x) + log(y)
f=subs(G,[x y],[1 2])
numeric=double(f)
```

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 ()***.

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 1.3

Write the following expression in matlab using symbolic variables

$$G = \frac{\sin(x)^2 - 1}{2\,x^2 + x}$$

And plot G when x varying from 2 to 4

```
syms x y
G = ((sin(x)^2)-1)/((2*x^2)+x)
pretty(G)
ezplot(G,[2 4])
```

Prepared by: Eyad Almasri
eyad.almasri@emu.edu.tr

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

Example 1.4

Given the following linear system

$$\boldsymbol{Ax = b}$$

Where A is a 2×2 invertible matrix, x is a column vector with 2 entries, and b is a column vector with 2 entries. Represent and solve this system using symbolic matlab.

```
% we assume A is invertible 2*2 matrix
A=sym('a',[2 2])
% syms a b c d
% A=[a b;c d]
x=sym('x',[2 1])
b=sym('b',[2 1])
S=inv(A)*b %% the solution of the system
```

Symbolic solve: The Symbolic Math Toolbox supports the solving of equations and systems of equations using **solve( )**. It supports solving multivariate equations, solving inequalities and solving with assumptions. Solutions can be found symbolically or numerically

Example 1.5

solve the following equation using matlab symbolic solve ( )

$$ax^2 - b = 0$$

```
s=solve('a*x^2 - b == 0')
pretty(s)
```

You can observe that the matlab automatically recognize that the unknown is x and solve the equation based on that assumption. We can change this presumption by explicit the desired variable inside the solve statement.

```
s=solve('a*x^2 - b == 0', 'a'); % solve for a
pretty(s)
```

Prepared by: Eyad Almasri
eyad.almasri@emu.edu.tr

Solving a set of algebraic equations: Matlab can help solving a set of equations as in the following example

Example 1.6

Solve the following set of equation

$$\begin{bmatrix} 1 & 1 & 1 \\ 5 & 5 & 7 \\ 3 & 1 & 3 \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 4 \\ 18 \\ 3 \end{bmatrix}$$

```
F=solve('x+y+z=4','5*x+5*y+7*z=18','3*x+1*y+3*z=3')
solution=[f.x
          f.y
          f.z]
```

In Calculus: The Symbolic Math Toolbox has a full set of calculus tools for applied mathematics. It can perform multivariate symbolic integration and differentiation.

Example 1.7

Consider the function

$$f(x,y) = \sin\left(\frac{\sqrt{e^{xy}+1}}{2xy}\right)$$

Use Matlab to find

$$\frac{\partial f(x,y)}{\partial y}$$

```
syms f(x,y)
f(x,y)=sin(sqrt(exp(x*y)+1)/(2*x*y));
pretty(f)
D=diff(f,y);
pretty(simplify(D))
```

Prepared by: Eyad Almasri
eyad.almasri@emu.edu.tr

Example 1.8

Consider the function

$$f(x) = x\ log(1+x)$$

Use Matlab to find

$$\int_0^1 f(x)dx$$

```
syms f(x)
f(x)=x*log(1+x)
int(f,x,0,1)
```

Differential Equations: The Symbolic Math Toolbox can analytically solve systems of ordinary differential equations using *dsolve*().

Example 1.9

Solve the first order ODEs

$$\frac{dy}{dx} = -ay$$

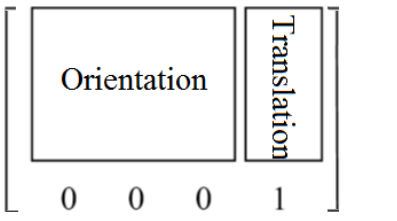
with the initial condition $y(0) = b$

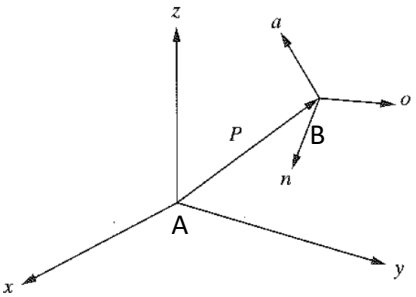
```
syms a b y(x)
dsolve(diff(y)== -a*y,y(0)==b)
```

2- General Transformation Matrices

A general transformation is a Block matrix that combines two sets of information, the orientation of one coordinate frame (the non-reference frame) w.r.t another (the reference frame), and the position of the non-reference frame w.r.t the reference frame

In other words, a frame can be expressed in terms of the fixed reference frame by three vectors describing its directional unit vector, as well as the forth vector describing its location



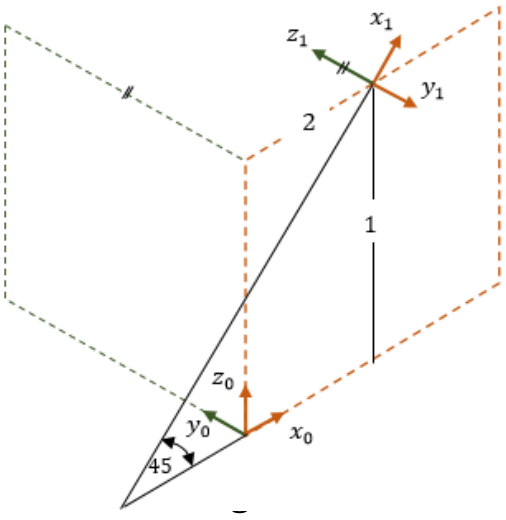


$$^{A}T_{B} = \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Example 2.1:

Finding the 4 by 4 homogeneous transformation matrix which represent the frame 1 relative to the frame 0

$$ {}^{0}T_{1} = \begin{bmatrix} \cos 45 & \cos 45 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ \sin 45 & -\sin 45 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$




```
%% General Transformation Matrix
To=[1/sqrt(2)    1/sqrt(2)     0      2
         0          0          1      1
    1/sqrt(2)   -1/sqrt(2)     0      1
         0          0          0      1];
%% Representative tools to draw the relation btw
two Frames
Hfr = [0 1 0 0 0 0 0
       0 0 0 1 0 0 0
       0 0 0 0 0 1 0
       1 1 1 1 1 1 1 ];
frame0={'';'x0';'';'y0';'';'z0';''};
frame1={'';'x1';'';'y1';'';'z1';''};
xp=To(1,4) ; yp=To(2,4); zp = To(3,4);
plot3( Hfr(1,:), Hfr(2,:), Hfr(3,:) ,'b',
'LineWidth',2);
hold on;
xlabel('X');ylabel('Y');zlabel('Z');
S1=size(Hfr);
for i=1:S1(2)
    c=Hfr(4,i); if c==0, c=1;end
    x=Hfr(1,i)/c;y=Hfr(2,i)/c;z=Hfr(3,i)/c;
    text(x,y,z,frame0(i));
end
Tfr = To * Hfr;
plot3( Tfr(1,:), Tfr(2,:), Tfr(3,:) ,'r-.',
'LineWidth',2);
axis equal; grid on;
S2=size(Tfr);
for i=1:S2(2)
    c=Tfr(4,i); if c==0, c=1;end
    x=Tfr(1,i)/c;y=Tfr(2,i)/c;z=Tfr(3,i)/c;
    text(x,y,z,frame1(i));
end
axis equal;
hold on;
x=xp; y=yp; z=zp;
xa= [ 0 x x x];ya= [ 0 0 y y];za= [ 0 0 0 z];
plot3(xa, ya, za);hold on;
plot3( x, y, z, 'ko', 'LineWidth',2);
```

Orthogonal Matrix

- A matrix is said to be orthogonal if and only if
  1- All its columns (rows) are unit vectors (vectors of length 1).
  2- Each of its columns (rows) is perpendicular to the rest.

If A is an orthogonal matrix, then

$$ \begin{cases} A^T A = A\, A^T = I_n \\ A^T = A^{-1} \\ Det(A) = 1 \end{cases} $$

It is proven that the **Orientation Part** of the General Transformation matrix is **Orthogonal matrix** based on the definition of the General transformation matrix (check [3] for prove)

Prepared by: Eyad Almasri
eyad.almasri@emu.edu.tr

- **Inversion of the General Transformation Matrix**

The General Transformation Matrix was introduced as a block-type matrix from the form:

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

Where R is an orthogonal matrix, and P is a vector.

This matrix is similar to the block matrix $\begin{bmatrix} A & B \\ 0 & D \end{bmatrix}$ which is known to have the following property

$$\begin{bmatrix} A & B \\ 0 & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & -A^{-1}B\,D^{-1} \\ 0 & D^{-1} \end{bmatrix}$$

Then, the inverse of the general transformation matrix $^{A}T_{B}$ as follow

$$\begin{bmatrix} \boldsymbol{R} & \boldsymbol{P} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{R^{T}} & \boldsymbol{-R^{T}P} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$$

- ✓ If $^{A}T_{B}$ represents the frame B w.r.t the frame A then,

$$^{B}T_{A} = \left(^{A}T_{B}\right)^{-1}$$

3- Basic Transformation matrices using Matlab toolbox

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

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

Prepared by: Eyad Almasri
eyad.almasri@emu.edu.tr

Example 3.1

Representation of a pure translation or a pure Rotation about an axis

Define a rotational transformation matrix about x-axis with angle t

Define a rotational transformation matrix about y-axis with angle t

Define a rotational transformation matrix about z-axis with angle t

Define a translation transformation matrix along x, y, z axes with distance $px\ py\ pz$ respectively. Replace $px\ py\ pz$ with 1,2,3 respectively.

```
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
A=transl(px,py,pz) % translate along x,y and z axes
AA=subs(A,[px,py,pz],[1,2,3])
```

References:

1- Moore, H. (2017). *MATLAB for Engineers*. Pearson.

2- Niku, S. (2010). *Introduction to robotics*. John Wiley & Sons.

Prepared by: Eyad Almasri
eyad.almasri@emu.edu.tr