
Computer Programming of Robot Kinematics

TERM PROJECT

FOUNDATIONS OF ROBOTICS
ROB-GY 6003
SECTION A

Prajwal Prakshep Somawanshi

Net ID: ps4518

Univ ID: N15786948



Contents

Programming Exercise

Chapter 2 - Spatial Descriptions and Transformation

Chapter 3 - Manipulator Kinematics

Chapter 4 - Inverse Manipulator Kinematics

##Notation used to describe transformations matrix are

- The notations used for Transformation matrices are T_{\quad} . Where The 1st blank is the frame of interest and the 2nd blank is the frame with respect to it is written.
- Example of the notation used in this exercise is 'Tab' which says transformation matrix of frame{b} with respect to frame {a}.##

Spatial Descriptions and Transformations

Programming exercise (part 2)

1. Creation function Atan2

- Function Atan2 calculates the angle of a given coordinate of a point considering the quadrant where the point lies.
- Function Atan2 considers the 'signs' of sine and cosine values. The arcTan2 function returns a single value of θ theta which lies in $-\pi < \theta \leq \pi$.
- Following program defines the function of atan2 to calculate the angle

```
Editor - C:\Users\somaw\Desktop\FOR PROJECT\arctan2.m
atan2program.m x arctan2.m x +
1      %%%FUNCTION - arctan2%%
2      %x,y are user input coordinates of a point in any of 4 quadrants
3      function Radian = arctan2(y,x)
4      if (x>0&&y>0)
5          Radian=atan(y/x);
6      elseif( x>0&&y<0)
7          Radian=atan(y/x);
8      elseif (x<0&& y>=0)
9          Radian=pi+atan(y/x);
10     elseif(x<0&&y<0)
11         Radian=atan(y/x)-pi;
12     elseif(x==0&&y==0)
13         Radian='Undefined angle'
14     elseif(x==0&&y>0)
15         Radian=pi/2;
16     else(x==0&&y<0)
17         Radian=-pi/2;
18     end
```

- The following program calls the defined function of atan2 and calculates the theta angle in radians and degree. The function atan2 is also called with the help of an example where the point lies in the 4th quadrant and the angle is determine by the defined function 'atan2'

```
Editor - C:\Users\somaw\Desktop\FOR PROJECT\atan2program.m*
+1 ForwardKinematics.m x KIN.m x WHERE.m x chapt3kine.m x Chapt3p3.m x Chapter3Eqn.
1   %%% MATLAB PROGRAMMING
2   %%% Function - arctan2 %%%
3   %clear all the variables and screen
4   clc;close all
5   clear all;
6   % input from user of the point co-ordinates of which angle is to be
7   % calculated
8   % enter the value of co-ordinates of point with signs
9   prompt1=' what is the value of X co-ordinate?';
10  prompt2=' what is the value of Y co-ordinate?';
11  x=input(prompt1)
12  y=input(prompt2)
13  % calculation of angle and which quadrant the point lies
14  %using defined function arctan2
15  %%%FUNCTION - arctan2%%
16  Radian=arctan2(y,x)
17  Degree=(Radian*180)/pi
```

```
Command Window
what is the value of X co-ordinate?3

x =

    3

what is the value of Y co-ordinate?-6

y =

   -6

Radian =

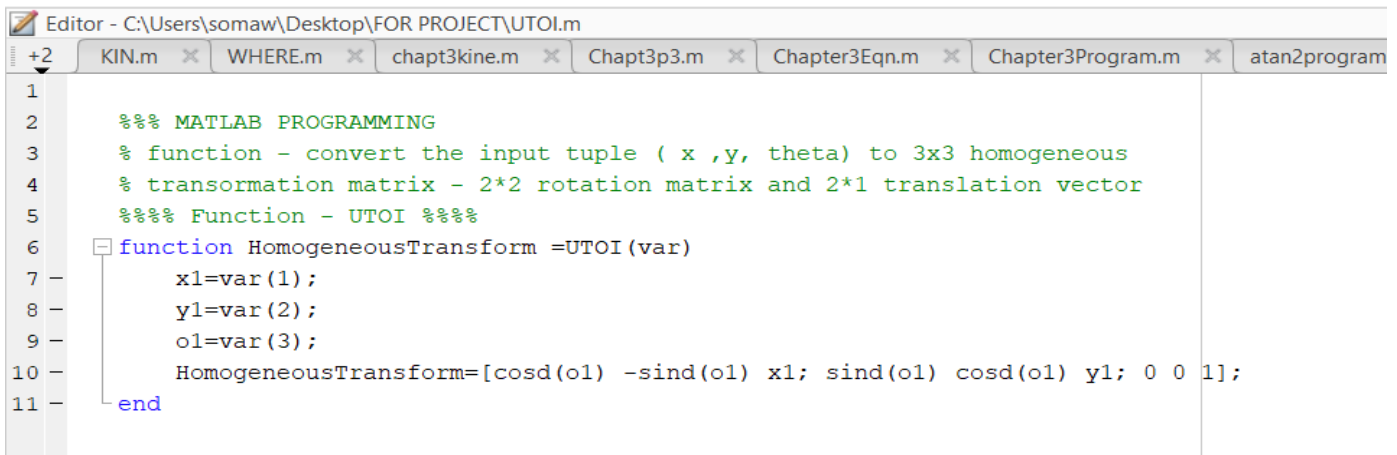
   -1.1071

Degree =

   -63.4349
```

2. Function- User form to Internal form

- The defined function UTOI (user form to internal form) takes the input from the user as a tuple where the elements of the tuple are x , y for position vector/translation, and theta for rotation matrix
- This function will transform the user input that is user form to internal form which is a 3*3 homogeneous matrix with a 2*2 rotation matrix and 2*1 position vector
- The following is the function defined for UTOI using Matlab programming



The screenshot shows a MATLAB editor window titled "Editor - C:\Users\somaw\Desktop\FOR PROJECT\UTOI.m". The window contains the following code:

```
1
2   %%% MATLAB PROGRAMMING
3   % function - convert the input tuple ( x ,y, theta) to 3x3 homogeneous
4   % transformation matrix - 2*2 rotation matrix and 2*1 translation vector
5   %%% Function - UTOI %%%
6   function HomogeneousTransform =UTOI(var)
7       x1=var(1);
8       y1=var(2);
9       o1=var(3);
10      HomogeneousTransform=[cosd(o1) -sind(o1) x1; sind(o1) cosd(o1) y1; 0 0 1];
11  end
```

- The function can be called by providing an input tuple (var). Then the function UTOI transforms the user form into an internal form. The call of this function is shown with the help of an example from user input.
- In the example below the user gave input in the form of a tuple with values for X, Y, Theta

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Program.m
Program.m x ForwardKinematics.m x KIN.m x WHERE.m x chapt3kine.m x Chapt3p3.m x Chapter3Eqn.m x Chapter3Program.m
1
2   %% MATLAB PROGRAMMING
3   %% Calling the defined function
4   %clear all the variables and screen
5   clc;close all
6   clear all;
7   % input from user - translation co-ordinates and angle of rotation for
8   % formation of 3x3 homogeneous translation matrix
9   prompt1=' what is the value of "x" "y" "theta in degree" for 1st H.T matrix?';
10  variable1=input(prompt1);
11  x1=variable1(1);
12  y1=variable1(2);
13  o1=variable1(3);
14  %Writing the 1st 3x3 homogeneous translation matrix by calling the defined
15  %function 'UTOI' using user input parameters
16  Answer2a=' Following in 3x3 homogeneous transformation martix with user form input parameters'
17  InternalFormHT=UTOI(variable1)
18  % where variable1 is in user form and answer is in Internal Form

```

- Entering the input values as x=3, y=5, and Theta = 30, will give a homogeneous matrix in internal form

```

Command Window
what is the value of "x" "y" "theta in degree" for 1st H.T matrix?[3 5 30]

Answer2a =

    ' Following in 3x3 homogeneous transformation martix with user form input parameters'

InternalFormHT =

    0.8660    -0.5000    3.0000
    0.5000     0.8660    5.0000
         0         0     1.0000

```

3. Function - Internal form to User form

- The defined function ITOU (internal form to user form) takes the input from the user as a 3*3 homogeneous matrix where 1st block is a 2*2 rotation matrix and 2nd block is a 2*1 position vector and 3rd row [0 0 1]

- This function will transform the internal form that is input to the user form which is a tuple where the elements of the tuple are x, y for position vector/translation, and theta for rotation matrix
- The following is the function defined for ITOU using Matlab programming

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\ITOU.m
+2 KIN.m WHERE.m chapt3kine.m Chapt3p3.m Chapter3Eqn.m Chapter3Program.m TMULT.m
1
2   %%% MATLAB PROGRAMMING
3   % function - convert user input 3x3 homogeneous to tuple ( x ,y, theta) where x ,y
4   % are the position vector and theta is angle of rotation
5   %%% Function - ITOU %%%
6   function tupleform =ITOU(MatHT)
7       X=MatHT(1,3);
8       Y=MatHT(2,3);
9       O=atan2d(MatHT(2,1),MatHT(1,1));
10      tupleform=[X Y O];
11  end

```

- The function can be called by providing an input matrix (MatHT). Then the function UTOI transforms the user form into an internal form. The call of this function is shown with the help of an example from user input.
- In the example below the user gave input in form 3*3 homogeneous matrix where the output is a tuple with values for X, Y, and Theta in the user form.'

```

23 % input from user- Internal Form Homogeneous Transformation matrix for
24 % formation of user form tuple ( X Y THETA)
25 %%% Function - ITOU %%%
26 - prompt2='input - 3x3 homogeneous transformation matrix to convert to user form tuple';
27 - MatHT=input(prompt2);
28 - Answer2b=' Following in tuple of ( X Y THETA in degree ) of Internal form input Homogeneous Transformation matrix';
29 - UserformTuple=ITOU(MatHT);
30 %the ' UserformTuple' is a in form ( X Y THETA)

```

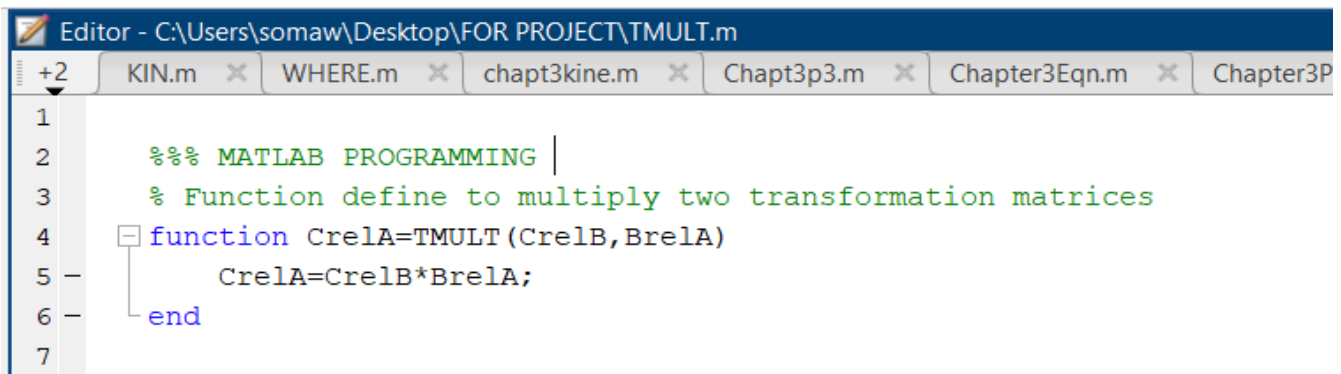
- In the following code, we will check the function ' ITOU' with the given input of a 3*3 homogeneous matrix.

Command Window

```
prompt2 =  
  
    'input - 3x3 homogeneous transformation matrix to convert to user form tuple'  
  
input - 3x3 homogeneous transformation matrix to convert to user form tuple[0.5 -0.866025 3; 0.866025 0.5 6; 0 0 1]  
  
Answer2b =  
  
    ' Following in tuple of ( X Y THETA in degree ) of Internal form input Homogeneous Transformation matrix'  
  
UserformTuple =  
  
    3.0000    6.0000    60.0000
```

4. Function - TMULT

- The function 'TMULT' is defined in the following code where the input arguments are two internal form matrices where the output of the function is a matrix that resultant of the multiplication of two input matrices



The screenshot shows the MATLAB Editor window with the file 'FOR PROJECT\TMULT.m' open. The editor displays the following code:

```
1  
2     %%% MATLAB PROGRAMMING |  
3     % Function define to multiply two transformation matrices  
4     function CreIA=TMULT(CreIB,BreIA)  
5     CreIA=CreIB*BreIA;  
6     end  
7
```

- After defining it we can call the function 'TMULT'. One example is by given the defined matrices as input arguments to the function 'TMULT' and other example will calling the function by taking input arguments by user
- So the solution is also mentioned below


```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Program.m
ATAN2.m x UTOI.m x Program.m x ITOU.m x TINVERT.m x TMULT.m x +
33 %
34 %Multiplication of two Input matrices or defined matrices
35 %%function called for multiplication of two transformation matrices in internal Form
36 %calling Predefined function TMULT - by defined matrices
37 %%%Function- TMULT%%
38 A=[0.5 0.866 1;-0.866 0.5 3; 0 0 1];
39 B=[0.707 -0.707 5;0.707 0.707 6; 0 0 1];
40 Answer='Following matrix is multiplication of A , B '
41 Answer = TMULT(A,B)
42 %Multiplication of matrices input by user in userform
43 prompt3='input - the value of "x" "y" "theta in degree" for 1st H.T matrix in user form';
44 prompt4='input - the value of "x" "y" "theta in degree" for 2nd H.T matrix in user form';
45 %convert them to Internal form by calling function UTOI
46 %%%FUNCTION -UTOI%%
47 variable3=input(prompt3);
48 variable4=input(prompt4);
49 InternalFormHT1=UTOI(variable3);
50 InternalFormHT2=UTOI(variable4);
51 %Following matrix is multiplication of internal form matrices formed by
52 %tuple inputs of (X, Y THETA) for 2 two matrices
53 %%% FUNCTION - TMULT%%
54 InternalFormMultiHT='Following matrix is multiplication of internal form matrices formed by tuple inputs of (X, Y THETA) for 2
55 InternalFormMultiHT=TMULT(InternalFormHT1,InternalFormHT2)
56 %

```

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Program.m
Command Window
Answer =

'Following matrix is multiplication of A , B '

Answer =

    0.9658    0.2588    8.6960
   -0.2588    0.9658    1.6700
         0         0    1.0000

input - the value of "x" "y" "theta in degree" for 1st H.T matrix in user form[3 6 45]
input - the value of "x" "y" "theta in degree" for 2nd H.T matrix in user form[4 5 60]

InternalFormMultiHT =

'Following matrix is multiplication of internal form matrices formed by tuple inputs of (X, Y THETA) for 2 two matrices'

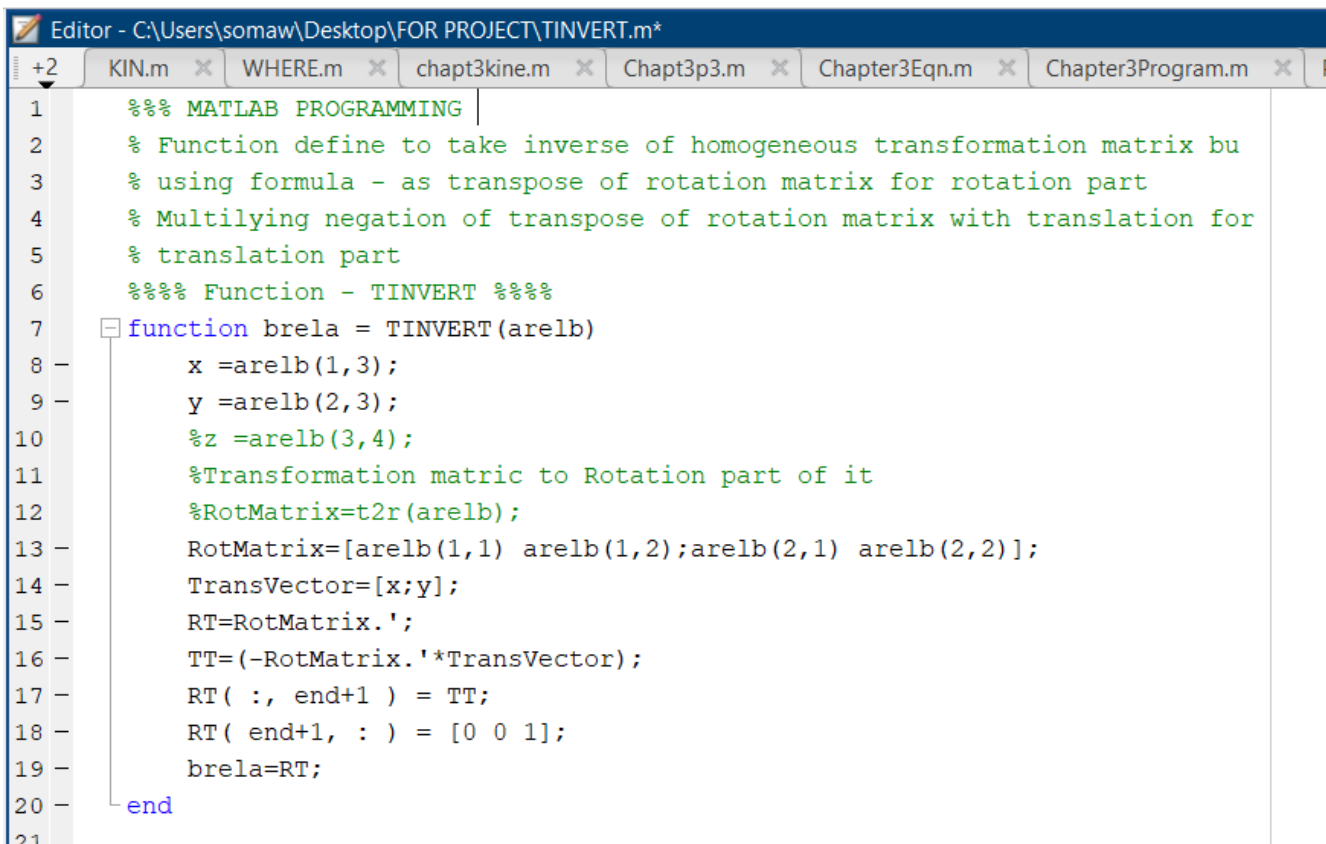
InternalFormMultiHT =

   -0.2588   -0.9659    2.2929
    0.9659   -0.2588   12.3640
         0         0    1.0000

```

5. Function - TINVERT

- The following function 'TINVERT' is a function defined to calculate the inverse of the given transformation matrix. The inverse of the transformation matrix describes the frame of interest and with respect to which frame with each other.
- The function is defined for the 3×3 transformation matrix where 1st 2×2 part is the rotation part and 2×1 is the Translation vector and the last row $[0 \ 0 \ 1]$;
- As per the the formula of the inversion of the 3×3 transformation matrix the 'TINVERT' function is defined as by using the formula where the transpose of rotation part is the rotation part of the answer and translation part is the multiplication of negation Rotation part transpose with position vector and this is used below



```
Editor - C:\Users\somaw\Desktop\FOR PROJECT\TINVERT.m*
+2  KIN.m  WHERE.m  chapt3kine.m  Chapt3p3.m  Chapter3Eqn.m  Chapter3Program.m
1      %%% MATLAB PROGRAMMING
2      % Function define to take inverse of homogeneous transformation matrix bu
3      % using formula - as transpose of rotation matrix for rotation part
4      % Multilying negation of transpose of rotation matrix with translation for
5      % translation part
6      %%% Function - TINVERT %%%
7      function brela = TINVERT(arelb)
8          x =arelb(1,3);
9          y =arelb(2,3);
10         %z =arelb(3,4);
11         %Transformation matric to Rotation part of it
12         %RotMatrix=t2r(arelb);
13         RotMatrix=[arelb(1,1) arelb(1,2);arelb(2,1) arelb(2,2)];
14         TransVector=[x;y];
15         RT=RotMatrix.';
16         TT=(-RotMatrix.'*TransVector);
17         RT( :, end+1 ) = TT;
18         RT( end+1, : ) = [0 0 1];
19         brela=RT;
20     end
21
```

Transform Equations / Compound Transformations

- Using the given internal form of three transformation matrices T_{ua} , T_{ab} , T_{uc} . Then using transformations arithmetic and compound transformations. To determine the transformation matrix $frame\{c\}$ relative to $frame\{b\}$.
- The following code calculates the final Transformation matrix of Frame C relative to B frame using the Function UTOI, Function TINVERT, Function TMULT

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Program.m*
Program.m* x +
72 %%
73 %%% MATLAB PROGRAMMING
74 %%% defining the given transformation in internal form that are given in
75 %%% user form Tau , Tab and Tcu using there user form
76 %%%Given the Frame definitions
77 - var1=[11 -1 30];
78 - var2=[0 7 45];
79 - var3=[-3 -3 -30];
80 %FUNCTION- USER TO INTERNAL FORM UTOI
81 - Tau=UTOI(var1);
82 - Tab=UTOI(var2);
83 - Tuc=UTOI(var3);
84 %%%FUNCTION - TINVERT
85 - Tua=TINVERT(Tau);
86 - Tcu=TINVERT(Tuc);
87 %Using the compounding indentity of transformation matrices
88 - Txy=TMULT(Tab,Tua);
89 %Following is the required Transformation matrix in Internal Form
90 - Tcb=TMULT(Txy,Tcu)
91 %%% FUNCTION - ITOU ( USER FORM )
92 - crelb=ITOU(Tcb)
93

```

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Program.m
Program.m x +
Command Window

Tcb =

    0.7071    -0.7071   -10.8840
    0.7071     0.7071    9.3616
         0         0     1.0000

crelb =

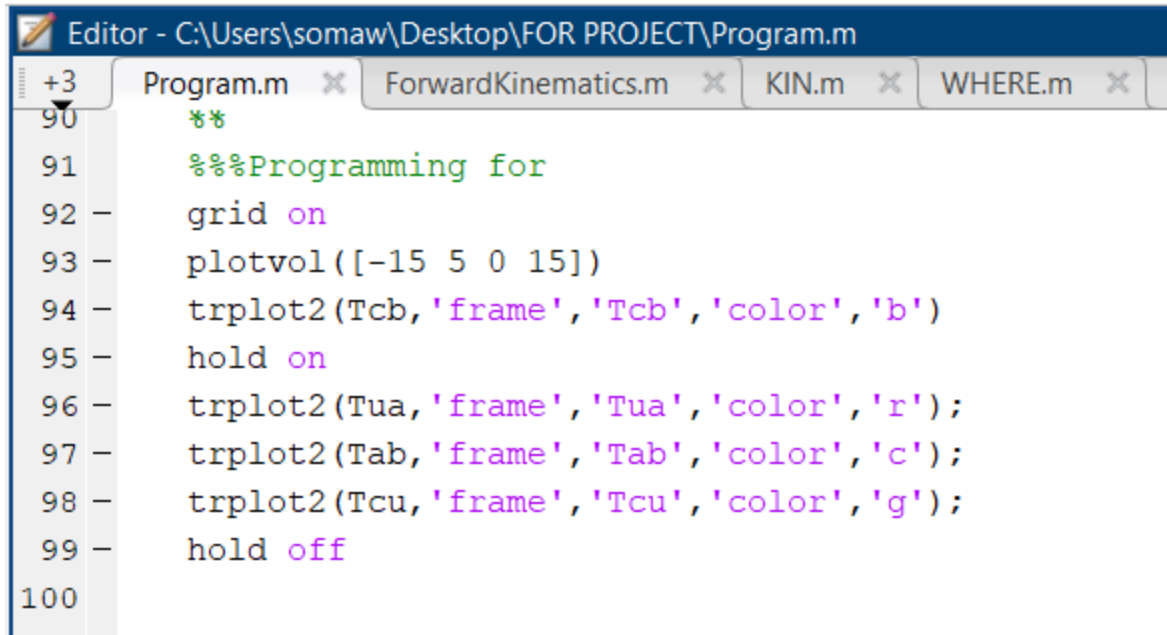
   -10.8840    9.3616   45.0000

```

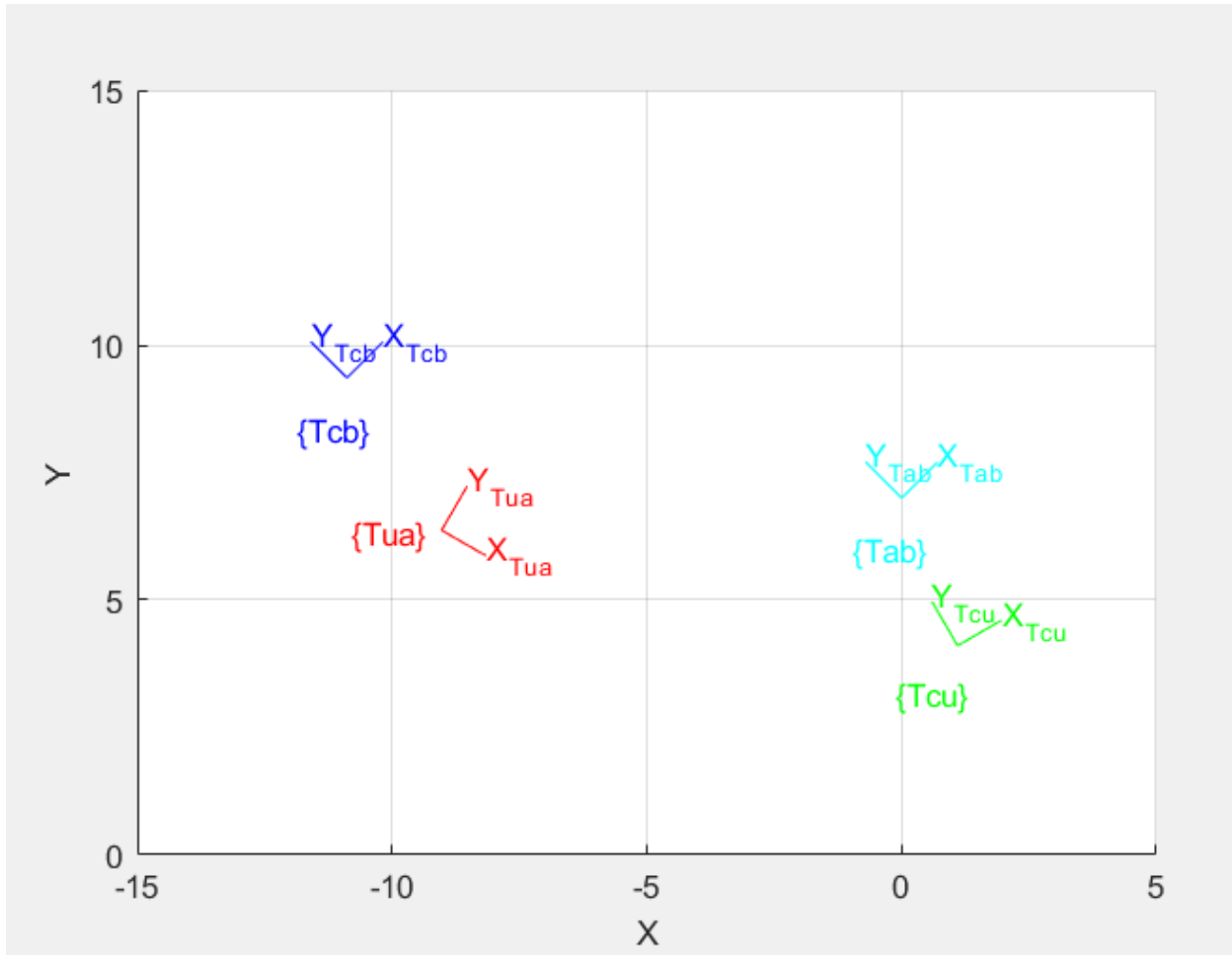
The above solution of crelb in internal and user form where the angle is in Degrees.

FRAME DIAGRAM- Compounding frames

Following is the program to draw the frames used for transformation compounding to find the Transformation matrix 3*3 Frame {c} relative to frame {b}



```
Editor - C:\Users\somaw\Desktop\FOR PROJECT\Program.m
Program.m x ForwardKinematics.m x KIN.m x WHERE.m x
90 %%
91 %%%Programming for
92 - grid on
93 - plotvol([-15 5 0 15])
94 - trplot2(Tcb,'frame','Tcb','color','b')
95 - hold on
96 - trplot2(Tua,'frame','Tua','color','r');
97 - trplot2(Tab,'frame','Tab','color','c');
98 - trplot2(Tcu,'frame','Tcu','color','g');
99 - hold off
100
```



Manipulator Kinematics

- The following programming exercise is answered according to the Standard DH convention and following the notations and terminologies used in lecture notes.
- The Transformation matrix used is a 3*3 matrix where 1st 2*2 is the rotation part and 2*1 is the translation part.
- The notations used for Transformation matrices are $T_{\text{ } _}$. Where The 1st blank is the frame of interest and the 2nd blank is the frame with respect to it is written.
- An example of the notation used in this exercise is T_{ab} which says transformation matrix of frame{b} with respect to frame {a}.

Programming exercise (part 3)

The following program computes homogeneous transformation matrices and forward kinematics.

- Derivation of link transformation $i-1 T_i$: define Frame $\{i\}$ relative to Frame $\{i-1\}$. Four transformations (sub-problems) – each of the four transformations will be a function of one DH parameter only.
- Transformations $\text{Rot}(z, \theta)$ $\text{Transl}(0, 0, d)$ $\text{Transl}(a, 0, 0)$ $\text{Rot}(x, \alpha)$

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapter3Eqn.m
TINVERT.m x chapt3kine.m x KIN.m x Chapter3Eqn.m x Chapt3p3.m x +
1  %% MATLAB PROGRAMMING
2  %% MATLAB R2021a and Robotics Toolbox released 10.4 is used
3  %Following is the program to compute homogeneous transformation matrices forward kinematics
4  clc; clear all;
5  syms theta alpha a d
6  %following Denavit-Hartenberg (DH) Convention
7  %Transformation matrix of frame{i} relative to frame {i-1}
8  %define = Rot(z,theta),Trans(0,0,d),Trans(a,0,0),Rot(x,alpha)
9  Rz=[cos(theta) -sin(theta) 0 0; sin(theta) cos(theta) 0 0; 0 0 1 0; 0 0 0 1];
10 Tz=[1 0 0 0; 0 1 0 0; 0 0 1 d; 0 0 0 1];
11 Rx=[1 0 0 0; 0 cos(alpha) -sin(alpha) 0; 0 sin(alpha) cos(alpha) 0; 0 0 0 1];
12 Tx=[1 0 0 a; 0 1 0 0; 0 0 1 0; 0 0 0 1];
13 %link Transformation T = Rot(z,theta)*Trans(0,0,d)*Trans(a,0,0)*Rot(x,alpha)
14 %link Transformation matrix for forward kinematics
15 %define: frame{i} relative to frame {i-1}
16 T = simplify(Rz*Tz*Tx*Rx);
17 Forwardkinematics=T

Command Window

Forwardkinematics =

[cos(theta), -cos(alpha)*sin(theta), sin(alpha)*sin(theta), a*cos(theta)]
[sin(theta), cos(alpha)*cos(theta), -sin(alpha)*cos(theta), a*sin(theta)]
[0, sin(alpha), cos(alpha), d]
[0, 0, 0, 1]

```

FUNCTION ' ForwardKinematics '

- The solution obtained from the above program of forward Kinematics matrix is used to define the function 'ForwardKinematics' where it computes the Transformation matrix of frame $\{i\}$ relative to frame $\{i-1\}$ by giving the following Joint Variables for 3R planar robot in 3*3 Transformation matrix

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\ForwardKinematics.m
Program.m x ForwardKinematics.m x KIN.m x +
1 %following function will calculate the homogeneous transformation matrices
2 %for each links for 3R Planar Robot in 3*3 Homogeneous TransformationMatrix
3 %using the solution developed from Forward kinematics program
4 %result of Transformation matrix of frame{i} relative to frame {i-1}
5 function TMatrix= ForwardKinematics(theta)
6     L=0.5;
7     TMatrix=[cosd(theta) -sind(theta) L*cosd(theta); sind(theta) cosd(theta) L*sind(theta); 0 0 1];
8 end

```

1. FUNCTION 'KIN'

- FUNCTION 'KIN' is defined using the defined function 'ForwardKinematics' where we can compute the transformation matrix (3*3) of wrist frame relative to the base frame for 3 R planar manipulator by giving the set of joint variables for the proper position orientation (pose) of end-effector or wrist frame

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\KIN.m*
+5 KIN.m* x WHERE.m x chapt3kine.m x Chapt3p3.m x Chapter3Eqn.m x Chapter3Program.m x TMULT
1 %following function will calculate the homogeneous transformation matrices
2 %for each links
3 %using the solution developed from Forward kinematics program
4 %result of Transformation matrix of frame{i} relative to frame {i-1}
5 function Twb=KIN(theta1,theta2,theta3)
6     %%FUNCTION - ForwardKinematics
7     T10=ForwardKinematics(theta1);
8     T21=ForwardKinematics(theta2);
9     T32=ForwardKinematics(theta3);
10    %%%FUNCTION - TMULT %%%
11    Twb=(TMULT(T10,T21),T32);
12
13 end
14

```

- The input arguments of the FUNCTION - KIN are the three joint variables theta1, theta2, and theta3 which compute the 3*3 transformation matrix wrist relative to the base frame.
- The above code computes the Transformation matrix wrist frame relative to the base.

2. FUNCTION 'WHERE'

- The following code defines the function/subroutine 'where' which calculates the tool frame relative to the station frame. According to the equation -

$$T(\text{Tool frame relative to Station frame}) = \text{Inverse of } T(\text{Station frame relative to Base frame}) * T(\text{Wrist frame relative to Base frame}) * T(\text{Tool frame relative to Wrist frame})$$
- The inverse of Tsb is found using TINVERT function

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\WHERE.m*
+12 KIN.m x WHERE.m* x chapt3kine.m x Chapt3p3.m x Chapter3Eqn.m x Chapter3Pro
1  %program to write transformation matrix of
2  %Tts = tool frame relative to station frame
3  %using compounding of transformation matrix
4  %%% FUNCTION - WHERE
5  %Tsb-transformation matrix of station relative to base
6  %Twb-transformation matrix of wrist relative to base
7  %Ttw-transformation matrix of tool relative to wrist
8
9  function Tts=WHERE(theta1,theta2,theta3,Tsb,Ttw)
10 -     Twb=KIN(theta1,theta2,theta3);
11     %%% FUNCTION - TINVERT %%%
12 -     iTsb=inv(Tsb);
13     %%%FUNCTION - TMULT %%%
14 -     Tts=TMULT(TMULT(iTsb,Twb),Ttw);
15 -     end
16

```

3. Calculate the position and orientation of the Tool frame relative to the Station frame

- To calculate the transformation matrix of tool frame relative to station frame for different configurations of joint variables we need to use the function 'WHERE' where the input arguments are Tsb, Twb, Ttw which are defined in the program with help of the given values
- For Ttw the given x, and y parameters in the translation are 0.1, 0.2 and the theta in rotation parameter is 30 degrees
- For Tsb the given x, y parameters in the translation are -0.1, 0.3 and the theta in rotation parameter is 0 degrees.
- The 1st given configuration of joint variables [0 90 -90] and given link parameters of link length [0.5 0.5 0.5]. According to these joint angles and link parameters, DH parameters are defined through which the forward kinematics that is Transformation of Wrist frame relative to the base frame for this particular configuration is calculated

(Twb1) using the function 'KIN' and later using the defined Ttw and Tsb and function 'WHERE' the program calculates Transformation of Tool frame relative to station frame for 1st given joint configurations

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapt3p3.m*
+2 KIN.m x WHERE.m x chapt3kine.m x Chapt3p3.m* x Chapter3Eqn.m x Chapter3Program.m x TMULT.m x TINVERT.m
1  %%% MATLAB PROGRAMMING
2  %define Ttw and Tsb for given set of values in 3*3 homogeneous
3  %transformation matrix
4  % given value for Ttw x=0.1 , y=0.2, theta=30;
5  Variable1=[0.1 0.2 30];
6  %%%FUNCTION UTOI
7  Ttw=UTOI(Variable1);
8  %given value for Tsb x=-0.1 , y=0.3, theta=0;
9  Variable2=[-0.1 0.3 0];
10 Tsb=UTOI(Variable2);
11 %calculation of wrist relative to base frame
12 %according to 1st set of joint variables
13 theta1=0;
14 theta2=90;
15 theta3=-90;
16 %calling the defined function for formation of homogeneous matrices 'KIN'
17 %%% FUNCTION - KIN%%
18 Twb1=KIN(theta1,theta2,theta3);
19 %Twb = Transformation matrix of wrist relative to base frame calculated by
20 %using forward kinematics using standard Denvait-Hartenberg
21 %calculating the position and orientation of tool relative to station frame
22 %for 1st joint variables configuration
23 %%%FUNCTION - WHERE%%
24
25 Tts1='Transformation of Tool frame relative to Station frame for 1st given Joint variables'
26 Tts1=WHERE(Tsb,Twb1,Ttw)
27 trels1=ITOU(Tts1)

```

The solution of the code is transformation matrix for tool relative to station frame for configuration 1 is in internal and user form with the angle in degree

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapt3p3.m
Command Window

Tts1 =

    'Transformation of Tool frame relative to Station frame for 1st given Joint variables'

Tts1 =

    0.8660    -0.5000    1.2000
    0.5000     0.8660     0.4000
         0         0     1.0000

trels1 =

    1.2000    0.4000   30.0000

```

- We will compute the value of Transformation of tool frame relative to station frame for 2nd given joint variables configuration [-23.6 -30.3 48.0] and same link parameters

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapt3p3.m
Program.m x ForwardKinematics.m x KIN.m x WHERE.m x chapt3kine.m x Chapt3p3.m x Chapter3Eqn.m x Chapter3Progr
34
35 %%
36 %calculation of wrist relative to base frame
37 %according to 2nd set of joint variables
38 - theta1=-23.6;
39 - theta2=-30.3;
40 - theta3=48;
41 %calling the defined function for formation of homogeneous matrices 'KIN'
42 %%% FUNCTION - KIN%%%
43 - Twb2=KIN(theta1,theta2,theta3);
44 %Twb = Transformation matrix of wrist relative to base frame calculated by
45 %using forward kinematics using standard Denavit-Hartenberg
46 %calculating the position and orientation of tool relative to station frame
47 %for 2nd joint variables configuration
48 %%%FUNCTION - WHERE%%%
49 - Tts2='Transformation of Tool frame relative to Station frame for 2nd given Joint variables'
50 - Tts2=WHERE(Tsb,Twb2,Ttw)
51 %FUNCTION ITOU to convert Transformation Matrix to user form
52 - trels2=ITOU(Tts2)
53

```

The solution of the code is transformation matrix for tool relative to station frame for configuration 2 is in internal and user form with the angle in degree

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapt3p3.m
Program.m ForwardKinematics.m KIN.m WHERE.m chapt3kine.m Chapt3p3.m Chapter3Eqn.m Chapter3
Command Window

Tts2 =

    'Transformation of Tool frame relative to Station frame for 2nd given Joint variables'

Tts2 =

    0.9128    -0.4083    1.4702
    0.4083     0.9128   -0.7669
         0         0     1.0000

trels2 =

    1.4702   -0.7669   24.1000

```

- We will compute the value of Transformation of tool frame relative to station frame for 3rd given joint variables configuration [130 40 12.0] and same link parameters

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapt3p3.m*
Program.m ForwardKinematics.m KIN.m WHERE.m chapt3kine.m Chapt3p3.m* Chapter3Eqn.m Chapt
55
56 %%
57 %calculation of wrist relative to base frame
58 %according to 3rd set of joint variables
59 - theta1=130;
60 - theta2=40;
61 - theta3=12;
62 %calling the defined function for formation of homogeneous matrices 'KIN'
63 %%% FUNCTION - KIN%%%
64 - Twb3=KIN(theta1,theta2,theta3);
65 %Twb = Transformation matrix of wrist relative to base frame calculated by
66 %using forward kinematics using standard Denavit-Hartenberg
67 %calculating the position and orientation of tool relative to station frame
68 %for 3rd joint variables configuration
69 %%%FUNCTION - WHERE%%%
70 - Tts3='Transformation of Tool frame relative to Station frame for 3rd given Joint variables'
71 - Tts3=WHERE(Tsb,Twb3,Ttw)
72 - trels3=ITOU(Tts3)
73

```

The solution of the code is transformation matrix for tool relative to station frame for configuration 3 is in internal and user form with the angle in degree

```
Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapt3p3.m
Command Window

Tts3 =

    'Transformation of Tool frame relative to Station frame for 3rd given Joint variables'

Tts3 =

    -0.8480    0.5299   -1.3065
    -0.5299   -0.8480   -0.0510
         0         0     1.0000

treIs3 =

    -1.3065   -0.0510  -148.0000
```

Simulation of 3 R planar manipulator for the above 3 configurations

The following code takes the input as a tuple of 3 joint angle variables of three revolute planar manipulators and models it in Matlab with the help of Peter Corke's Robotic Toolbox 10.4. We can see the 3d View for these configurations.

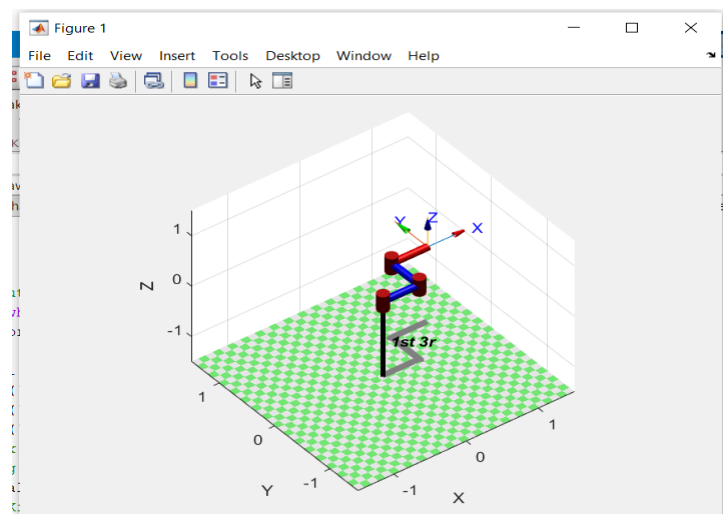
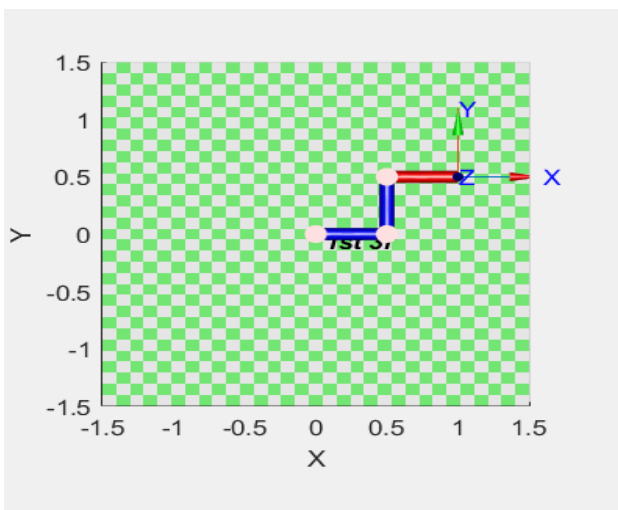
```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapter3Program.m
Chapt3p3.m x Chapter3Eqn.m x WHERE.m x trial.m x KIN.m x chapt3kine.m x Program.m x Chapter3Pr
1 - clear all; clc; close all;
2 - L1=0.5;
3 - L2=0.5;
4 - L3=0.5;
5 - %User input as joint angles of 3 revolute joints
6 - prompt1='what is joint variables for 3 revolute joints planar robot in radians ? ';
7 - qi=input(prompt1);
8
9 - DH=[0 0 L1 0 ; 0 0 L2 0 ; 0 0 L3 0 ];
10 - L(1)=Link('revolute' , 'd', DH(1,2), 'a', DH(1,3), 'alpha',DH(1,4),'standard');
11 - L(2)=Link('revolute' , 'd', DH(2,2), 'a', DH(2,3), 'alpha',DH(2,4),'standard');
12 - L(3)=Link('revolute' , 'd', DH(3,2), 'a', DH(3,3), 'alpha',DH(3,4),'standard');
13 - %L(3)=Link(DH(3,1:4),'standard');
14 - %Following programs simulate the manipulator for the given joint variables
15 - my3r=SerialLink(L,'name','1st 3r');
16 - %Forward Kinematics
17 - Ti= fkine(my3r,qi);
18 - %figure of RRR manipulator in 3D
19 - plot(my3r,qi),view(50,40)
20 - %figure of RRR manipulator in 3D
21 - plot(my3r,qi),view(3);
22

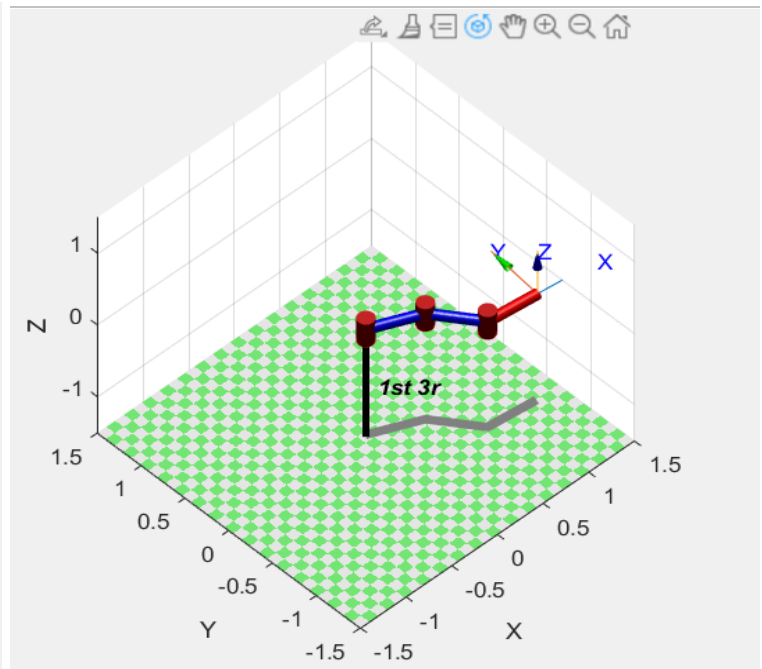
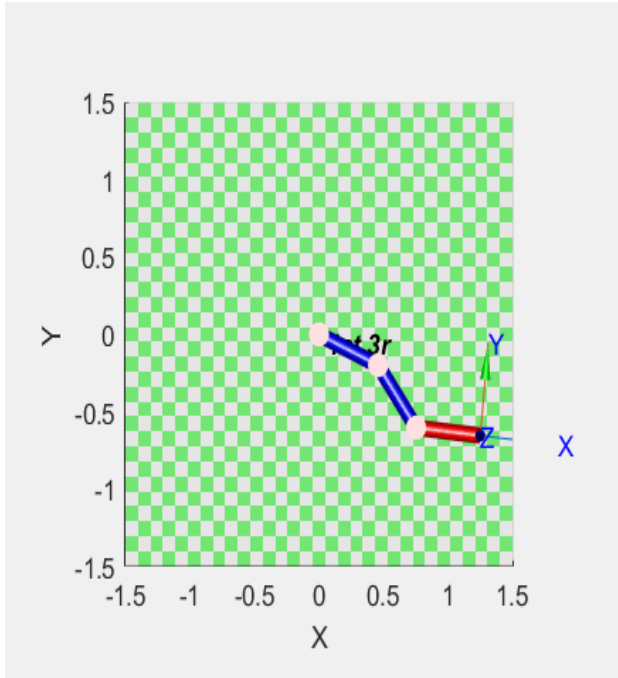
```

Using the above code and providing input arguments as given joint variable configuration

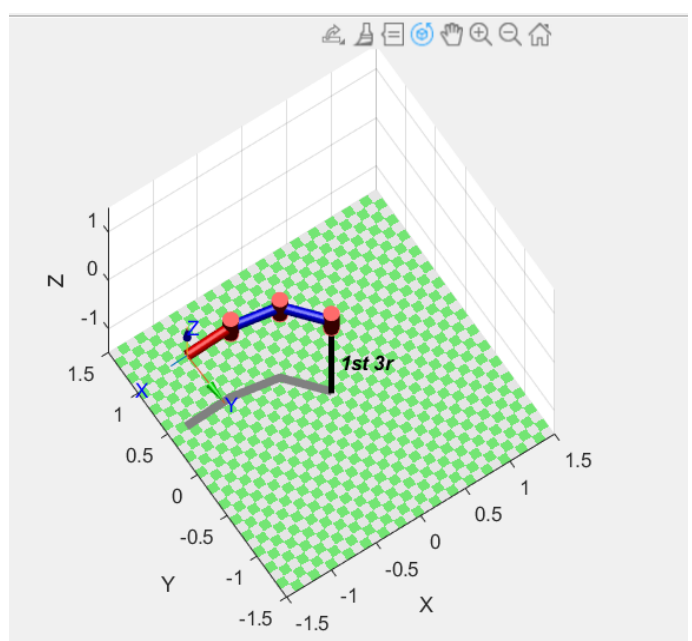
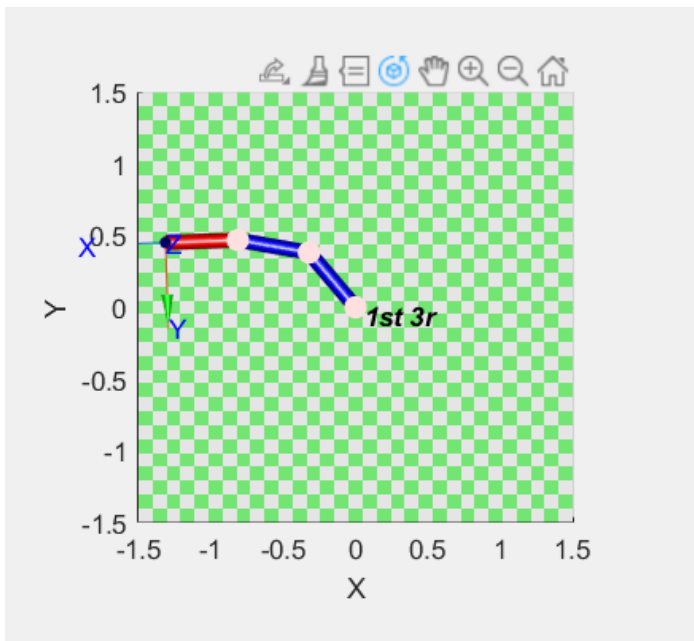
1. Joint configurations [0 90 -90]



2. Joint configurations [-23.6 -30.3 48.0]



3. Joint configuration [130 40 12.0]



Inverse Manipulator Kinematics

Programming exercise (part 4)

- For calculating the joint variables we need to compare the given transformation matrix (3×3) of the wrist frame relative to the base frame with the Forward Kinematics Transformation matrix of the Wrist relative to the base with angles Theta1, Theta2, Theta3 using the Standard DH method
- Following is the Forward Kinematics of 3 R planar Robot
- The following is Symbolic Forward Transformation Matrix of Wrist Frame relative to Base Frame used to compute the joint variables using the FUNCTIONS like INVKIN and SOLVE

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\chapt3kine.m
Command Window

Twb =

'Transformation matrix of wrist relative to base frame for 3R Planar robot'

Twb =

[cos(theta1 + theta2 + theta3), -sin(theta1 + theta2 + theta3), 0, a2*cos(theta1 + theta2) + a1*cos(theta1) + a3*cos(theta1 + theta2 + theta3)]
[sin(theta1 + theta2 + theta3), cos(theta1 + theta2 + theta3), 0, a2*sin(theta1 + theta2) + a1*sin(theta1) + a3*sin(theta1 + theta2 + theta3)]
[0, 0, 0, 1]
[0, 0, 0, 0]

```

1. FUNCTION -INVKIN

- FUNCTION 'INVKIN' is defined for joint computing variables (theta1, theta2, theta3) by giving input argument as the transformation matrix (3×3) of wrist frame relative to the base frame for 3 R planar manipulator and current joint variables which is the latest joint variables for the previous configuration for the proper position orientation (pose) of end-effector or wrist frame
- Here in this exercise, it is given $L=a1=a2=a3=0.5$, so substituting the given link length parameters and comparing the input argument transformation matrix with the values of the above Transformation matrix of the wrist relative to the base frame.
- The matrix values to calculate the theta1, theta2, theta3 (1,1) (2,1) (1,3) and (2,3) are used and the nonlinear equations are solved to get joint variables
- Comparing the solutions of joint variables for the given goal frame with the current joint variables to compute whether the particular solution is Near or Far.
- The Current value for 1st Transformation are [0 0 0] to check with the zero configuration the current values to check the near and far of the next configuration are previous configuration joint angles to determine the comment and print whether the solution is near or far compare to the current position
- Checked the Inverse joint variables with function WHERE - verify

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\INVKIN.m*
+8  TMULT.m  TINVERT.m  ITOU.m  INVKIN.m*  Untitled  Chapter4.m  CHECKSOLUTIONS.m  SOLVE.m
1  %%%FUNCTION - INVKIN %%%
2  function JointVariables=INVKIN(Twb,CurrentJointAngle)
3  -      anglephi=arctan2(Twb(2,1),Twb(1,1));
4  -      Xcoordinate=Twb(1,3);
5  -      Ycoordinate=Twb(2,3);
6  -      syms theta1 theta2 theta3;
7  -      eq1=theta1+theta2+theta3==anglephi;
8  -      eq2=0.5*(cos(theta1)+cos(theta1+theta2)+cos(theta1+theta2+theta3))==Xcoordinate;
9  -      eq3=0.5*(sin(theta1)+sin(theta1+theta2)+sin(theta1+theta2+theta3))==Ycoordinate;
10 -      [theta1 ,theta2 ,theta3]= solve(eq1,eq2,eq3,theta1,theta2,theta3);
11 -      JointVariables=[theta1 theta2 theta3];
12 -      JointVariablesr=double(JointVariables);
13 -      JointVariables1=JointVariablesr*180/pi;
14 -      %%%Function CHECKSOLUTIONS for Range of degrees%%
15 -      JointVariables=CHECKSOLUTIONS(JointVariables1)
16 -      %to check whether the soltuin is NEAR or FAR compare the Computed
17 -      %angles with the current value of Joint Variables
18 -      Sum1 =[abs(JointVariables(1,1)-CurrentJointAngle(1,1))
19 -            abs(JointVariables(1,2)-CurrentJointAngle(1,2))
20 -            abs(JointVariables(1,3)-CurrentJointAngle(1,3));
21 -      Sum2 =[abs(JointVariables(2,1)-CurrentJointAngle(1,1))
22 -            abs(JointVariables(2,2)-CurrentJointAngle(1,2))
23 -            abs(JointVariables(2,3)-CurrentJointAngle(1,3));
24 -      if (Sum1>Sum2)
25 -          [JointVariables]='True Near'
26 -      elseif (Sum1<Sum2)
27 -          [JointVariables]='True Far'
28 -      %%%Update currentJointAngle with JointVariables to check for new
29 -      %%%Near and far
30 -
31 - end

```

The above program [a b c] corresponds to three joint variables = Theta1, Theta2, and Theta3.

We need to check the values whether it lies in the given range of -170 to +170 degrees, and if the values are not in the range we should ignore those values. If the negative values of degrees are the same as the positive values of degrees we will consider those values if they lie in the range of Joint Variables.

The following code checks and manipulates the corresponding values of joint variables

FUNCTION - CHECK SOLUTIONS


```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\CHECKSOLUTIONS.m*
+7 Chapter3Program.m x TMULT.m x TINVERT.m x ITOU.m x INVKIN.m x Untitled x Chapter4.m x CHECKSOLUTIONS.m* x SOL
1 function JOINTVARIABLES=CHECKSOLUTIONS(jointvariables)
2     if (jointvariables>-170&&jointvariables<170)
3         RangeJointVariables=jointvariables;
4         %% -170<Range<170
5         %No Solution values
6         elseif (jointvariables>170&&jointvariables<190)|| (jointvariables<-170&&jointvariables>-190)
7             RangeJointVariables=No SOLUTION ;
8         %2nd Quadrant Values
9         elseif (jointvariables<-170)
10            RangeJointVariables=jointvariables+360;
11        %3rd and 4th Quadrant Values
12        elseif (jointvariables>190)
13            RangeJointVariables=jointvariables-360;
14    end
15 end

```

2. FUNCTION - SOLVE

- FUNCTION 'SOLVE' computes Transformation matrix of wrist frame relative to base frame with input arguments in tuple of {S} relative to {B} frame , {T} relative to {S} frame, and {T} relative to {w} frame. Using the defined functions of TINVERT and TMULT we can find the Transformation matrix of the wrist frame relative to the base frame

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\SOLVE.m*
+6 Chapter3Eqn.m x Chapter3Program.m x TMULT.m x TINVERT.m x ITOU.m x INVKIN.m
1 %%%Matlab Programming
2 %%%Using Transformation compounding arithmetic
3 %%%Tts=Tbs*Twb*Ttw
4 %%% As per Transform Equations- Twb=Tsb*Tts*Twt
5 %As the values of
6 function Twb=SOLVE(srelb,trels,trelw)
7 %%%FUNCTION - UTOI
8     Tsb=UTOI(srelb);
9     Tts=UTOI(trels);
10    Ttw=UTOI(trelw);
11    %%%FUNCTION - TINVERT
12    Twt=TINVERT(Ttw);
13    %%%FUNCTION - TMULT
14    Twb=TMULT(TMULT(Tsb,Tts),Twt);
15 end

```

3. Calculate Joint Variables

- The following code uses the given tuple of the tool relative to wrist and station relative to the base frame, there are 4 different configurations given for the goal frame, calculating the joint variables with considering the range of the joint angles.

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapter4.m*
+8  TMULT.m  TINVERT.m  ITOU.m  INVKIN.m  Untitled  Chapter4.m*  CHECKSOLU
1   %%Matlab Programming
2   %Given User Form tuple of Transformation Matrix
3   %Tool with respect to wrist and Station frame respect to Base
4   trelw=[0.1 0.2 30];
5   srelb=[-0.1 0.3 0];
6   %%As per given question for 1st Goal Frame is
7   trels1=[0 0 -90];
8   %for the 1st joint variables current joint variables is [0 0 0]
9   CurrentJointVariables1=[0 0 0];
10  %%Calling defined function to calculate Wrist relative to Base Frame
11  %%FUNCTION - SOLVE
12  Twb1=SOLVE(srelb,trels,trelw);
13  JointVariables1='Joint Variables that are in Range'
14  JointVariables1=INVKIN(Twb1,CurrentJointVariables1);
15

```

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapter4.m
+8  TMULT.m  TINVERT.m  ITOU.m  INVKIN.m  Untitled  Chapter4.m  CHECKSOLUTIO
22  %Matlab Programming
23  %Given User Form tuple of Transformation Matrix
24  %Tool with respect to wrist and Station frame respect to Base
25  trelw=[0.1 0.2 30];
26  srelb=[-0.1 0.3 0];
27  %%As per given question for 2ndGoal Frame is
28  trels2=[0.6 -0.3 45];
29  CurrentJointVariables2=JOINTVARIABLES1;
30  %%Calling defined function to calculate Wrist relative to Base Frame
31  %%FUNCTION - SOLVE
32  Twb2=SOLVE(srelb,trels,trelw)
33  JointVariables2='Joint Variables that are in Range'
34  JointVariables2=INVKIN(Twb2,CurrentJointVariables2)
35

```

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapter4.m
+8  TMULT.m  TINVERT.m  ITOU.m  INVKIN.m  Untitled  Chapter4.m  CHECKSOLUTIO
37
38  %%
39  %Matlab Programming
40  %Given User Form tuple of Transformation Matrix
41  %Tool with respect to wrist and Station frame respect to Base
42  trelw=[0.1 0.2 30];
43  srelb=[-0.1 0.3 0];
44  %%As per given question for 3rdGoal Frame is
45  trels3=[-0.4 0.3 120];
46  CurrentJointVariables3=JOINTVARIABLES2;
47  %%Calling defined function to calculate Wrist relative to Base Frame
48  %%FUNCTION - SOLVE
49  Twb3=SOLVE(srelb,trels,trelw)
50  JointVariables3='Joint Variables that are in Range'
51  JointVariables3=INVKIN(Twb3,CurrentJointVariables3)
52

```

```

Editor - C:\Users\somaw\Desktop\FOR PROJECT\Chapter4.m
+8  TMULT.m  TINVERT.m  ITOU.m  INVKIN.m  Untitled  Chapter4.m  CHECKSOLUT
54  %%%
55  %%%Matlab Programming
56  %Given User Form tuple of Transformation Matrix
57  %Tool with respect to wrist and Station frame respect to Base
58  -  trelw=[0.1 0.2 30];
59  -  srelb=[-0.1 0.3 0];
60  %%%As per given question for 4th Goal Frame is
61  -  trels4=[0.8 1.4 30];
62  -  CurrentJointVariables4=JOINTVARIABLES3;
63  %%%Calling defined function to calculate Wrist relative to Base Frame
64  %%%FUNCTION - SOLVE
65  -  Twb4=SOLVE(srelb,trels,trelw)
66  -  JointVariables4='Joint Variables that are in Range'
67  -  JointVariables4=INVKIN(Twb4,CurrentJointVariables4);
68

```

```

Editor - Untitled3*
Command Window
Twb1 =

    -0.5000    0.8660   -0.2232
    -0.8660   -0.5000    0.4866
         0         0    1.0000

JointVariables1 =

    'Joint Variables that are in Range'

JointVariables1 =

    65.2579    46.1462   128.5958

JointVariables1 =

    'True  Near'

```

```

Editor - Untitled3*
Command Window
Twb2 =

    0.9659   -0.2588    0.4552
    0.2588    0.9659   -0.2191
         0         0    1.0000

JointVariables2 =

    'Joint Variables that are in Range'

JointVariables2 =

   -164.0980   139.0764    40.0216

JointVariables2 =

    'True  Near'

```

```

Editor - Untitled3*
Command Window
Twb3 =

    0   -1.0000   -0.3000
    1.0000    0    0.5000
    0    0    1.0000

JointVariables3 =

    'Joint Variables that are in Range'

JointVariables3 =

    107.4576   145.0848  -162.5424

JointVariables3 =

    'True   Far'

JointVariables3 =

    'Joint Variables that are in Range'

JointVariables3 =

    -107.4576  -145.0848   -17.4576

JointVariables3 =

    'True   Near'

```

```

Command Window
Twb4 =

    1.0000    0    0.6000
    0    1.0000    1.5000
    0    0    1.0000

ans =

    logical

    0

JointVariables4 =

    1.0e+02 *

    0.7330 - 0.7829i    0.0000 + 1.5657i   -1.6330 - 0.7829i
    0.7330 + 0.7829i    0.0000 - 1.5657i   -1.6330 + 0.7829i

```

Therefore No Solution for 4th Configuration as the Joint variables calculated are imaginary