Pham Thi Ngoc Thao - 20215435 - Master Program – University of Ulsan

Energy Conversion Circuit Laboratory (ECCL) | UoU #7-115-1 | Email: [ptnthao1776@gmail.com](mailto:ptnthao1776@gmail.com)

**Homework #4**

*Mechanics and Control of Robot Manipulators*

In HW#3, I had some errors, below is the fixed code.

Find 8 sets of solutions from Inverse Kinematics with the .

clear all;

clc;

%DH parameter

alpha = [0 -90 0 -90 90 -90] ; %link twist

a = [0 0 1 0.3 0 0]; %link length

d = [0 0 0.5 1 0 0]; %link offset

theta = [30 30 30 30 30 30]; % joint variable

P6\_T = [0 0 0.3 1];

%Apply forward kinematics joints

T = [];

for n = 1:6

matT = [cosd(theta(n)) -sind(theta(n)) 0 a(n);

sind(theta(n))\*cosd(alpha(n)) cosd(theta(n))\*cosd(alpha(n)) -sind(alpha(n)) -sind(alpha(n))\*d(n);

sind(theta(n))\*sind(alpha(n)) cosd(theta(n))\*sind(alpha(n)) cosd(alpha(n)) cosd(alpha(n))\*d(n);

0 0 0 1];

T = [T; {matT}];

end

T0\_6 = T{1}\*T{2}\*T{3}\*T{4}\*T{5}\*T{6}

P0\_T = T0\_6.\*P6\_T

%---------------------------------------------------------------------------------

%calulate T0\_T, T06 with the code in a.

%problem b: 8 set parameter

px = T0\_6(1,4);

py = T0\_6(2,4);

pz = T0\_6(3,4);

r = T0\_6(1:3,1:3);

th = zeros(6,8);

th(1,1:4) = atan2d(py,px) - atan2d(d(3), sqrt(px^2 + py^2 -d(3)^2));

th(1,5:8) = atan2d(py,px) - atan2d(d(3), -sqrt(px^2 + py^2 -d(3)^2));

K = (px^2 + py^2 + pz^2 - a(3)^2 - a(4)^2 - d(3)^2 - d(4)^2)/(2\*a(3));

th(3,[1,2,5,6]) = atan2d(a(4),d(4)) - atan2d(K, sqrt(a(4)^2 + d(4)^2 - K^2));

th(3,[3,4,7,8]) = atan2d(a(4),d(4)) - atan2d(K, -sqrt(a(4)^2 + d(4)^2 - K^2));

th23 = atan2d((-a(4)-a(3)\*cosd(th(3,:)))\*pz + (cosd(th(1,:))\*px + sind(th(1,:))\*py).\*(d(4)-a(3)\*sind(th(3,:))),(a(3)\*sind(th(3,:))-d(4))\*pz+(a(4)+a(3)\*cosd(th(3,:))).\*(cosd(th(1,:))\*px + sind(th(1,:))\*py));

th(2,:) = th23 - th(3,:);

th(4,:) = atan2d(-r(1,3)\*sind(th(1,:))+r(2,3)\*cosd(th(1,:)),-r(1,3)\*cosd(th(1,:)).\*cosd(th23)-r(2,3)\*sind(th(1,:)).\*cosd(th23) + r(3,3)\*sind(th23));

s5 = -r(1,3)\*(cosd(th(1,:)).\*cosd(th23).\*cosd(th(4,:)) + sind(th(1,:)).\*sind(th(4,:))) - r(2,3)\*(sind(th(1,:)).\*cosd(th23).\*cosd(th(4,:)) - cosd(th(1,:)).\*sind(th(4,:))) + r(3,3)\*sind(th23).\*cosd(th(4,:));

c5 = -r(1,3)\*cosd(th(1,:)).\*sind(th23) - r(2,3)\*sind(th(1,:)).\*sind(th23) - r(3,3)\*cosd(th23);

th(5,:) = atan2d(s5,c5);

s6 = -r(1,1)\*(cosd(th(1,:)).\*cosd(th23).\*sind(th(4,:)) - sind(th(1,:)).\*cosd(th(4,:))) - r(2,1)\*(sind(th(1,:)).\*cosd(th23).\*sind(th(4,:)) + cosd(th(1,:)).\*cosd(th(4,:))) + r(3,1)\*sind(th23).\*sind(th(4,:));

c6 = r(1,1)\*((cosd(th(1,:)).\*cosd(th23).\*cosd(th(4,:)) + sind(th(1,:)).\*sind(th(4,:))).\*cosd(th(5,:)) - cosd(th(1,:)).\*sind(th23).\*sind(th(5,:))) + r(2,1)\*((sind(th(1,:)).\*cosd(th23).\*cosd(th(4,:)) - cosd(th(1,:)).\*sind(th(4,:))).\*cosd(th(5,:)) - sind(th(1,:)).\*sind(th23).\*sind(th(5,:))) - r(3,1)\*(sind(th23).\*cosd(th(4,:)).\*cosd(th(5,:)) + cosd(th23).\*sind(th(5,:)));

th(6,:) = atan2d(s6,c6);

th(4,[2 4 6 8]) = th(4,[2 4 6 8]) + 180;

th(5,[2 4 6 8]) = -th(5,[2 4 6 8]);

th(6,[2 4 6 8]) = th(6,[2 4 6 8]) + 180;

th

The result is

Text, letter

Description automatically generated

**At that instance, all joint velocities are 0.1 rad / sec with the robot configuration of prob. 1 of HW3**

If possible, Write the Matlab Program to do next problems as

1. Find the linear and angular velocities of the tool through velocity propagation

Linear and angular velocities of the tool can be calculated as below

Matlab code

clear all;

clc;

%DH parameter

alpha = [0 -pi/2 0 -pi/2 pi/2 -pi/2] ; %link twist

a = [0 0 1 0.3 0 0]; %link length

d = [0 0 0.5 1 0 0]; %link offset

theta = [pi/6 pi/6 pi/6 pi/6 pi/6 pi/6]; % joint variable

omega=[0;0;0.1];

P6\_T = [0 0 0.3 1];

%Apply forward kinematics joints

T = [];

R = [];

for n = 1:6

matT = [cos(theta(n)) -sin(theta(n)) 0 a(n);

sind(theta(n))\*cos(alpha(n)) cos(theta(n))\*cos(alpha(n)) -sin(alpha(n)) -sin(alpha(n))\*d(n);

sin(theta(n))\*sin(alpha(n)) cos(theta(n))\*sin(alpha(n)) cos(alpha(n)) cos(alpha(n))\*d(n);

0 0 0 1];

T = [T; {matT}];

R = [R; {matT(1:3,1:3)}];

end

R = [R; {[1 0 0; 0 1 0; 0 0 1]}];

P\_0\_1 = [0; 0; 0];

P\_1\_2 = [0;0;0];

P\_2\_3 = [a(3);0;d(3)];

P\_3\_4=[a(4);d(4);0];

P\_4\_5=[ 0;0;0];

P\_5\_6=[0;0;0];

P\_6\_t=[0;0;0.3];

w\_0\_0=[0;0;0];

v\_0\_0=[0;0;0];

w\_1\_1=R{1}'\*w\_0\_0+[0;0;0.1];

v\_1\_1=R{1}'\*(v\_0\_0+cross(w\_0\_0,P\_0\_1));

w\_2\_2=R{2}'\*w\_1\_1+[0;0;0.1];

v\_2\_2=R{2}'\*(v\_1\_1+cross(w\_1\_1,P\_1\_2));

w\_3\_3=R{3}'\*w\_2\_2+[0;0;0.1];

v\_3\_3=R{3}'\*(v\_2\_2+cross(w\_2\_2,P\_2\_3));

w\_4\_4=R{4}'\*w\_3\_3+[0;0;0.1];

v\_4\_4=R{4}'\*(v\_3\_3+cross(w\_3\_3,P\_3\_4));

w\_5\_5=R{5}'\*w\_4\_4+[0;0;0.1];

v\_5\_5=R{5}'\*(v\_4\_4+cross(w\_4\_4,P\_4\_5));

w\_6\_6=R{6}'\*w\_5\_5+[0;0;0.1];

v\_6\_6=R{6}'\*(v\_5\_5+cross(w\_5\_5,P\_5\_6));

w\_t\_t=R{7}'\*w\_6\_6+0;

v\_t\_t=R{7}'\*(v\_6\_6+cross(w\_6\_6,P\_6\_t));

w\_0\_t=R{1}\*R{2}\*R{3}\*R{4}\*R{5}\*R{6}\*R{7}\*w\_6\_6

v\_0\_t=R{1}\*R{2}\*R{3}\*R{4}\*R{5}\*R{6}\*R{7}\*v\_t\_t

V\_0=[v\_0\_t;w\_0\_t]

The results are:

Text

Description automatically generated

1. Find the Jacobian at that instant.

The matrix defining the frame {i} in reference to the base frame is calculated as below:

 where 

Then the Jacobian matrix is calculated as below:



Matlab code:

T0\_6 = T{1}\*T{2}\*T{3}\*T{4}\*T{5}\*T{6}

P0\_T = T0\_6.\*P6\_T

%%%%%%%%%%%%% Jacobian

P\_0\_t=P0\_T(1:3, 4);

k\_0\_1=T{1}(1:3,3) ;

P0\_1=T{1}(1:3,4) ;

J1=[cross(k\_0\_1,(P\_0\_t-P0\_1));k\_0\_1]

T0\_2=T{1}\*T{2}

k\_0\_2=T0\_2(1:3,3) ;

P0\_2=T0\_2(1:3,4) ;

J2=[cross(k\_0\_2,(P\_0\_t-P0\_2));k\_0\_2]

T0\_3=T{1}\*T{2}\*T{3}

k\_0\_3=T0\_3(1:3,3) ;

P0\_3=T0\_3(1:3,4) ;

J3=[cross(k\_0\_3,(P\_0\_t-P0\_3));k\_0\_3]

T0\_4=T{1}\*T{2}\*T{3}\*T{4}

k\_0\_4=T0\_4(1:3,3) ;

P0\_4=T0\_4(1:3,4) ;

J4=[cross(k\_0\_4,(P\_0\_t-P0\_4));k\_0\_4]

%

T0\_5=T{1}\*T{2}\*T{3}\*T{4}\*T{5}

k\_0\_5=T0\_5(1:3,3) ;

P0\_5=T0\_5(1:3,4) ;

J5=[cross(k\_0\_5,(P\_0\_t-P0\_5));k\_0\_5]

T0\_6=T{1}\*T{2}\*T{3}\*T{4}\*T{5}\*T{6}

k\_0\_6=T0\_6(1:3,3) ;

P0\_6=T0\_6(1:3,4) ;

J6=[cross(k\_0\_6,(P\_0\_t-P0\_6));k\_0\_6]

J=[J1 J2 J3 J4 J5 J6]

The result of Jacobian

Text, letter

Description automatically generated

1. With the inverse of Jacobian and the obtained results, do velocity inverse  
   kinematics to find the joint velocities.

Inverse velocity is calculated as below:

Malab code:

%%%%%%%%%%%%%%%%%%%%%%%%%Inverse velocity%

th\_dot=inv(J)\*V\_0

The result is

Text

Description automatically generated