Energy Conversion Circuit Laboratory (ECCL) | UoU #7-115-1 | Email: 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 ${}^{0}_{T}T$.

```
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

th =

30.0000	30.0000	30.0000	30.0000	-3.3985	-3.3985	-3.3985	-3.3985
32.4688	32.4688	-213.7730	-213.7730	41.9230	41.9230	-219.2126	-219.2126
30.0000	30.0000	183.3985	183.3985	30.0000	30.0000	183.3985	183.3985
32.3101	212.3101	16.1525	196.1525	-53.6497	126.3503	-22.0084	157.9916
27.8868	-27.8868	116.0187	-116.0187	23.6669	-23.6669	120.3743	-120.3743
27.3607	207.3607	63.8057	243.8057	105.7857	285.7857	43.0166	223.0166

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

$$ {}^{i+1}\omega_{i+1} = {}^{i+1}_{i}R\,{}^{i}\omega_{i} + \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} $$

$$ {}^{i+1}v_{i+1} = {}^{i+1}_{i}R\left({}^{i}v_{i} + {}^{i}\omega_{i} \times {}^{i}P_{i+1}\right) $$

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:

```
V_0 =

   -0.3452
    0.0535
    0.0593
   -0.2824
    0.2685
   -0.0518
```

2. Find the Jacobian at that instant.

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

$$ {}^{0}_{i}T = {}^{0}_{1}T\,{}^{1}_{2}T \ldots {}^{i-1}_{i}T = \begin{bmatrix} {}^{0}_{i}R & {}^{0}_{i}P \\ 0 & 1 \end{bmatrix} \quad \text{where } {}^{0}_{i}R = \begin{bmatrix} {}^{0}_{i}s & {}^{0}_{i}j & {}^{0}_{i}k \end{bmatrix} $$

Then the Jacobian matrix is calculated as below:

$$ J_{\theta i} = \begin{bmatrix} {}^{0}_{i}k \times ({}^{0}_{T}P - {}^{0}_{i}P) \\ {}^{0}_{i}k \end{bmatrix} $$

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

```
J =

   -0.4351   -0.9806   -0.5476         0         0         0
   -0.0563   -0.5661   -0.3161         0         0         0
         0   -0.1687    0.4847         0         0         0
         0   -0.5000   -0.5000   -0.7500   -0.1102   -1.0541
         0    0.8660    0.8660   -0.0079    0.7534    0.2067
    1.0000    0.0000    0.0000   -0.5000   -0.2205   -0.2421
```

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

Inverse velocity is calculated as below:

$$\dot{\Theta} = {}^{0}J^{-1}(\Theta)\ {}^{0}V$$

Malab code:

```
%%%%%%%%%%%%%%%%%%%%%%%%%Inverse velocity%
th_dot=inv(J)*V_0
```

The result is

```
th_dot =

    1.2972
   -0.2443
    0.0372
    3.1459
    1.1749
   -1.9952
```