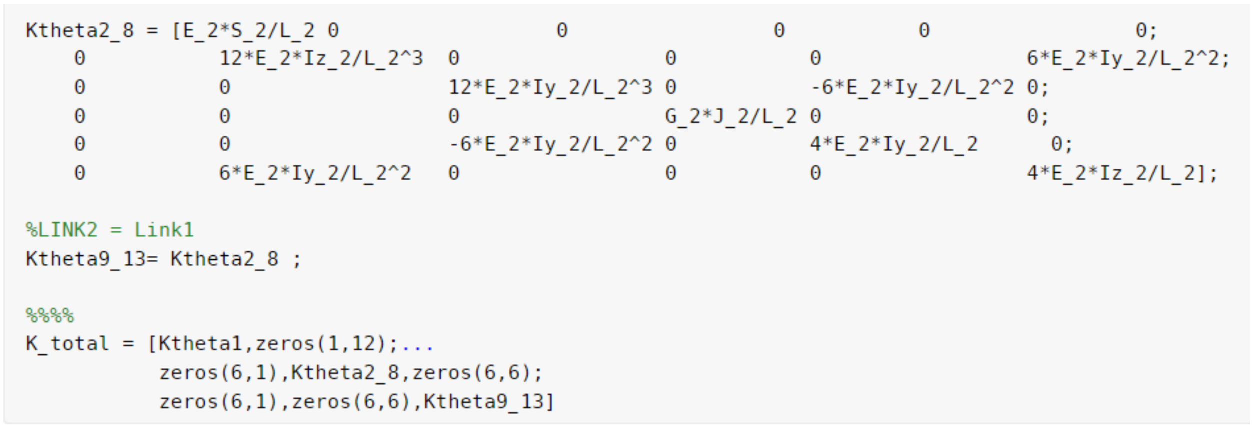
Jose Corona

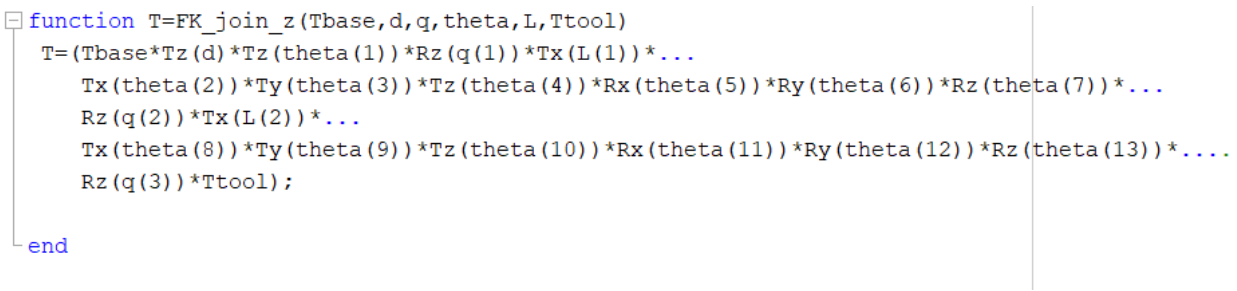
Home Work 2

K theta is 13x13, for a leg chain is



```
Ktheta2_8 = [E_2*S_2/L_2 0                  0                  0           0                  0;
    0           12*E_2*Iz_2/L_2^3  0                  0           0                  6*E_2*Iy_2/L_2^2;
    0           0                  12*E_2*Iy_2/L_2^3  0           -6*E_2*Iy_2/L_2^2  0;
    0           0                  0                  G_2*J_2/L_2 0                  0;
    0           0                  -6*E_2*Iy_2/L_2^2  0           4*E_2*Iy_2/L_2      0;
    0           6*E_2*Iy_2/L_2^2   0                  0           0                  4*E_2*Iz_2/L_2];

%LINK2 = Link1
Ktheta9_13= Ktheta2_8 ;

%%%%
K_total = [Ktheta1,zeros(1,12);...
           zeros(6,1),Ktheta2_8,zeros(6,6);
           zeros(6,1),zeros(6,6),Ktheta9_13]
```

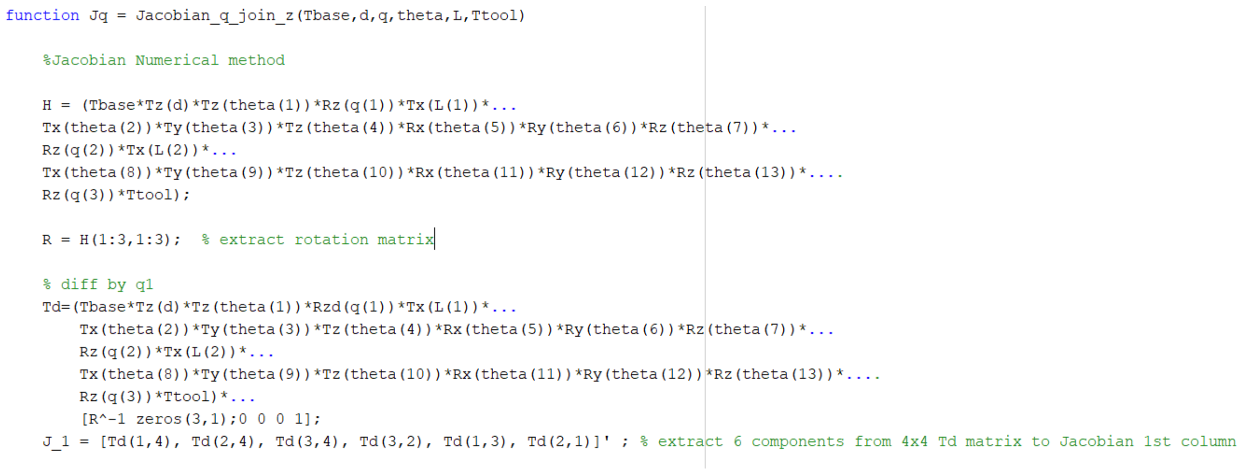
The FK of the robot was implemented



```
function T=FK_join_z(Tbase,d,q,theta,L,Ttool)
 T=(Tbase*Tz(d)*Tz(theta(1))*Rz(q(1))*Tx(L(1))*...
    Tx(theta(2))*Ty(theta(3))*Tz(theta(4))*Rx(theta(5))*Ry(theta(6))*Rz(theta(7))*...
    Rz(q(2))*Tx(L(2))*...
    Tx(theta(8))*Ty(theta(9))*Tz(theta(10))*Rx(theta(11))*Ry(theta(12))*Rz(theta(13))*....
    Rz(q(3))*Ttool);

end
```

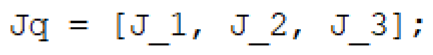
Where for each leg chain just have to enter the correspond Tbase, transform of the axis from global axis to the local leg chain axis.

The jacobian for the passive joints was calculated, for each chain leg is need to calculate the righ Tbase transform.



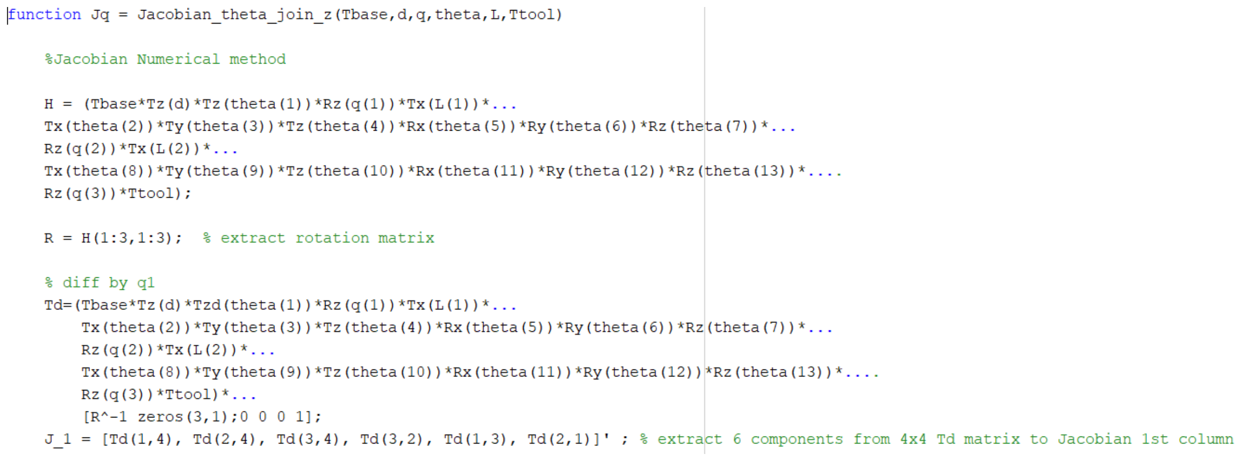
```
function Jq = Jacobian_q_join_z(Tbase,d,q,theta,L,Ttool)

    %Jacobian Numerical method

    H = (Tbase*Tz(d)*Tz(theta(1))*Rz(q(1))*Tx(L(1))*...
    Tx(theta(2))*Ty(theta(3))*Tz(theta(4))*Rx(theta(5))*Ry(theta(6))*Rz(theta(7))*...
    Rz(q(2))*Tx(L(2))*...
    Tx(theta(8))*Ty(theta(9))*Tz(theta(10))*Rx(theta(11))*Ry(theta(12))*Rz(theta(13))*....
    Rz(q(3))*Ttool);

    R = H(1:3,1:3);  % extract rotation matrix

    % diff by q1
    Td=(Tbase*Tz(d)*Tz(theta(1))*Rzd(q(1))*Tx(L(1))*...
        Tx(theta(2))*Ty(theta(3))*Tz(theta(4))*Rx(theta(5))*Ry(theta(6))*Rz(theta(7))*...
        Rz(q(2))*Tx(L(2))*...
        Tx(theta(8))*Ty(theta(9))*Tz(theta(10))*Rx(theta(11))*Ry(theta(12))*Rz(theta(13))*....
        Rz(q(3))*Ttool)*...
        [R^-1 zeros(3,1);0 0 0 1];
    J_1 = [Td(1,4), Td(2,4), Td(3,4), Td(3,2), Td(1,3), Td(2,1)]' ; % extract 6 components from 4x4 Td matrix to Jacobian 1st column
```

There where 3 passive joints. So the jacobian was 6*3

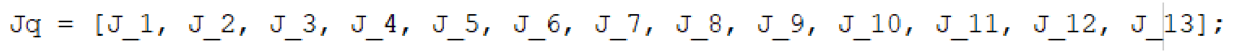


```
Jq = [J_1, J_2, J_3];
```

The jacobian for the virtual joints it was of 6*13

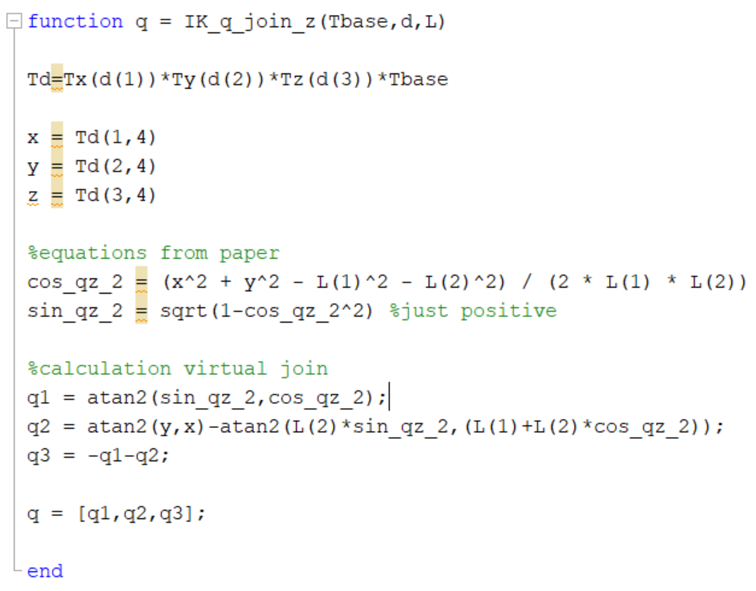


```
function Jq = Jacobian_theta_join_z(Tbase,d,q,theta,L,Ttool)

    %Jacobian Numerical method

    H = (Tbase*Tz(d)*Tz(theta(1))*Rz(q(1))*Tx(L(1))*...
    Tx(theta(2))*Ty(theta(3))*Tz(theta(4))*Rx(theta(5))*Ry(theta(6))*Rz(theta(7))*...
    Rz(q(2))*Tx(L(2))*...
    Tx(theta(8))*Ty(theta(9))*Tz(theta(10))*Rx(theta(11))*Ry(theta(12))*Rz(theta(13))*....
    Rz(q(3))*Ttool);

    R = H(1:3,1:3);  % extract rotation matrix

    % diff by q1
    Td=(Tbase*Tz(d)*Tzd(theta(1))*Rz(q(1))*Tx(L(1))*...
        Tx(theta(2))*Ty(theta(3))*Tz(theta(4))*Rx(theta(5))*Ry(theta(6))*Rz(theta(7))*...
        Rz(q(2))*Tx(L(2))*...
        Tx(theta(8))*Ty(theta(9))*Tz(theta(10))*Rx(theta(11))*Ry(theta(12))*Rz(theta(13))*....
        Rz(q(3))*Ttool)*...
        [R^-1 zeros(3,1);0 0 0 1];
    J_1 = [Td(1,4), Td(2,4), Td(3,4), Td(3,2), Td(1,3), Td(2,1)]' ; % extract 6 components from 4x4 Td matrix to Jacobian 1st column
```



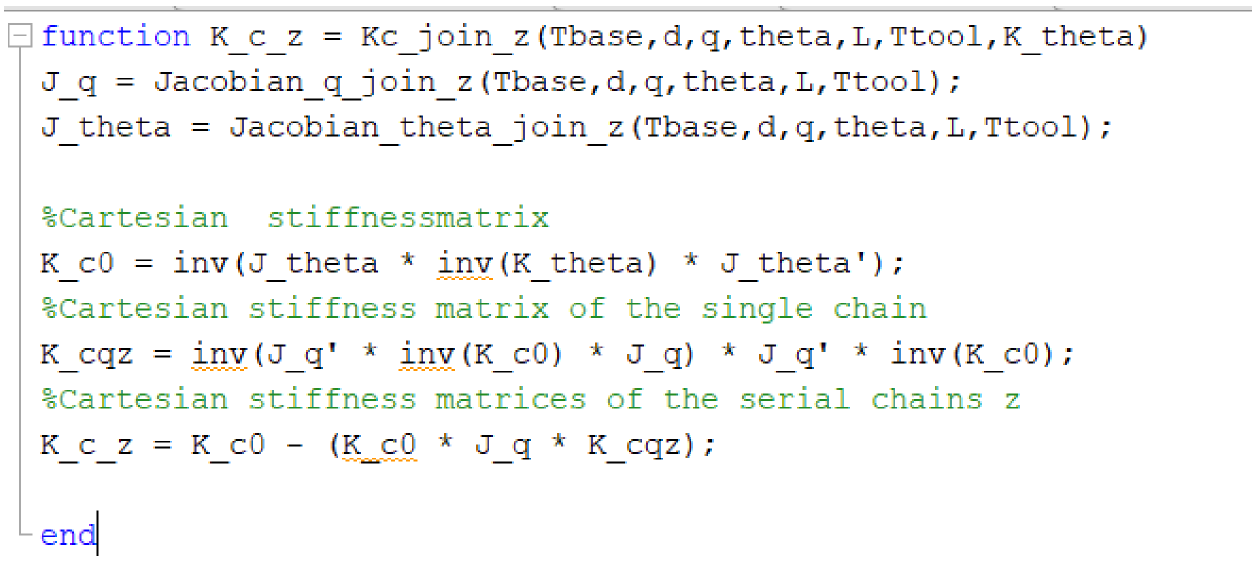
```
Jq = [J_1, J_2, J_3, J_4, J_5, J_6, J_7, J_8, J_9, J_10, J_11, J_12, J_13];
```

Also the IK was calculated.



```
function q = IK_q_join_z(Tbase,d,L)

Td=Tx(d(1))*Ty(d(2))*Tz(d(3))*Tbase

x = Td(1,4)
y = Td(2,4)
z = Td(3,4)

%equations from paper
cos_qz_2 = (x^2 + y^2 - L(1)^2 - L(2)^2) / (2 * L(1) * L(2))
sin_qz_2 = sqrt(1-cos_qz_2^2) %just positive

%calculation virtual join
q1 = atan2(sin_qz_2,cos_qz_2);
q2 = atan2(y,x)-atan2(L(2)*sin_qz_2,(L(1)+L(2)*cos_qz_2));
q3 = -q1-q2;

q = [q1,q2,q3];

end
```

And the Cartesian stiffness matrices of the serial chains z



```
function K_c_z = Kc_join_z(Tbase,d,q,theta,L,Ttool,K_theta)
J_q = Jacobian_q_join_z(Tbase,d,q,theta,L,Ttool);
J_theta = Jacobian_theta_join_z(Tbase,d,q,theta,L,Ttool);

%Cartesian  stiffnessmatrix
K_c0 = inv(J_theta * inv(K_theta) * J_theta');
%Cartesian stiffness matrix of the single chain
K_cqz = inv(J_q' * inv(K_c0) * J_q) * J_q' * inv(K_c0);
%Cartesian stiffness matrices of the serial chains z
K_c_z = K_c0 - (K_c0 * J_q * K_cqz);

end
```

Git Hub File:

https://github.com/Jose-R-Corona/AR-HomeTask2