Legends

(i) $S_{12} = \sin(\theta_1 + \theta_2)$

(ii) $C_{12} = \cos(\theta_1 + \theta_2)$

(iii) $S_{123} = \sin(\theta_1 + \theta_2 + \theta_3)$

(iv) $S_1 \rightarrow \sin\theta_1$; $C_1 \rightarrow \cos\theta_1$

(v) $S_{12-3} \rightarrow \sin(\theta_1 + \theta_2 - \theta_3)$

$S_{1-23} \rightarrow \sin(\theta_1 - \theta_2 + \theta_3)$

Similar for cosines.

$C_{1-2} \rightarrow \cos(\theta_1 - \theta_2)$.

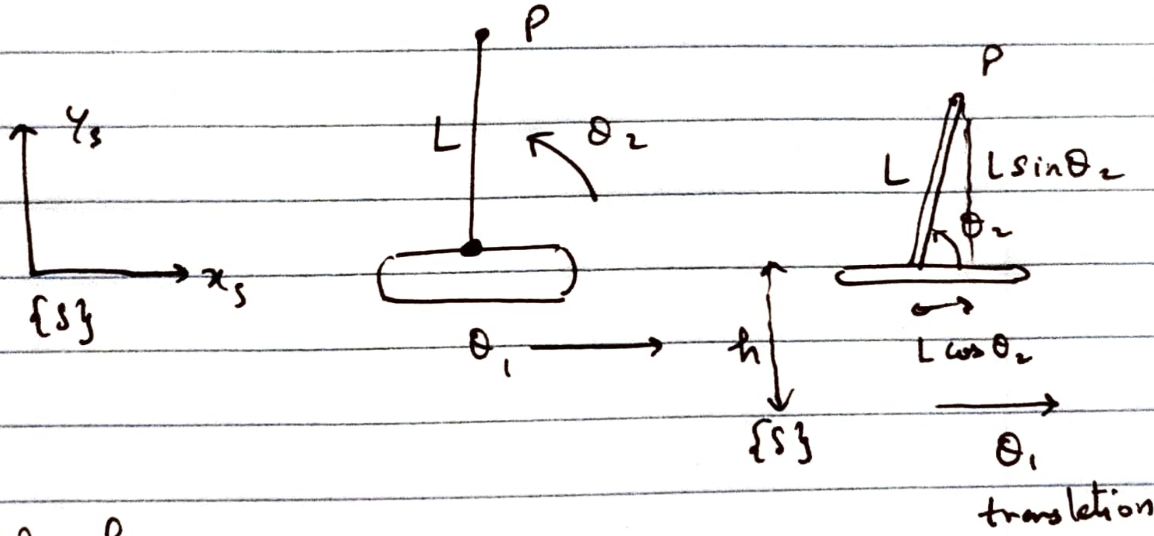
Nishant Bharali

906641308

Robotics & Automation

HW - 4

1.1.



We know,

$$\boxed{\dot{p} = J(\theta)\,\dot{\theta}} \quad \text{(velocity of the end-effector).}$$

Here,

$$p(t) = \begin{bmatrix} \theta_1(t) + L\cos\theta_2(t) \\ h + L\sin\theta_2(t) \end{bmatrix}$$

$$\therefore \quad \dot{p} = \frac{\partial p(t)}{\partial t} = \begin{bmatrix} \dot{\theta}_1 - L\dot{\theta}_2 \sin\theta_2 \\ \dot{\theta}_2 \cos\theta_2 \end{bmatrix}$$

Now, we separate $\dot{\theta}_1$ and $\dot{\theta}_2$ from the above equation :

$$\dot{p} = \begin{bmatrix} 1 & -L\sin\theta_2 \\ 0 & \cos\theta_2 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \Rightarrow \boxed{J(\theta) = \begin{bmatrix} 1 & -L\sin\theta_2 \\ 0 & \cos\theta_2 \end{bmatrix}}$$

1.21 We know:

$$[V_s] = \dot{T}\,T^{-1};$$

&, we know the robot has 3 joints :-

$$T = e^{[S_1]\theta_1}\, e^{[S_2]\theta_2}\, e^{[S_3]\theta_3} M$$

∴ We solve for $\dot{T} \times T^{-1}$; ($\theta(t)$ is a fn. of time)

$$\Rightarrow \dot{T} = [S_1]\dot{\theta}_1 e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} M \;+$$

$$e^{[S_1]\theta_1} [S_2]\dot{\theta}_2 e^{[S_2]\theta_2} e^{[S_3]\theta_3} M \;+$$

$$e^{[S_1]\theta} e^{[S_2]\theta_2} [S_3]\dot{\theta}_3 e^{[S_3]\theta_3} M \quad \text{—— (I)}$$

&,

$$T^{-1} = M^{-1} e^{-[S_3]\theta_3} e^{-[S_2]\theta_2} e^{-[S_1]\theta_1} \quad \text{—— (II)}$$

∴ Using eqns (I) & (II) :-

$[\mathcal{V}_s] = \dot{T}T^{-1}$

$$\Rightarrow [\mathcal{V}_s] = [S_1]\dot{\theta}_1 e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} M \cdot (M^{-1} e^{-[S_3]\theta_3} e^{-[S_2]\theta_2} e^{-[S_1]\theta_1})$$

$$+ e^{[S_1]\theta_1}[S_2]\dot{\theta}_2 e^{[S_2]\theta_2} e^{[S_3]\theta_3} M \cdot (M^{-1} e^{-[S_3]\theta_3} e^{-[S_2]\theta_2} e^{-[S_1]\theta_1})$$

$$+ e^{[S_1]\theta_1} e^{[S_2]\theta_2} [S_3]\dot{\theta}_3 e^{[S_3]\theta_3} M \cdot (M^{-1} e^{-[S_3]\theta_3} e^{-[S_2]\theta_2} e^{-[S_1]\theta_1})$$

$$= [S_1]\dot{\theta}_1 + e^{[S_1]\theta_1}[S_2]\dot{\theta}_2 e^{-[S_1]\theta_1} + e^{[S_1]\theta_1} e^{[S_2]\theta} [S]_3\dot{\theta}_3 \cdot e^{-[S_2]\theta_2} e^{-[S_1]\theta_1}$$

We know;

for a transformation matrix T;

$$T[S]T^{-1} = [Ad_T S]$$

Here;

$$\rightarrow e^{[S_n]\theta_n} = T_n \, ; \; e^{-[S_n]\theta_n} = \left(e^{[S_n]\theta_n}\right)^{-1} = T_n^{-1}$$

As they are some sort of transformation matrix.

Which follows the formula / simplification :

$$T[S]T^{-1} = [Ad_T S]$$

∴ Continuing simplification :-

$$\Rightarrow [V_s] = [S_1]\dot{\theta}_1 + \dot{\theta}_2 \left[e^{[S_1]\theta_1} [S_2] e^{-[S_1]\theta_1} \right]$$

$$+ \dot{\theta}_3 \left[e^{[S_1]\theta_1 + [S_2]\theta_2} [S_3] \left(e^{[S_1]\theta_1 + [S_2]\theta_2} \right)^{-1} \right]$$

Applying the properties of adjoint operators :-

$$\therefore [V_s] = [S_1]\dot{\theta}_1 + \dot{\theta}_2 \left[Ad_{e^{[S_1]\theta_1}} S_2 \right]$$

$$+ \dot{\theta}_3 \left[Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2}} S_3 \right]$$

$$\Rightarrow V_s = S_1\dot{\theta}_1 + Ad_{e^{[S_1]\theta_1}} S_2 \dot{\theta}_2 + Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2}} S_3 \dot{\theta}_3$$

$$\Rightarrow V_s = \begin{bmatrix} S_1 & Ad_{e^{[S_1]\theta_1}} S_2 & Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2}} S_3 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_2 \end{bmatrix}$$

$$\Rightarrow \boxed{V_s = J_s(\theta)\,\dot{\theta}}$$, where $J_s(\theta)$

(spatial Jacobian)

$\boxed{1.31}$ $V = J(\theta)\dot{\theta}$

Since, V is a six-dimensional vector, and robot has n joints.

The $J_s(\theta)$, matrix will have n columns, each column will be $Ad_{e^{[S_1]\theta_1} \cdots e^{[S_{i-1}]\theta_{i-1}}} S_i$

The Jacobian J will be investible if it has full rank, which means

$rank(J) = min(6, n)$.

Thus, it depends upon the fact if the columns of J are linearly independent or not and ~~how~~ how many of them are.

$\therefore$ For a square matrix ; $\underline{det(J) = 0}$

else, $det(JJ^T) = 0$.

→ If the determinant is not equal to 0, then's it's full rank and it is INVERTIBLE.

→ If determinant is equal to 0, then lost rank and is not INVERTIBLE

$\boxed{2.11}$

$$M = \begin{bmatrix} 1 & 0 & 0 & 3L \\ 0 & 1 & 0 & L \\ 0 & 0 & 1 & 0 \\ & 0 & & 1 \end{bmatrix}$$

Since, all axes of $\{s\}$ align with $\{b\}$'s axes including z-axis coming out of screen.

- Joint 1 :-

Revolute joint ;

$$\therefore \quad S_1 = \begin{bmatrix} \omega_s \\ -\omega_s \times q \end{bmatrix}$$

ω_s is a unit vector in +ve axis rotation direction

$$\omega_{s_1} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad q_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$S_1 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$$

- Joint 2 :-

$$\omega_{s_2} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \qquad q_2 = \begin{bmatrix} L & 0 & 0 \end{bmatrix}^T$$

$$S_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & L & 0 \end{bmatrix}^T$$

$$\begin{pmatrix}0\\0\\-1\end{pmatrix} \times \begin{pmatrix}2L\\L\\0\end{pmatrix} =$$

- Joint 3 :-

$$\omega_{s_3} = \begin{bmatrix}0\\0\\1\end{bmatrix} \qquad q_3 = \begin{bmatrix}2L\\L\\0\end{bmatrix}$$

$$\therefore S_3 = \begin{bmatrix}0\\0\\1\\L\\-2L\\0\end{bmatrix} = \begin{bmatrix}0 & 0 & 1 & L & -2L & 0\end{bmatrix}^T$$

$$\therefore T_{sb}(\theta) = e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} M$$

Using MATLAB, we find $T_{sb}(\theta)$:

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q2ForwardKinematics.m

Q2ForwardKinematics.m | Q2AdjointJacobian.m

```matlab
clc; close all;
syms L theta1 theta2 theta3 real

%Forward Kinematics for Question 2
M = [1 0 0 3*L; 0 1 0 L; 0 0 1 0; 0 0 0 1];
S1 = [0; 0; 1; 0; 0; 0];
S2 = [0; 0; 1; 0; L; 0];
S3 = [0; 0; 1; L; -2*L; 0];

T_final = expm(bracket(S1) * theta1) * expm(bracket(S2) * theta2) * expm(bracket(S3) * theta3) * M;
T1 = expm(bracket(S1) * theta1);
T2 = expm(bracket(S2) * theta2);
T3 = expm(bracket(S3) * theta3);
T_12 = T1*T2;

function S_matrix = bracket(S)
    S_matrix = [ 0 -S(3) S(2) S(4);
            S(3) 0 -S(1) S(5);
            -S(2) S(1) 0 S(6);
            0 0 0 0];
end
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> disp(simplify(expand(T_final)))
[cos(theta1 + theta2 + theta3), -sin(theta1 + theta2 + theta3), 0, L*(cos(theta1 + theta2 + theta3) + 3*cos(theta1 + theta2) - sin(theta1 + theta2) - cos(theta1))]
[sin(theta1 + theta2 + theta3),  cos(theta1 + theta2 + theta3), 0, L*(sin(theta1 + theta2 + theta3) + cos(theta1 + theta2) + 3*sin(theta1 + theta2) - sin(theta1))]
[                             0,                              0, 1,                                                                                              0]
[                             0,                              0, 0,                                                                                              1]

>>
```

2.2 Planar robot has 3 joints :

$\therefore \; J_s(\theta) = \left[S_1 \quad Ad_{e^{[S_1]\theta_1}} S_2 \quad Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2}} S_3 \right]$

& $V_s = J_s(\theta)\,\dot{\theta}$

Using MATLAB :-

$$e^{[S_1]\theta_1} = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$e^{[S_2]\theta_2} = \begin{bmatrix} c_2 & -s_2 & 0 & L(c_2 - 1) \\ s_2 & c_2 & 0 & Ls_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Now, for adjoint calculations :

$$Ad_{e^{[S_1]\theta_1}} = \begin{bmatrix} c_1 & -s_1 & 0 & 0 & 0 & 0 \\ s_1 & c_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_1 & -s_1 & 0 \\ 0 & 0 & 0 & s_1 & c_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q2AdjointJacobian.m

Q2ForwardKinematics.m | Q2AdjointJacobian.m

```
%VERIFICATION VALUES:

 %T_expm1 = T1 = expm(bracket(S1) * theta1)
  T_expm1 = [[cos(theta1), -sin(theta1), 0, 0]
[sin(theta1),  cos(theta1), 0, 0]
[          0,            0, 1, 0]
[          0,            0, 0, 1]];

  % T_expm12 = expm(bracket(S1) * theta1) * expm(bracket(S2) * theta2)
  T_expm12 = [[cos(theta1)*cos(theta2) - sin(theta1)*sin(theta2), - cos(theta1)*sin(theta2) - cos(theta2)*sin(theta1), 0, - cos(theta1)*(L - L*cos(theta2)) - L*sin(theta1)*sin(theta2)]
[cos(theta1)*sin(theta2) + cos(theta2)*sin(theta1),   cos(theta1)*cos(theta2) - sin(theta1)*sin(theta2), 0,   L*cos(theta1)*sin(theta2) - sin(theta1)*(L - L*cos(theta2))]
[                                                0,                                                   0, 1,                                                             0]
[                                                0,                                                   0, 0,                                                             1]];

  %Simplified version of the final Tsb(theta) forward kinematics
  T_sb = [[cos(theta1 + theta2 + theta3), -sin(theta1 + theta2 + theta3), 0, L*(cos(theta1 + theta2 + theta3) + 3*cos(theta1 + theta2) - sin(theta1 + theta2) - cos(theta1))]
[sin(theta1 + theta2 + theta3),  cos(theta1 + theta2 + theta3), 0, L*(sin(theta1 + theta2 + theta3) + cos(theta1 + theta2) + 3*sin(theta1 + theta2) - sin(theta1))]
[                            0,                              0, 1,                                                                                              0]
[                            0,                              0, 0,                                                                                              1]];
%Inverse of Tsb(theta) for calculation of Body Jacobian
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> disp(simplify(expand(T_expm12)))
[cos(theta1 + theta2), -sin(theta1 + theta2), 0, L*(cos(theta1 + theta2) - cos(theta1))]
[sin(theta1 + theta2),  cos(theta1 + theta2), 0, L*(sin(theta1 + theta2) - sin(theta1))]
[                   0,                     0, 1,                                      0]
[                   0,                     0, 0,                                      1]

fx >>
```

$\therefore$ Since, the calculations get complex for longer adjoints, using MATLAB :

$\Rightarrow Ad_{e^{[S_1]\theta_1}} S_2$

$$= \begin{bmatrix} 0 \\ 0 \\ 1 \\ -LS_1 \\ LC_1 \\ 0 \end{bmatrix}$$

Now for $Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2}} S_3 =$

$$\begin{bmatrix} 0 \\ 0 \\ 1 \\ 3L(C_2S_1 + C_1S_2) + L(C_1C_2 - S_1S_2) - LS_1 \\ 3L(S_1S_2 - C_1C_2) + L(C_1S_2 + S_1C_2) + LC_1 \\ 0 \end{bmatrix}$$

C: ▸ Users ▸ bhara ▸ OneDrive ▸ Documents ▸ MATLAB

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q2AdjointJacobian.m

Q2ForwardKinematics.m | Q2AdjointJacobian.m | +

```matlab
    [                                                    0,                                    0,

    %Adjoint calculations for all the respective Transformation matrices
    R1 = T_expm1(1:3, 1:3);
    p1 = T_expm1(1:3, 4);
    Adjoint_1 = [R1, zeros(3, 3);
        bracket(p1)*R1, R1;];

     R12 = T_expm12(1:3, 1:3);
    p12 = T_expm12(1:3, 4);
    Adjoint_2 = [R12, zeros(3, 3);
        bracket(p12)*R12, R12;];

     R_sb = T_sb(1:3, 1:3);
    p_sb = T_sb(1:3, 4);
    Adjoint_final = [R_sb, zeros(3, 3);
        bracket(p_sb)*R_sb, R_sb;];

    R_inverseQ2 = inverse_Q2(1:3, 1:3);
    p_inverseQ2 = inverse_Q2(1:3, 4);
    Adjoint_inverseQ2 = [R_inverseQ2, zeros(3, 3);
        bracket(p_inverseQ2)*R_inverseQ2, R_inverseQ2;];
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> disp(simplify(expand(Adjoint_2*S3)))
                                                           0
                                                           0
                                                           1
L*(cos(theta1 + theta2) + 3*sin(theta1 + theta2) - sin(theta1))
L*(sin(theta1 + theta2) - 3*cos(theta1 + theta2) + cos(theta1))
                                                           0
```

fx

Space Jacobian $J_s(\theta)$:

$$J_s(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & -LS_1 & 3L(C_2S_1 + C_1S_2) + L(C_1C_2 - S_1S_2) - LS_1 \\ 0 & LC_1 & 3L(S_1S_2 - C_1C_2) + L(C_1S_2 + C_2S_1) + LC_1 \\ 0 & 0 & 0 \end{bmatrix}$$

Space Jacobian $J_s(\theta)$:

$$J_s(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & -LS_1 & L(3S_{12} + C_{12} - S_1) \\ 0 & +LC_1 & L(-3C_{12} + S_{12} + C_1) \\ 0 & 0 & 0 \end{bmatrix}$$

$\boxed{2.3}$ **Body Jacobian :**

$$V_b = J_b(\theta)\,\dot{\theta}$$

$$\rightarrow J_b(\theta) = Ad_{T_{sb}^{-1}}\, J_s(\theta)$$

Using MATLAB :-

$$J_b(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ -L(S_{23} + C_3 - 3S_3) & -L(C_3 - 3S_3) & 0 \\ L(3C_3 - C_{23} + S_3 + 1) & L(3C_3 + S_3 + 1) & L \\ 0 & 0 & 0 \end{bmatrix}$$

C: ▸ Users ▸ bhara ▸ OneDrive ▸ Documents ▸ MATLAB

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q2AdjointJacobian.m

Q2ForwardKinematics.m | Q2AdjointJacobian.m

```
inverse_Q2 = [[ cos(theta1 + theta2 + theta3), sin(theta1 + theta2 + theta3), 0, -L*(3*cos(theta3) - cos(theta2 + theta3) + sin(theta3) + 1)]
[-sin(theta1 + theta2 + theta3), cos(theta1 + theta2 + theta3), 0,     -L*(sin(theta2 + theta3) + cos(theta3) - 3*sin(theta3))]
[                             0,                             0, 1,                                                           0]
[                             0,                             0, 0,                                                           1]];

%Final Space jacobian evaluation
   SpaceJac = [[0, 0, 0]
[0, 0, 0]
[1, 1, 1]
[0, -L*sin(theta1), L*(cos(theta1 + theta2) + 3*sin(theta1 + theta2) - sin(theta1))]
[0, L*cos(theta1), L*(sin(theta1 + theta2) - 3*cos(theta1 + theta2) + cos(theta1))]
[0, 0, 0]];

%Final Body jacobian evaluation
  BodyJac = [[                                            0,                                   0, 0]
[                                                         0,                                   0, 0]
[                                                         1,                                   1, 1]
[   -L*(sin(theta2 + theta3) + cos(theta3) - 3*sin(theta3)),    -L*(cos(theta3) - 3*sin(theta3)), 0]
[L*(3*cos(theta3) - cos(theta2 + theta3) + sin(theta3) + 1), L*(3*cos(theta3) + sin(theta3) + 1), L]
[                                                         0,                                   0, 0]];
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> disp(simplify(expand(Adjoint_inverseQ2*SpaceJac)))
[                                                         0,                                   0, 0]
[                                                         0,                                   0, 0]
[                                                         1,                                   1, 1]
[   -L*(sin(theta2 + theta3) + cos(theta3) - 3*sin(theta3)),    -L*(cos(theta3) - 3*sin(theta3)), 0]
[L*(3*cos(theta3) - cos(theta2 + theta3) + sin(theta3) + 1), L*(3*cos(theta3) + sin(theta3) + 1), L]
[                                                         0,                                   0, 0]
```

Zoom: 100% | UTF-8 | CRLF | script | Ln 48 | Col 103

2.4 Geometric Jacobian ($J(\theta)$) :-

$$\begin{bmatrix} \omega_s \\ \dot{p} \end{bmatrix} = J(\theta)\,\dot{\theta}$$

$$\Rightarrow J(\theta) = \begin{bmatrix} R_{sb} & 0 \\ 0 & R_{sb} \end{bmatrix} J_b(\theta)$$

Here R_{sb} can be taken from $T_{sb}(\theta)$ calculated through MATLAB :-

$$R_{sb} = \begin{bmatrix} C_{123} & -S_{123} & 0 \\ S_{123} & C_{123} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Using MATLAB, we get :-

$$\therefore J(\theta) = \left[\begin{array}{c|c|c} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ -L(S_{123}+C_{12}+3S_{12}-S_1) & -L(S_{123}+C_{12}+3S_{12}) & -LS_{123} \\ L(C_{123}+3C_{12}-S_{12}-C_1) & L(C_{123}+3C_{12}-S_{12}) & LC_{123} \\ 0 & 0 & 0 \end{array}\right]$$

C: ▸ Users ▸ bhara ▸ OneDrive ▸ Documents ▸ MATLAB

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q2AdjointJacobian.m

Q2ForwardKinematics.m | Q2AdjointJacobian.m

```
    p_inverseQ2 = inverse_Q2(1:3, 4);
    Adjoint_inverseQ2 = [R_inverseQ2, zeros(3, 3);
        bracket(p_inverseQ2)*R_inverseQ2, R_inverseQ2;];

    %Rsb used for Geometric Jacobian calculation
    R_geometric = [[cos(theta1 + theta2 + theta3), -sin(theta1 + theta2 + theta3), 0]
        [sin(theta1 + theta2 + theta3), cos(theta1 + theta2 + theta3), 0]
        [0, 0, 1]];

    Geometric_Matrix = [R_geometric, zeros(3, 3);
        zeros(3, 3), R_geometric];

    Geometric_Jacobian = Geometric_Matrix*BodyJac;

    %Px and Py from the geometric jacobian for singularity and manipulability
    %calculations
    J = [[-L*(sin(theta1 + theta2 + theta3) + cos(theta1 + theta2) + 3*sin(theta1 + theta2) - sin(theta1)), -L*(sin(theta1 + theta2 + theta3) + cos(theta1 + theta2) + 3*sin(theta1 + theta2)), -
        [L*(cos(theta1 + theta2 + theta3) + 3*cos(theta1 + theta2) - sin(theta1 + theta2) - cos(theta1)),  L*(cos(theta1 + theta2 + theta3) + 3*cos(theta1 + theta2) - sin(theta1 + theta2)),  L*

    %Skew symmetric convertor
    function w_matrix = bracket(w)
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> disp(simplify(expand(Geometric_Matrix*BodyJac)))
[                                                                          0,                                                              0,
[                                                                          0,                                                              0,
[                                                                          1,                                                              1,
[-L*(sin(theta1 + theta2 + theta3) + cos(theta1 + theta2) + 3*sin(theta1 + theta2) - sin(theta1)), -L*(sin(theta1 + theta2 + theta3) + cos(theta1 + theta2) + 3*sin(theta1 + theta2)), -L*s
[ L*(cos(theta1 + theta2 + theta3) + 3*cos(theta1 + theta2) - sin(theta1 + theta2) - cos(theta1)),  L*(cos(theta1 + theta2 + theta3) + 3*cos(theta1 + theta2) - sin(theta1 + theta2)),  L*c
[                                                                          0,                                                              0,
fx
```

3.11 home position: M :-

$$M = \begin{bmatrix} 1 & 0 & 0 & L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L_1+L_2 \\ & 0 & & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ as all axes align with each other in the same direction for both frames {s} and {b}.

- Now for screws S_1 :

$\omega_s = [0\ 0\ 1]^T$ $\quad q = [0\ 0\ L_1]^T$ or $[0\ 0\ 0]^T$

$$\therefore \quad S_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- for screw S_2 :

Prismatic joint so, $S_2 = \begin{bmatrix} v_s \\ 0 \end{bmatrix}$

$$v_s = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$$

$$S_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- for screw S_3 :

$$S_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- for screw S_4 :

$$\omega_s = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \qquad q = \begin{bmatrix} 0 & 0 & L_1 + L_2 \end{bmatrix}^T$$

$$\therefore \quad S_4 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ L_1 + L_2 \\ 0 \end{bmatrix}$$

$$\therefore \quad T_{sb}(\theta) = e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} e^{[S_4]\theta_4} M$$

<u>Using MATLAB and simplification</u>, we get :-

$$T_{sb}(\theta) = \begin{bmatrix} C_{12} & -\dfrac{S_{1+2-3-4}}{2} - \dfrac{S_{1234}}{2} & \dfrac{C_{1+2-3-4}}{2} - \dfrac{C_{1234}}{2} & T_{sb}(1,4) \\ S_{12} & \dfrac{C_{1+2-3-4}}{2} + \dfrac{C_{1234}}{2} & \dfrac{S_{1+2-3-4}}{2} - \dfrac{S_{1234}}{2} & T_{sb}(2,4) \\ 0 & S_{34} & C_{34} & T_{sb}(3,4) \\ 0 & 0 & 0 & T_{sb}(4,4) \end{bmatrix}$$

Since, values are long; i'll write the last column separately :-

$$T_{sb}(1,4) = \frac{L_1 C_{12-3}}{2} + \frac{L_2 C_{1+2-3}}{2} + L_2 C_{12} - \frac{L_1 C_{123}}{2} - \frac{L_2 C_{123}}{2}$$

$$T_{sb}(2,4) = \frac{L_1 S_{12-3}}{2} + \frac{L_2 S_{12-3}}{2} + L_2 S_{12} - \frac{L_1 S_{123}}{2} - \frac{L_2 S_{123}}{2}$$

$$T_{sb}(3,4) = C_3 (L_1 + L_2)$$

$$T_{sb}(4,4) = 1$$

C: ▸ Users ▸ bhara ▸ OneDrive ▸ Documents ▸ MATLAB

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q3AdjointJacobian.m

Q3AdjointJacobian.m | Q3ForwardKinematics.m

```
[                 0,                  0, 0, 1]];

  % T_expm123 = expm(bracket(S1) * theta1) * expm(bracket(S2) * theta2) * expm(bracket(S3) * theta3)
  T_expm123 = [[cos(theta1 + theta2), -sin(theta1 + theta2)*cos(theta3),  sin(theta1 + theta2)*sin(theta3), 0]
[sin(theta1 + theta2),  cos(theta1 + theta2)*cos(theta3), -cos(theta1 + theta2)*sin(theta3), 0]
[                 0,                  sin(theta3),                  cos(theta3), 0]
[                 0,                            0,                            0, 1]];

   %Simplified version of the final Tsb(theta) forward kinematics
  T_sb = [[cos(theta1 + theta2), - sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)/2, cos(theta1 + theta2 - theta3 - theta4)/2 - cos(theta1 + theta2 + theta3
[sin(theta1 + theta2),   cos(theta1 + theta2 - theta3 - theta4)/2 + cos(theta1 + theta2 + theta3 + theta4)/2, sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)
[                 0,                                          sin(theta3 + theta4),                                          cos(theta3 + theta
[                 0,                                                             0,

  %Inverse of Tsb(theta) for calculation of Body Jacobian
  T_sb_inverse = [[                                                 cos(theta1 + theta2),                                                 sin(theta1 + theta2),
[- sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)/2, cos(theta1 + theta2 - theta3 - theta4)/2 + cos(theta1 + theta2 + theta3 + theta4)/2, sin(theta3 + theta
[  cos(theta1 + theta2 - theta3 - theta4)/2 - cos(theta1 + theta2 + theta3 + theta4)/2, sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)/2, cos(theta3 + theta
[                                                                                    0,                                                                                    0,

  %Final Space jacobian evaluation
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> Q3ForwardKinematics
>> disp(simplify(expand(T_finalQ3)))
[cos(theta1 + theta2), - sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)/2, cos(theta1 + theta2 - theta3 - theta4)/2 - cos(theta1 + theta2 + theta3 + the
[sin(theta1 + theta2),   cos(theta1 + theta2 - theta3 - theta4)/2 + cos(theta1 + theta2 + theta3 + theta4)/2, sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + the
[                 0,                                         sin(theta3 + theta4),                                          cos(theta3 + t
[                 0,                                                            0,

fx >>
```

Zoom: 100% | UTF-8 | CRLF | script / bracket | Ln 100 | Col 31

$\boxed{3.4}$

$$e^{[S_1]\theta_1} = \begin{bmatrix} C_1 & -S_1 & 0 & 0 \\ S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$e^{[S_1]\theta_1} e^{[S_2]\theta_2}$$

$$= \begin{bmatrix} C_{12} & -S_{12} & 0 & 0 \\ S_{12} & C_{12} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

&,

$$e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3}$$

$$= \begin{bmatrix} C_{12} & -C_3 S_{12} & S_3 S_{12} & 0 \\ S_{12} & C_3 C_{12} & -S_3 C_{12} & 0 \\ 0 & S_3 & C_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q3AdjointJacobian.m

Q3ForwardKinematics.m | Q3AdjointJacobian.m | +

```
 %VERIFICATION VALUES:

 %T_expm1 = T1 = expm(bracket(S1) * theta1)
  T_expm1 = [[cos(theta1), -sin(theta1), 0, 0]
[sin(theta1),  cos(theta1), 0, 0]
[          0,            0, 1, 0]
[          0,            0, 0, 1]];

  % T_expm12 = expm(bracket(S1) * theta1) * expm(bracket(S2) * theta2)
  T_expm12 = [[cos(theta1 + theta2), -sin(theta1 + theta2), 0, 0]
[sin(theta1 + theta2),  cos(theta1 + theta2), 0, 0]
[                   0,                      0, 1, 0]
[                   0,                      0, 0, 1]];

  % T_expm123 = expm(bracket(S1) * theta1) * expm(bracket(S2) * theta2) * expm(bracket(S3) * theta3)
  T_expm123 = [[cos(theta1 + theta2), -sin(theta1 + theta2)*cos(theta3),  sin(theta1 + theta2)*sin(theta3), 0]
[sin(theta1 + theta2),  cos(theta1 + theta2)*cos(theta3), -cos(theta1 + theta2)*sin(theta3), 0]
[                   0,                      sin(theta3),                       cos(theta3), 0]
[                   0,                                0,                                 0, 1]];

   %Simplified version of the final Tsb(theta) forward kinematics
  T_sb = [[cos(theta1 + theta2), - sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)/2, cos(theta1 + theta2 - theta3 - theta4)/2 - cos(theta1 + theta2 + theta3
[sin(theta1 + theta2),   cos(theta1 + theta2 - theta3 - theta4)/2 + cos(theta1 + theta2 + theta3 + theta4)/2, sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)
[                   0,                                          sin(theta3 + theta4),                                          cos(theta3 + theta
[                   0,                                                              0,
```

Command Window

$\therefore$ For 4-joints : (Space Jacobian)

$$J_s(\theta) = \left[S_1 \quad Ad_{e^{[S_1]\theta_1}} S_2 \quad Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2}} S_3 \quad Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3}} S_4 \right]$$

$$= \begin{bmatrix} 0 & 0 & C_{12} & C_{12} \\ 0 & 0 & S_{12} & S_{12} \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -C_3(L_1+L_2)\,S_{12} \\ 0 & 0 & 0 & C_3(L_1+L_2)\,C_{12} \\ 0 & 0 & 0 & S_3(L_1+L_2) \end{bmatrix}$$

3.31 Body Jacobian $J(\theta)$:

$$J_b(\theta) = Ad_{T_{sb}^{-1}} \, J_s(\theta) \qquad \left[\because Ad_{T_{sb}^{-1}} = Ad_{T_{sb}}^{-1} \right]$$

Using MATLAB, we compute and simplify :-

$$J_b(\theta) = \begin{bmatrix} 0 & 0 & 1 & 1 \\ S_{34} & S_{34} & 0 & 0 \\ C_{34} & C_{34} & 0 & 0 \\ S_3(L_1+L_2) & S_3(L_1+L_2) & 0 & 0 \\ L_2 C_{34} & L_2 C_{34} & -C_4(L_1+L_2) & 0 \\ -L_2 S_{34} & -L_2 S_{34} & S_4(L_1+L_2) & 0 \end{bmatrix}$$

3.4 Geometric Jacobian :-

$$\begin{bmatrix} \omega_s \\ \dot{p} \end{bmatrix} = J(\theta)\,\dot{\theta}$$

$$J(\theta) = \begin{bmatrix} R_{sb} & 0 \\ 0 & R_{sb} \end{bmatrix} J_s(\theta)$$

Here, R_{sb} from $T_{sb}(\theta)$:

$$R_{sb} = \begin{bmatrix} C_{12} & \dfrac{-S_{12-3-4} - S_{1234}}{2} & \dfrac{C_{12-3-4} - C_{1234}}{2} \\ S_{12} & \dfrac{C_{12-3-4} + C_{1234}}{2} & \dfrac{S_{12-3-4} - S_{1234}}{2} \\ 0 & S_{34} & C_{34} \end{bmatrix}$$

Computing Using MATLAB :-

$$
\begin{bmatrix}
0 & 0 & C_{12} & C_{12} \\
0 & 0 & S_{12} & S_{12} \\
1 & 1 & 0 & 0 \\
\frac{(L_1+L_2)}{2}(S_{123} - S_{12-3}) - L_2 S_{12} & \text{repeat} & +S_{12}C_3(L_1+L_2) & 0 \\
\frac{(L_1+L_2)}{2}(S_{12-3} - C_{123}) + L_2 C_{12} & \text{repeat} & -C_{12}C_3(L_1+L_2) & 0 \\
0 & 0 & -S_3(L_1+L_2) & 0
\end{bmatrix}
$$

C: ▸ Users ▸ bhara ▸ OneDrive ▸ Documents ▸ MATLAB

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q3AdjointJacobian.m

Q3ForwardKinematics.m | Q3AdjointJacobian.m | +

```
[                                                                        0,

  %Final Space jacobian evaluation
  SpaceJacQ3 = [[0, 0, cos(theta1)*cos(theta2) - sin(theta1)*sin(theta2), cos(theta1)*cos(theta2) - sin(theta1)*sin(theta2)]
[0, 0, cos(theta1)*sin(theta2) + cos(theta2)*sin(theta1), cos(theta1)*sin(theta2) + cos(theta2)*sin(theta1)]
[1, 1, 0, 0]
[0, 0, 0, -cos(theta3)*(cos(theta1)*sin(theta2) + cos(theta2)*sin(theta1))*(L1 + L2)]
[0, 0, 0, cos(theta3)*(cos(theta1)*cos(theta2) - sin(theta1)*sin(theta2))*(L1 + L2)]
[0, 0, 0, sin(theta3)*(L1 + L2)]];


  %Final Body jacobian evaluation
  BodyJacQ3 = [[                0,                     0,                     1, 1]
[   sin(theta3 + theta4),     sin(theta3 + theta4),                       0, 0]
[   cos(theta3 + theta4),     cos(theta3 + theta4),                       0, 0]
[  sin(theta3)*(L1 + L2),    sin(theta3)*(L1 + L2),                       0, 0]
[ L2*cos(theta3 + theta4),  L2*cos(theta3 + theta4), -cos(theta4)*(L1 + L2), 0]
[-L2*sin(theta3 + theta4), -L2*sin(theta3 + theta4),  sin(theta4)*(L1 + L2), 0]];

  %Final Geometric Jacobian evaluation
  GeometricJacQ3 = [[0, 0, cos(theta1 + theta2), cos(theta1 + theta2)]
[0, 0, sin(theta1 + theta2), sin(theta1 + theta2)]
[1, 1, 0, 0]
[(L1*sin(theta1 + theta2 + theta3))/2 - (L2*sin(theta1 + theta2 - theta3))/2 - L2*sin(theta1 + theta2) - (L1*sin(theta1 + the
[(L1*cos(theta1 + theta2 - theta3))/2 + (L2*cos(theta1 + theta2 - theta3))/2 + L2*cos(theta1 + theta2) - (L1*cos(theta1 + the
[0, 0, -sin(theta3)*(L1 + L2), 0]];

  %Adjoint calculations for all the respective Transformation matrices
  R1 = T_expm1(1:3, 1:3);
  p1 = T_expm1(1:3, 4);
  Adjoint_1 = [R1, zeros(3, 3);
      bracket(p1)*R1, R1;];
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> disp(simplify(expand(Adjoint_sb_inverse*SpaceJacQ3)))
[                0,                        0,
[   sin(theta3 + theta4),     sin(theta3 + theta4),
[   cos(theta3 + theta4),     cos(theta3 + theta4),
[  sin(theta3)*(L1 + L2),    sin(theta3)*(L1 + L2),
[ L2*cos(theta3 + theta4),  L2*cos(theta3 + theta4), -co
[-L2*sin(theta3 + theta4), -L2*sin(theta3 + theta4),  si

>> disp(simplify(expand(Geometric_JacobianQ3)))
[
[
[
[(L1*sin(theta1 + theta2 + theta3))/2 - (L2*sin(theta1 +
[(L1*cos(theta1 + theta2 - theta3))/2 + (L2*cos(theta1 +
[

fx >>
```

C: ▸ Users ▸ bhara ▸ OneDrive ▸ Documents ▸ MATLAB

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q3AdjointJacobian.m *

Q3ForwardKinematics.m | Q3AdjointJacobian.m *

```matlab
    R12 = T_expm12(1:3, 1:3);
    p12 = T_expm12(1:3, 4);
    Adjoint_12 = [R12, zeros(3, 3);
        bracket(p12)*R12, R12;];

    R123 = T_expm123(1:3, 1:3);
    p123 = T_expm123(1:3, 4);
    Adjoint_123 = [R123, zeros(3, 3);
        bracket(p123)*R123, R123;];

     R_sb_inverse = T_sb_inverse(1:3, 1:3);
    p_sb_inverse = T_sb_inverse(1:3, 4);
    Adjoint_sb_inverse = [R_sb_inverse, zeros(3, 3);
        bracket(p_sb_inverse)*R_sb_inverse, R_sb_inverse;];

    %Geometric Jacobian calculation
    R_geometricQ3 = [[cos(theta1 + theta2), - sin(theta1 + theta2 - theta3 - theta4)/2 - sin(theta1 + theta2 + theta3 + theta4)
        [sin(theta1 + theta2),   cos(theta1 + theta2 - theta3 - theta4)/2 + cos(theta1 + theta2 + theta3 + theta4)/2, sin(theta
        [0, sin(theta3 + theta4), cos(theta3 + theta4)]];

    Geometric_MatrixQ3 = [R_geometricQ3, zeros(3, 3);
        zeros(3, 3), R_geometricQ3];

    Geometric_JacobianQ3 = Geometric_MatrixQ3*BodyJacQ3;

    %Skew symmetric convertor
    function w_matrix = bracket(w)
      w_matrix = [ 0 -w(3) w(2);
      w(3) 0 -w(1);
      -w(2) w(1) 0];
    end
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> disp(simplify(expand(Adjoint_sb_inverse*SpaceJacQ3)))
[                    0,                     0,
[   sin(theta3 + theta4),     sin(theta3 + theta4),
[   cos(theta3 + theta4),     cos(theta3 + theta4),
[  sin(theta3)*(L1 + L2),    sin(theta3)*(L1 + L2),
[ L2*cos(theta3 + theta4),  L2*cos(theta3 + theta4), -co
[-L2*sin(theta3 + theta4), -L2*sin(theta3 + theta4),  si

>> disp(simplify(expand(Geometric_JacobianQ3)))
[
[
[
[(L1*sin(theta1 + theta2 + theta3))/2 - (L2*sin(theta1 +
[(L1*cos(theta1 + theta2 - theta3))/2 + (L2*cos(theta1 +
[

>>
```

Zoom: 100% | UTF-8 | CRLF | script | Ln 89 | Col 3

[4] finding the screws for each joint :

$$S_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad S_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ -L_1 \\ 0 \end{bmatrix} \qquad S_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ -L_1-L_2 \\ 0 \end{bmatrix}$$

$\therefore$ Space Jacobian :

$$J_s(\theta) = \begin{bmatrix} S_1 & Ad_{e^{[S_1]\theta_1}} S_2 & Ad_{e^{[S_1]\theta_1} e^{[S_2]\theta_2}} S_3 \end{bmatrix}$$

$\Rightarrow$ Using MATLAB, we get :

$$J_s(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & L_1 S_1 & L_1 S_1 + L_2 S_{12} \\ 0 & -LC_1 & -L_1 C_1 - L_2 C_{12} \\ 0 & 0 & 0 \end{bmatrix}$$

Then we proceed to find the Geometric Jacobian :-

$$J(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ -L_1S_1 - L_2S_{12} - L_3S_{123} & -L_2S_{12} - L_3S_{123} & -L_3S_{123} \\ L_1C_1 + L_2C_{12} + L_3C_{123} & L_2C_{12} + L_3C_{123} & L_3C_{123} \\ 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \\ \dot{p}_x \\ \dot{p}_y \\ \dot{p}_z \end{bmatrix}$$

bottom three rows capture linear velocity.

$$\left(\because \begin{bmatrix} \omega_s \\ \dot{p} \end{bmatrix} = J(\theta)\,\dot{\theta} \right)$$

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \dot{P}_x \\ \dot{P}_y \\ \dot{P}_z \end{bmatrix} = \begin{bmatrix} (-L_1S_1 - L_2S_{12} - L_3S_{123})\dot{\theta}_1 + (-L_2S_{12} - L_3S_{123})\dot{\theta}_2 + (-L_3S_{123})\dot{\theta}_3 \\ (L_1C_1 + L_2C_{12} + L_3C_{123})\dot{\theta}_1 + (L_2C_{12} + L_3C_{123})\dot{\theta}_2 + (L_3C_{123})\dot{\theta}_3 \\ 0 \end{bmatrix}$$

- Case I :-

Joint positions $\theta_1 = \frac{-\pi}{8}$, $\theta_2 = \frac{\pi}{4}$, $\theta_3 = \pi/8$

$$\Rightarrow \begin{bmatrix} \dot{P}_x \\ \dot{P}_y \\ \dot{P}_z \end{bmatrix} = \begin{bmatrix} -L_1(0.707)\dot{\theta}_1 - 1.08L_1\dot{\theta}_2 - 0.707L_1\dot{\theta}_3 \\ 2.55L_1\dot{\theta}_1 + 1.63L_1\dot{\theta}_2 + 0.707L_1\dot{\theta}_3 \\ 0 \end{bmatrix}$$

Here, we want to maximize $|\dot{p}_x|$;

$\therefore$ $\dot{P}_x = (-0.707L_1\dot{\theta}_1 - 1.08L_1\dot{\theta}_2 - 0.707L_1\dot{\theta}_3)$

Since, we want to find out which joint to ~~maximi~~ actuate to maximize $|\dot{p}_x|$.

$\therefore$ Thus, upon actuating joint$_2$ $\dot{\theta}_2 = 1$ rad/s.

$\Rightarrow$ $\dot{P}_x = -1.08L_1$; taking the absolute ;

$|\dot{P}_x| = 1.08L_1$ for actuating joint 2 $(\dot{\theta}_2)$.

• Case II :

$\theta_1 = 3\pi/4$; $\theta_2 = -\pi/4$; $\theta_3 = 0$

* We want to maximize $|\dot{p}_x|$:

$\therefore \dot{p}_x = (-2.707 L_1 \dot{\theta}_1 - 2L_1 \dot{\theta}_2 - L_1 \dot{\theta}_3)$

$\therefore$ By actuating Joint 1 ; $\dot{\theta}_1 = 1$ rad/s.

We get :

$\dot{p}_x = -2.707 L_1$

$\Rightarrow \boxed{|\dot{p}_x| = 2.707 L_1}$ by actuating Joint 1 ($\dot{\theta}_1 = 1$ r/s)

• Case III :-

$\theta_1 = \pi/2$; $\theta_2 = -\pi/8$; $\theta_3 = -\pi/2$.

And we want to maximize $|\dot{p}_y|$ this time :

Substituting the initial joint positions given above :-

$$\dot{p}_y = (1.3066\, L_1 \dot{\theta}_1 + 1.3066\, L_1 \dot{\theta}_2 + 0.924 L_1 \dot{\theta}_3)$$

∴ Thus, we can either actuate Joint 1 or 2 ; i.e. either $\dot{\theta}_1 = 1$ rad/s or $\dot{\theta}_2 = 1$ rad/s, both will result in equal solutions .

$|\dot{p}_y| = 1.3066 L_1$ (for either $\dot{\theta}_1$ or $\dot{\theta}_2$ actuation) .

$\boxed{5.1}$ We have the Geometric Jacobian for the provided robot configuration :

$$J(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ -L(S_{123}+C_{12}+3S_{12}-S_1) & -L(S_{123}+C_{12}+3S_{12}) & -LS_{123} \\ L(C_{123}+3C_{12}-S_{12}-C_1) & L(C_{123}+3C_{12}-S_{12}) & LC_{123} \\ 0 & 0 & 0 \end{bmatrix}$$

$\therefore$ We take out the linear velocities :

$$\underbrace{\begin{bmatrix} \dot{P}_x \\ \dot{P}_y \end{bmatrix}}_{\text{task space}} = \underbrace{\begin{bmatrix} -L(S_{123}+C_{12}+3S_{12}-S_1) & -L(S_{123}+C_{12}+3S_{12}) & -LS_{123} \\ L(C_{123}+3C_{12}-S_{12}-C_1) & L(C_{123}+3C_{12}-S_{12}) & LC_{123} \end{bmatrix}}_{\text{Jacobian } J} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

$$\Rightarrow J = \begin{bmatrix} -L(S_{123} + C_{12} + 3S_{12} - S_1) & -L(S_{123} + C_{12} + 3S_{12}) & -LS_{123} \\ L(C_{123} + 3C_{12} - S_{12} - C_1) & L(C_{123} + 3C_{12} - S_{12}) & LC_{123} \end{bmatrix}$$

$$\det(JJ^T) = L^4 \Big(3\cos(\theta_2 + 2\theta_3) - 3\cos(2\theta_2 + \theta_3)$$

$$+ \sin(\theta_2 + 2\cdot\theta_3) + \sin(2\theta_2 + \theta_3)$$

$$- 4\cos(2\theta_2) - 8\cos(2\theta_3) + 3\sin(2\theta_2)$$

$$- 6\sin(2\theta_3) - \cos(2\theta_2 + 2\theta_3) - 3\cos(\theta_2) +$$

$$3\cos(\theta_3) + \sin(\theta_2) + \sin(\theta_3) + 16 \Big).$$

$$\Rightarrow \det JJ^T = 0$$

Solving this gives:

$\theta_1 = 0$, $\theta_2 = -0.3218$ rad. (-18.4349°)

$\theta_3 = 0.3218$ rad. (18.4349°) (Jacobian drops Rank)

The joint positions for Singularity:

$\boxed{\theta_1 = 0}$ $\boxed{\theta_2 = -0.3218 \text{ radians}}$ $\boxed{\theta_3 = 0.3218 \text{ rads}}$

for $\theta_1 = -0.245$ or 2.8966 radians

-14 degrees or 165.96 degrees

$\theta_2 = \theta_3 = 0$

$$\Rightarrow \begin{bmatrix} \dot{P}_x \\ \dot{P}_y \end{bmatrix} = \begin{bmatrix} 0 \\ -3.153 L \dot{\theta}_1 - 4.123 L \dot{\theta}_2 - 0.97 L \dot{\theta}_3 \end{bmatrix}$$

In this configuration, for these specific joint values, we can only see that the robot can instantaneously move in y-direction only, as $\dot{P}_x$ is 0.

Similarly:

for $\theta_1 = -1.816$ or 1.3258 radians

-104.03 or 75.96 degrees

We get:

$$\begin{bmatrix} \dot{P}_x \\ \dot{P}_y \end{bmatrix} = \begin{bmatrix} 3.153\,\dot{\theta}_1 + 4.123\,\dot{\theta}_2 + 0.9701\,\dot{\theta}_3 \\ 0 \end{bmatrix}$$

for these joint values, the robot can only instantaneously move in x-axis.

→ $\text{Rank}(J(\theta)) < 2$ for both cases.

$= 1$

$\therefore$ for $\theta = [1.3258, 0, 0]$ joint positions,

the robot can only ~~sim~~ instantaneously move in x-direction (i.e. the x-axis).

& for $\theta = [2.8966, 0, 0]$ joint positions,

the robot can only instantaneously move in y-directions (y-axis).

γ_1 we know,

$n = 2$, robot's task space's dimensions.

for both of those joint positions configurations,

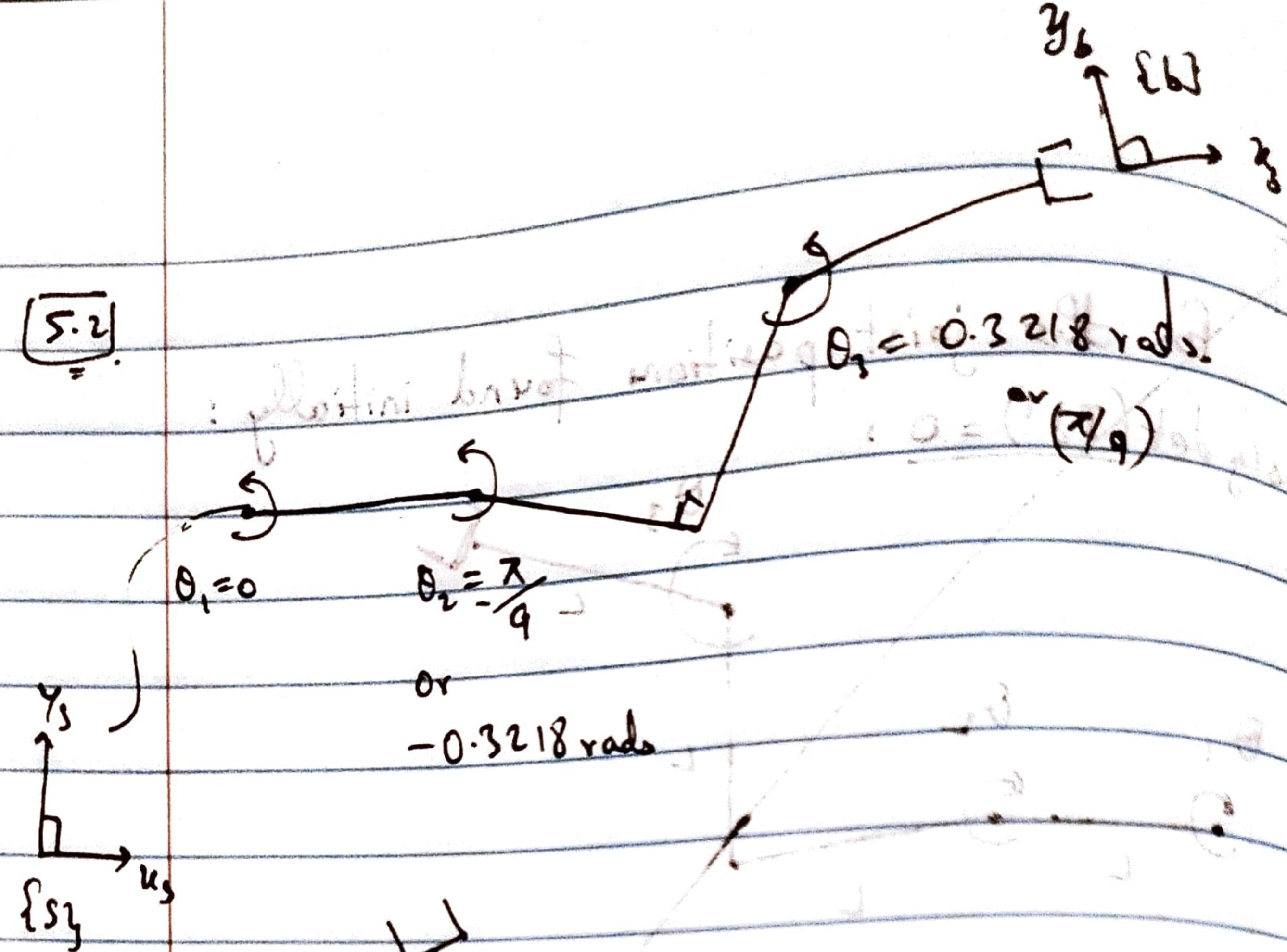
rank of $J(\theta)$ drops below n, i.e. 2, losing movement/rotation in one of the axes respectively.

- We found 2 cases of singularities above.

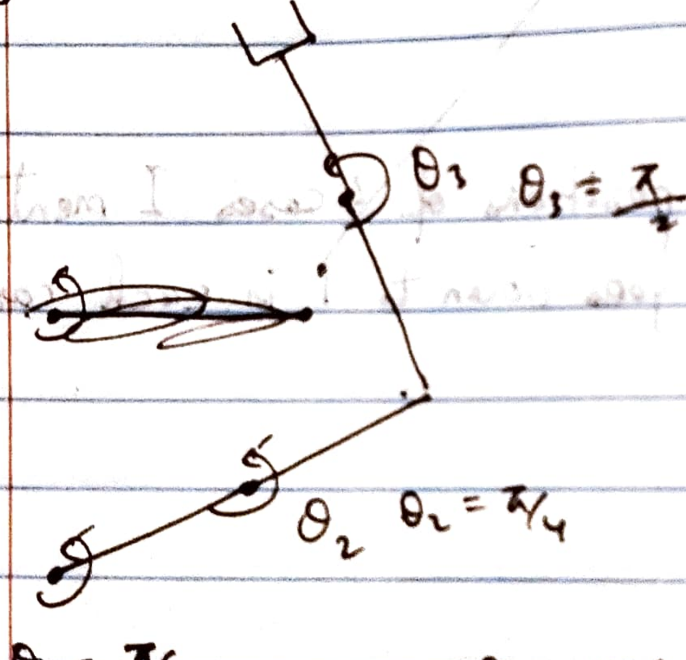
  $[\text{rank}(J(\theta)) < 2]$

Here, $\det(JJ^T) = 0$ to further verify.

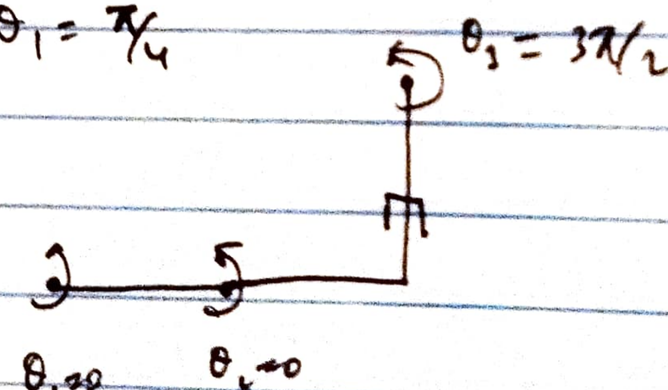
5.2



y_b
{b}
x_b
$\theta_3 = 0.3218$ rads.
or ($\pi/9$)
$\theta_1 = 0$
$\theta_2 = -\pi/9$
or
-0.3218 rads
y_s
x_s
{s}



θ_3
$\theta_3 = \frac{\pi}{2}$
θ_2
$\theta_2 = \pi/4$
$\theta_1 = \pi/4$



$\theta_3 = 3\pi/2$
$\theta_1 = 0$
$\theta_2 = 0$

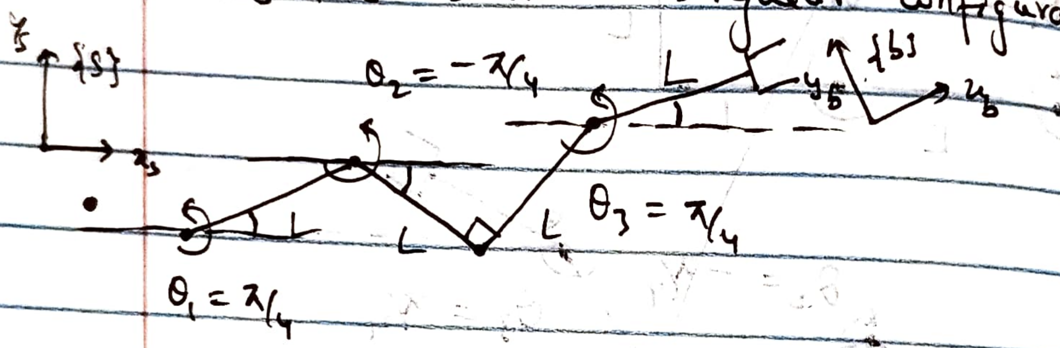
$\boxed{5.3}$ for $\theta = [\pi/4, -\pi/4, \pi/4]$.

putting in $\det JJ^T = 0$ ..

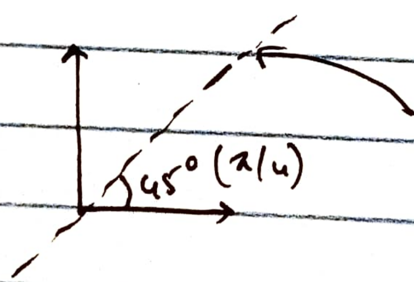
We get the result as 6

$\Rightarrow$ $6 \neq 0$.

- Thus this is not a singular configuration.



- The robot can move in the x-y plane



Plot of movement possibility.

- Robot cannot move out of it's task space, that is along the z-axis or rotations about x/y axes.

[5.4] We want to find a joint position that maximizes $\sqrt{\det(JJ^T)}$.

$$\therefore J(\theta) = \begin{bmatrix} -L(S_{123} + C_{12} + 3S_{12} - S_1) & -L(S_{123} + C_{12} + 3S_{12}) & -LS_{123} \\ L(C_{123} + 3C_{12} - S_{12} - C_1) & L(C_{123} + 3C_{12} - S_{12}) & LC_{123} \end{bmatrix}$$

$\sqrt{\det(JJ^T)}$ = Long solution (thus, referring to MATLAB :-

We maximize the robot's manipulability when

$\rightarrow$ $\theta = 0 \pm \frac{n\pi}{9}$.

C: ▸ Users ▸ bhara ▸ OneDrive ▸ Documents ▸ MATLAB

Editor - C:\Users\bhara\OneDrive\Documents\MATLAB\Q2AdjointJacobian.m

Q2ForwardKinematics.m | Q2AdjointJacobian.m | +

```matlab
    p_sb = T_sb(1:3, 4);
    Adjoint_final = [R_sb, zeros(3, 3);
        bracket(p_sb)*R_sb, R_sb;];

    R_inverseQ2 = inverse_Q2(1:3, 1:3);
    p_inverseQ2 = inverse_Q2(1:3, 4);
    Adjoint_inverseQ2 = [R_inverseQ2, zeros(3, 3);
        bracket(p_inverseQ2)*R_inverseQ2, R_inverseQ2;];

    %Rsb used for Geometric Jacobian calculation
    R_geometric = [[cos(theta1 + theta2 + theta3), -sin(theta1 + theta2 + theta3), 0]
        [sin(theta1 + theta2 + theta3), cos(theta1 + theta2 + theta3), 0]
        [0, 0, 1]];

    Geometric_Matrix = [R_geometric, zeros(3, 3);
        zeros(3, 3), R_geometric];

    Geometric_Jacobian = Geometric_Matrix*BodyJac;

    %Px and Py from the geometric jacobian for singularity and manipulability
    %calculations
    J = [[-L*(sin(theta1 + theta2 + theta3) + cos(theta1 + theta2) + 3*sin(theta1 + theta2) - sin(theta1)), -L*(sin(theta1 + th
        [L*(cos(theta1 + theta2 + theta3) + 3*cos(theta1 + theta2) - sin(theta1 + theta2) - cos(theta1)),  L*(cos(theta1 + thet

    %Skew symmetric convertor
    function w_matrix = bracket(w)
      w_matrix = [ 0 -w(3) w(2);
      w(3) 0 -w(1);
      -w(2) w(1) 0];
    end
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>> Q2AdjointJacobian
>> disp(simplify(det(expand(J*J'))))
L^4*(3*cos(theta2 + 2*theta3) - 3*cos(2*theta2 + theta3

>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
```

$\boxed{61}$

$\boxed{6.11}$ Noting down Geometric Jac. of the configuration :-

$$\begin{bmatrix} 0 & 0 & C_{12} & C_{12} \\ 0 & 0 & S_{12} & S_{12} \\ 1 & \cancel{1} & 0 & 0 \\ \frac{(L_1+L_2)}{2}(S_{123} - S_{12-3}) - L_2 S_{12} & \text{repeat} & +S_{12}C_3(L_1+L_2) & 0 \\ \frac{(L_1+L_2)}{2}(C_{12-3} - C_{123}) + L_2 C_{12} & \text{repeat} & -C_{12}C_3(L_1+L_2) & 0 \\ 0 & 0 & -S_3(L_1+L_2) & 0 \end{bmatrix}$$

Here, ω_x, ω_y have values thus movement occurs,

as well as $\dot{p}_x$ & $\dot{p}_y$, $\dot{p}_z$ have dependent variables.

Here, det $JJ^T \neq 0$.

due to the invertibility.

$$\underline{\det(JJ^T) \neq 0}$$

Due to having angular velocity as well as linear velocity, the rank is unable to go below task space n.

Rank (J) doesn't go below the task space n.

∴ Alternatively, there is not any singular ~~po~~ configuration that exists.

Rank doesn't go below n.

$\boxed{6.2}$ $\sqrt{\det(JJ^T)}$

∴ Upon calculation, we get

$$\underline{\theta = 0 \pm n\pi}$$

Variable manipulability due to the different dependent values in ω_x, ω_y & $\dot{P}_x, P_y, P_z$.