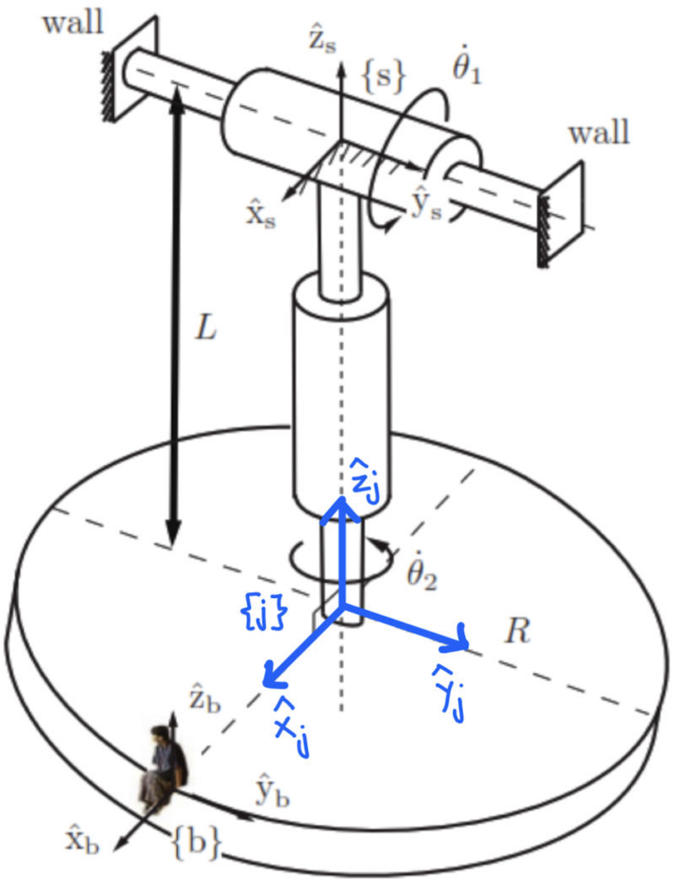
Introduction to Robotics - Homework # 4

Huzaifah Tariq Ahmed (ha07151) & Syed Asghar Abbas Zaidi (sz07201)

Problem - 1



a)

* We have assigned a frame $\{j\}$ in the figure above. The orientation at $t=0$ is same as frame $\{s\}$. So in order to find ${}^{s}T_b$ we will calculate ${}^{s}T_j$ and ${}^{j}T_b$ and multiply them.

* ${}^{s}T_j$ is basically modelling a rotation of frame $\{s\}$ by $\theta_1(t)$ about $\hat{y}_s$ axis, and a translation about $\hat{z}_s$ axis by $-L$ or -20.

* We know that angular velocity $(\dot{\theta}_1) = 1$ rad/s, so θ_1 at time t will be;

$$\theta_1(t) = \int_0^t \dot{\theta}_1(u)\,du = t \text{ radians}$$

* ${}^{j}T_b$ is modelling a rotation by $\theta_2(t)$ about $\hat{z}_j$ axis, and then a translation by R or 10, across the $\hat{x}_j$ axis.

* As both the joints have angular velocity of 1 rad/s, $\dot{\theta}_2 = 1$, and therefore;

$$\theta_2(t) = \int_0^t \dot{\theta}_2(u)\,du = t \text{ radians.}$$

* We will now use matlab to calculate ${}^{s}T_j$ and ${}^{j}T_b$ seperately and take product of both in order to get ${}^{s}T_b$.

```
% Define symbolic variables and parameters
syms t  % Symbolic variables for time and joint angles

L = 20; R = 10;  % Length parameters

% Transformation matrix from frame s to frame j
% Define rotation-only transformation matrix for frame s to frame j
Tsj_rotation = [cos(t), 0, sin(t), 0; 0, 1, 0, 0; -sin(t), 0, cos(t), 0; 0,
0, 0, 1]
```

Tsj_rotation =

$$\begin{pmatrix} \cos(t) & 0 & \sin(t) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(t) & 0 & \cos(t) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
% Define translation vector for frame s to frame j
Tsj_translation = [1, 0, 0, 0; 0, 1, 0, 0; 0, 0, 1, -L; 0, 0, 0, 1]
```

```
Tsj_translation = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1   -20
     0     0     0     1
```

```
% Compute transformation matrix from frame s to frame j
Tsj = Tsj_rotation * Tsj_translation
```

Tsj =

$$\begin{pmatrix} \cos(t) & 0 & \sin(t) & -20\sin(t) \\ 0 & 1 & 0 & 0 \\ -\sin(t) & 0 & \cos(t) & -20\cos(t) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
% Transformation matrix from frame j to frame b
% Define rotation-only transformation matrix for frame j to frame b
Tjb_rotation = [cos(t), -sin(t), 0, 0; sin(t), cos(t), 0, 0; 0, 0, 1, 0; 0,
0, 0, 1]
```

Tjb_rotation =

$$\begin{pmatrix} \cos(t) & -\sin(t) & 0 & 0 \\ \sin(t) & \cos(t) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
% Define translation vector for frame j to frame b
Tjb_translation = [1, 0, 0, R; 0, 1, 0, 0; 0, 0, 1, 0; 0, 0, 0, 1]
```

```
Tjb_translation = 4×4
     1     0     0    10
     0     1     0     0
```

```
     0     0     1     0
     0     0     0     1
```

```
% Compute transformation matrix from frame j to frame b
Tjb = Tjb_rotation * Tjb_translation
```

Tjb =

$$\begin{pmatrix} \cos(t) & -\sin(t) & 0 & 10\cos(t) \\ \sin(t) & \cos(t) & 0 & 10\sin(t) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
% Transformation matrix from frame s to frame b via frame j
Tsb = Tsj * Tjb
```

Tsb =

$$\begin{pmatrix} \cos(t)^2 & \sigma_1 & \sin(t) & 10\cos(t)^2 - 20\sin(t) \\ \sin(t) & \cos(t) & 0 & 10\sin(t) \\ \sigma_1 & \sin(t)^2 & \cos(t) & -20\cos(t) - 10\cos(t)\sin(t) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\sigma_1 = -\cos(t)\sin(t)$$

b)

We know that $S(\omega) = \dot{R}R^{-1}$;

where $S(\omega)$ is a skew symmetric matrix.

$$\Rightarrow S({}^s\omega_{sb}) = {}^s\dot{R}_b\, {}^sR_b^{-1} = {}^s\dot{R}_b\, {}^sR_b^T$$

$$\Rightarrow {}^s\dot{R}_b = \begin{bmatrix} -2\cos(t)\sin(t) & \sin^2(t)-\cos^2(t) & \cos(t) \\ \cos(t) & -\sin(t) & 0 \\ \sin^2(t)-\cos^2(t) & 2\cos(t)\sin(t) & -\sin(t) \end{bmatrix}$$

$$\Rightarrow {}^sR_b^T = \begin{bmatrix} \cos^2(t) & \sin(t) & -\cos(t)\sin(t) \\ -\cos(t)\sin(t) & \cos(t) & \sin^2(t) \\ \sin(t) & 0 & \cos(t) \end{bmatrix}$$

```
sRb_dot = [-2*cos(t)*sin(t), (sin(t)^(2))-(cos(t)^(2)), cos(t);
           cos(t), -sin(t), 0;
           (sin(t)^(2))-(cos(t)^(2)), 2*cos(t)*sin(t), -sin(t)]
```

sRb_dot =

$$\begin{pmatrix} -2\cos(t)\sin(t) & \sin(t)^2-\cos(t)^2 & \cos(t) \\ \cos(t) & -\sin(t) & 0 \\ \sin(t)^2-\cos(t)^2 & 2\cos(t)\sin(t) & -\sin(t) \end{pmatrix}$$

```
sRb_trans = [(cos(t)^(2)), sin(t), -cos(t)*sin(t);
             -cos(t)*sin(t), cos(t), (sin(t)^(2));
             sin(t), 0, cos(t)]
```

sRb_trans =

$$\begin{pmatrix} \cos(t)^2 & \sin(t) & -\cos(t)\sin(t) \\ -\cos(t)\sin(t) & \cos(t) & \sin(t)^2 \\ \sin(t) & 0 & \cos(t) \end{pmatrix}$$

```
S_s_w_sb = simplify(sRb_dot*sRb_trans)
```

S_s_w_sb =

$$\begin{pmatrix} 0 & -\cos(t) & 1 \\ \cos(t) & 0 & -\sin(t) \\ -1 & \sin(t) & 0 \end{pmatrix}$$

```
% Extract components of the angular velocity vector
w = [S_s_w_sb(3, 2); S_s_w_sb(1, 3); S_s_w_sb(2, 1)]
```

w =

$$\begin{pmatrix} \sin(t) \\ 1 \\ \cos(t) \end{pmatrix}$$

Therefore this is the angualr velocity we get in rad/s.

c)

Linear Velocity can easily be calculated, as it is the derivative of the position vector.

* As in a Transformation matrix, 4th column denotes the position vector;
* ${}^sV_{sb}$ is derivative of 4th column of ${}^sT_b$.
* ${}^sV_{sb} = {}^s\dot{O}_b$

$$\Rightarrow {}^sO_b = \begin{bmatrix} 10\cos^2(t) - 20\sin(t) \\ 10\sin(t) \\ -20\cos(t) - 10\cos(t)\sin(t) \end{bmatrix}$$

$$\Rightarrow {}^s\dot{O}_b = {}^sV_{sb} = \begin{bmatrix} -20\cos(t) - 10\sin(2t) \\ 10\cos(t) \\ 20\sin(t) - 10 + 20\sin^2(t) \end{bmatrix}$$

d)

Spatial Velocity vector in {S} frame would be;

$$\begin{bmatrix} {}^{s}V_{sb} \\ {}^{s}\omega_{sb} \end{bmatrix}$$

* However, we need this vector in {B} frame, so we need;

${}^{b}V_{sb}$ and ${}^{b}\omega_{sb}$.

$\Rightarrow {}^{b}V_{sb} = {}^{b}R_{s} \, {}^{s}V_{sb}$

$\Rightarrow {}^{b}\omega_{sb} = {}^{b}R_{s} \, {}^{s}\omega_{sb}$

$\Rightarrow {}^{b}R_{s} = {}^{s}R_{b}^{-1} = {}^{s}R_{b}^{T}$

```
bRs = sRb_trans;
sVsb = [-20*cos(t)-10*sin(2*t); 10*cos(t); 20*sin(t)-10+20*(sin(t)^(2))]
```

sVsb =

$$\begin{pmatrix} -10\sin(2\,t) - 20\cos(t) \\ 10\cos(t) \\ 20\sin(t)^2 + 20\sin(t) - 10 \end{pmatrix}$$

```
sWsb = w
```

sWsb =

$$\begin{pmatrix} \sin(t) \\ 1 \\ \cos(t) \end{pmatrix}$$

```
bVsb = bRs*sVsb
```

bVsb =

$$\begin{pmatrix} 10\cos(t)\sin(t) - \cos(t)^2\sigma_2 - \cos(t)\sin(t)\sigma_1 \\ 10\cos(t)^2 + \sin(t)^2\sigma_1 + \cos(t)\sin(t)\sigma_2 \\ \cos(t)\sigma_1 - \sin(t)\sigma_2 \end{pmatrix}$$

where

$$\sigma_1 = 20\sin(t)^2 + 20\sin(t) - 10$$

$$\sigma_2 = 10\sin(2\,t) + 20\cos(t)$$

```
bWsb = bRs*sWsb
```

bWsb =

$$\begin{pmatrix} \sin(t) \\ \cos(t) \\ \cos(t)^2 + \sin(t)^2 \end{pmatrix}$$

```
spatial_vel = simplify([bVsb;bWsb])
```

spatial_vel =

$$\begin{pmatrix} -20\cos(t) \\ 20\sin(t) + 10 \\ -10\cos(t) \\ \sin(t) \\ \cos(t) \\ 1 \end{pmatrix}$$

e)

When the rider is positioned at the origin of the body frame, their linear velocity is denoted as ${}^{b}V_{sb}$ in the body frame coordinates, and ${}^{s}V_{sb}$ in the fixed frame coordinates.

Problem - 2

$R \in SO(3)$, $a, b \in \mathbb{R}^3$, $S \in \mathbb{R}^{3\times 3}$,

a) $\underline{R(a \times b) = Ra \times Rb}$

* Let $a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$ and $b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$

* Also let $R = [c_1 \quad c_2 \quad c_3]$;

where c_1, c_2, c_3 are columns of R matrix.

* Then, $Ra = a_1 c_1 + a_2 c_2 + a_3 c_3$

and

$$Rb = b_1 c_1 + b_2 c_2 + b_3 c_3$$

* Hence, $Ra \times Rb = (a_1 c_1 + a_2 c_2 + a_3 c_3) \times (b_1 c_1 + b_2 c_2 + b_3 c_3)$

* Using Distributive Property of Cross Product,

$$= (a_1 c_1 + a_2 c_2 + a_3 c_3) \times b_1 c_1 + (a_1 c_1 + a_2 c_2 + a_3 c_3) \times b_2 c_2 + (a_1 c_1 + a_2 c_2 + a_3 c_3) \times b_3 c_3$$

* As cross products are not commutative,

$$= -b_1 c_1 \times (a_1 c_1 + a_2 c_2 + a_3 c_3) - b_2 c_2 \times (a_1 c_1 + a_2 c_2 + a_3 c_3)$$

$-b_3 c_3 \times (a_1 c_1 + a_2 c_2 + a_3 c_3)$

* Again using distributive property,

$$
\begin{aligned}
= & -a_1 b_1 (c_1 \times c_1) - a_2 b_1 (c_1 \times c_2) - a_3 b_1 (c_1 \times c_3) \\
& -a_1 b_2 (c_2 \times c_1) - a_2 b_2 (c_2 \times c_2) - a_3 b_2 (c_2 \times c_3) \\
& -a_1 b_3 (c_3 \times c_1) - a_2 b_3 (c_3 \times c_2) - a_3 b_3 (c_3 \times c_3)
\end{aligned}
$$

* Columns of rotation matrix are orthonormal,
so $c_1 \times c_1 = 0$, $c_2 \times c_2 = 0$, $c_3 \times c_3 = 0$,
$c_1 \times c_2 = c_3$, $c_1 \times c_3 = c_2$, $c_2 \times c_3 = c_1$

* Therefore,

$$
\begin{aligned}
&= -a_2 b_1 c_3 + a_3 b_1 c_2 + a_1 b_2 c_3 - a_3 b_2 c_1 - a_1 b_3 c_2 + a_2 b_3 c_1 \\
&= (a_2 b_3 - a_3 b_2) c_1 + (a_3 b_1 - a_1 b_3) c_2 + (a_1 b_2 - a_2 b_1) c_3
\end{aligned}
$$

$$= \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix} \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$$

* Which is equal to;

$$R(a \times b)$$

b) $\underline{RS(a)R^T = S(Ra)}$

* Let b, be a 3×1 vector, then;

$$\begin{aligned} S(Ra)b &= Ra \times b \\ &= Ra \times (RR^{-1})b \\ &= Ra \times (RR^T)b \\ &= Ra \times R(R^T b) \end{aligned}$$

* Using the Property proved in part (a);

$$\Rightarrow S(Ra)b = R(a \times R^T b)$$
$$\Rightarrow S(Ra)b = RS(a)R^T b$$

* Since this holds for arbitrary vector b,

$$\Rightarrow S(Ra) = RS(a)R^T$$

Problem - 3

a)

We start off with the DH Parameters of the UR5e robot.

DH Parameters:

Link	a_i	α_i	d_i	θ_i
1	0	$\pi/2$	162.5	θ_1
2	425	0	0	θ_2
3	392.2	0	0	θ_3
4	0	$\pi/2$	133	θ_4
5	0	$-\pi/2$	97.5	θ_5
6	0	0	91.55	θ_6

We then calculate ${}^0T_6$ using this.

```
clear all; clc;
syms theta1(t) theta2(t) theta3(t) theta4(t) theta5(t) theta6(t);
alpha_values = [90, 0, 0, 90, -90, 0]; % Alpha values for each link
link_length = [0 425 392.2 0 0 0]; % Length of each link
offset_values = [162.5 0 0 133 97.5 91.55]; % Offset values
joint_angles = [theta1(t) theta2(t) theta3(t) theta4(t) theta5(t)
theta6(t)]; % Joint angle variations

T_06 = [];
% Transformation Matrices
for i = 1:6
    switch i
        case 1
            T01 =
[cos(joint_angles(i)), -sin(joint_angles(i))*cosd(alpha_values(i)),
sind(alpha_values(i))*sin(joint_angles(i)),
link_length(i)*cos(joint_angles(i)); ...
                sin(joint_angles(i)),
cos(joint_angles(i)).*cosd(alpha_values(i)),
-sind(alpha_values(i))*cos(joint_angles(i)),
sin(joint_angles(i))*link_length(i); ...
```

```
                0, sind(alpha_values(i)), cosd(alpha_values(i)),
offset_values(i); ...
                0, 0, 0, 1];
        case 2
            T12 =
[cos(joint_angles(i)), -sin(joint_angles(i))*cosd(alpha_values(i)),
sind(alpha_values(i))*sin(joint_angles(i)),
link_length(i)*cos(joint_angles(i)); ...
                sin(joint_angles(i)),
cos(joint_angles(i)).*cosd(alpha_values(i)),
-sind(alpha_values(i))*cos(joint_angles(i)),
sin(joint_angles(i))*link_length(i); ...
                0, sind(alpha_values(i)), cosd(alpha_values(i)),
offset_values(i); ...
                0, 0, 0, 1];
        case 3
            T23 =
[cos(joint_angles(i)), -sin(joint_angles(i))*cosd(alpha_values(i)),
sind(alpha_values(i))*sin(joint_angles(i)),
link_length(i)*cos(joint_angles(i)); ...
                sin(joint_angles(i)),
cos(joint_angles(i)).*cosd(alpha_values(i)),
-sind(alpha_values(i))*cos(joint_angles(i)),
sin(joint_angles(i))*link_length(i); ...
                0, sind(alpha_values(i)), cosd(alpha_values(i)),
offset_values(i); ...
                0, 0, 0, 1];
        case 4
            T34 =
[cos(joint_angles(i)), -sin(joint_angles(i))*cosd(alpha_values(i)),
sind(alpha_values(i))*sin(joint_angles(i)),
link_length(i)*cos(joint_angles(i)); ...
                sin(joint_angles(i)),
cos(joint_angles(i)).*cosd(alpha_values(i)),
-sind(alpha_values(i))*cos(joint_angles(i)),
sin(joint_angles(i))*link_length(i); ...
                0, sind(alpha_values(i)), cosd(alpha_values(i)),
offset_values(i); ...
                0, 0, 0, 1];
        case 5
            T45 =
[cos(joint_angles(i)), -sin(joint_angles(i))*cosd(alpha_values(i)),
sind(alpha_values(i))*sin(joint_angles(i)),
link_length(i)*cos(joint_angles(i)); ...
                sin(joint_angles(i)),
cos(joint_angles(i)).*cosd(alpha_values(i)),
-sind(alpha_values(i))*cos(joint_angles(i)),
sin(joint_angles(i))*link_length(i); ...
                0, sind(alpha_values(i)), cosd(alpha_values(i)),
offset_values(i); ...
```

```
                0, 0, 0, 1];
        case 6
            T56 =
[cos(joint_angles(i)), -sin(joint_angles(i))*cosd(alpha_values(i)),
sind(alpha_values(i))*sin(joint_angles(i)),
link_length(i)*cos(joint_angles(i)); ...
                sin(joint_angles(i)),
cos(joint_angles(i)).*cosd(alpha_values(i)),
-sind(alpha_values(i))*cos(joint_angles(i)),
sin(joint_angles(i))*link_length(i); ...
                0, sind(alpha_values(i)), cosd(alpha_values(i)),
offset_values(i); ...
                0, 0, 0, 1];
            T06 = T01 * T12 * T23 * T34 * T45 * T56;
    end
end

% Approximate expressions with variable precision arithmetic
precision = 2; % Adjust precision as needed
T01 = simplify(T01)
```

T01 =

$$\begin{pmatrix} \cos(\theta_1(t)) & 0 & \sin(\theta_1(t)) & 0 \\ \sin(\theta_1(t)) & 0 & -\cos(\theta_1(t)) & 0 \\ 0 & 1 & 0 & \frac{325}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T12 = simplify(T12)
```

T12 =

$$\begin{pmatrix} \cos(\theta_2(t)) & -\sin(\theta_2(t)) & 0 & 425\cos(\theta_2(t)) \\ \sin(\theta_2(t)) & \cos(\theta_2(t)) & 0 & 425\sin(\theta_2(t)) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T23 = simplify(T23)
```

T23 =

$$\begin{pmatrix} \cos(\theta_3(t)) & -\sin(\theta_3(t)) & 0 & \frac{1961\cos(\theta_3(t))}{5} \\ \sin(\theta_3(t)) & \cos(\theta_3(t)) & 0 & \frac{1961\sin(\theta_3(t))}{5} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T34 = simplify(T34)
```

T34 =

$$\begin{pmatrix} \cos(\theta_4(t)) & 0 & \sin(\theta_4(t)) & 0 \\ \sin(\theta_4(t)) & 0 & -\cos(\theta_4(t)) & 0 \\ 0 & 1 & 0 & 133 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T45 = simplify(T45)
```

T45 =

$$\begin{pmatrix} \cos(\theta_5(t)) & 0 & -\sin(\theta_5(t)) & 0 \\ \sin(\theta_5(t)) & 0 & \cos(\theta_5(t)) & 0 \\ 0 & -1 & 0 & \frac{195}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T56 = simplify(T56)
```

T56 =

$$\begin{pmatrix} \cos(\theta_6(t)) & -\sin(\theta_6(t)) & 0 & 0 \\ \sin(\theta_6(t)) & \cos(\theta_6(t)) & 0 & 0 \\ 0 & 0 & 1 & \frac{1831}{20} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T06 = simplify(T06)
```

T06 =

$$\left(\begin{matrix} \cos(\theta_6(t))\,\sigma_3 - \cos(\theta_1(t))\sin(\theta_6(t))\sin(\sigma_4) & -\sin(\theta_6(t))\,\sigma_3 - \cos(\theta_1(t))\cos(\theta_6(t))\sin(\sigma_4) & \cos(\theta_5(t \\ -\cos(\theta_6(t))\,\sigma_2 - \sin(\theta_1(t))\sin(\theta_6(t))\sin(\sigma_4) & \sin(\theta_6(t))\,\sigma_2 - \cos(\theta_6(t))\sin(\theta_1(t))\sin(\sigma_4) & -\cos(\theta_1(\\ \sin(\theta_6(t))\cos(\sigma_4) + \cos(\theta_5(t))\cos(\theta_6(t))\sin(\sigma_4) & \cos(\theta_6(t))\cos(\sigma_4) - \cos(\theta_5(t))\sin(\theta_6(t))\sin(\sigma_4) & \\ 0 & 0 & \end{matrix} \right.$$

where

$$\sigma_1 = \theta_2(t) + \theta_3(t)$$

$$\sigma_2 = \cos(\theta_1(t))\sin(\theta_5(t)) - \cos(\theta_5(t))\sin(\theta_1(t))\cos(\sigma_4)$$

$$\sigma_3 = \sin(\theta_1(t))\sin(\theta_5(t)) + \cos(\theta_1(t))\cos(\theta_5(t))\cos(\sigma_4)$$

$$\sigma_4 = \theta_2(t) + \theta_3(t) + \theta_4(t)$$

```
Pos_vector = simplify(T06(1:3,4))
```

```
Pos_vector =
```

$$\left(\begin{array}{r} 133\sin(\theta_1(t)) + \frac{1831\cos(\theta_5(t))\sin(\theta_1(t))}{20} + 425\cos(\theta_1(t))\cos(\theta_2(t)) + \frac{1961\cos(\theta_1(t))\cos(\theta_2(t))\cos(\theta_3(t))}{5} \\ 425\cos(\theta_2(t))\sin(\theta_1(t)) - 133\cos(\theta_1(t)) - \frac{1831\cos(\theta_1(t))\cos(\theta_5(t))}{20} + \frac{1961\cos(\theta_2(t))\cos(\theta_3(t))\sin(\theta_1(t)}{5} \\ 425\sin(\theta_2(t)) + \frac{1961\sin(\sigma_2)}{5} - \frac{195\cos(\theta_4(t))\cos(\sigma_2)}{2} + \frac{19}{} \end{array}\right.$$

where

$$\sigma_1 = \cos(\theta_2(t) + \theta_3(t) + \theta_4(t))$$

$$\sigma_2 = \theta_2(t) + \theta_3(t)$$

```
% Calculate Jacobian Matrix
J1 = diff(Pos_vector, theta1(t));
J2 = diff(Pos_vector, theta2(t));
J3 = diff(Pos_vector, theta3(t));
J4 = diff(Pos_vector, theta4(t));
J5 = diff(Pos_vector, theta5(t));
J6 = diff(Pos_vector, theta6(t));

% Combine into a Jacobian matrix
Jv1 = simplify(expand([J1, J2, J3, J4, J5, J6]))
```

```
Jv1 =
```

$$\left(\begin{array}{l} 133\cos(\theta_1(t)) - 425\cos(\theta_2(t))\sin(\theta_1(t)) + \frac{1831\cos(\theta_1(t))\cos(\theta_5(t))}{20} - \frac{1961\cos(\theta_2(t))\cos(\theta_3(t))\sin(\theta_1(t}{5} \\ 133\sin(\theta_1(t)) + \frac{1831\cos(\theta_5(t))\sin(\theta_1(t))}{20} + 425\cos(\theta_1(t))\cos(\theta_2(t)) + \frac{1961\cos(\theta_1(t))\cos(\theta_2(t))\cos(\theta_3(t))}{5} \\ \end{array} \right.$$

where

$$\sigma_1 = 8500\sin(\theta_2(t)) + \sigma_{10} - 1950\cos(\sigma_{11}) + 7844\sin(\sigma_8) - \sigma_9$$

$$\sigma_2 = \frac{1831\sin(\sigma_{12})}{40}$$

$$\sigma_3 = \frac{1831\sin(\sigma_{13})}{40}$$

$$\sigma_4 = \frac{195\sin(\sigma_{11})}{2}$$

$$\sigma_5 = \frac{1961\cos(\sigma_8)}{5}$$

$$\sigma_6 = \sigma_{10} - 1950\cos(\sigma_{11}) + 7844\sin(\sigma_8) - \sigma_9$$

$$\sigma_7 = 1950\cos(\sigma_{11}) - \sigma_{10} + \sigma_9$$

$$\sigma_8 = \theta_2(t) + \theta_3(t)$$

$$\sigma_9 = \frac{1831\cos(\sigma_{12})}{2}$$

$$\sigma_{10} = \frac{1831\cos(\sigma_{13})}{2}$$

$$\sigma_{11} = \theta_2(t) + \theta_3(t) + \theta_4(t)$$

$$\sigma_{12} = \theta_2(t) + \theta_3(t) + \theta_4(t) - \theta_5(t)$$

$$\sigma_{13} = \theta_2(t) + \theta_3(t) + \theta_4(t) + \theta_5(t)$$

b)

All the joints in UR5e are revolute, therefore ith column of J is:

$$J_i = \begin{bmatrix} {}^0Z_{i-1} \times ({}^0P_n - {}^0P_{i-1}) \\ {}^0Z_{i-1} \end{bmatrix}$$

* $^0Z_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

* Determine $^0T_{i-1} = A_1 A_2 \dots A_{i-1}$
  → $^0Z_{i-1}$ is first three elements of 3rd column of $^0T_{i-1}$

* $^0O_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

* Determine ^0T_j
  → ^0O_j is first 3 elements of 4th column of ^0T_j.

```
% Calculate Transformation Matrices
T02 = simplify(T01 * T12);
T03 = simplify(T01 * T12 * T23);
T04 = simplify(T01 * T12 * T23 * T34);
T05 = simplify(T01 * T12 * T23 * T34 * T45);

% Define z-axis vectors for each frame
z00 = [0; 0; 1];
z01 = T01(1:3, 3);
z02 = T02(1:3, 3);
z03 = T03(1:3, 3);
z04 = T04(1:3, 3);
z05 = T05(1:3, 3);

% Calculate partial Jacobians
j1 = cross(z00, (T06(1:3, 4) - [0; 0; 0]));
j2 = cross(z01, (T06(1:3, 4) - T01(1:3, 4)));
j3 = cross(z02, (T06(1:3, 4) - T02(1:3, 4)));
j4 = cross(z03, (T06(1:3, 4) - T03(1:3, 4)));
j5 = cross(z04, (T06(1:3, 4) - T04(1:3, 4)));
j6 = cross(z05, (T06(1:3, 4) - T05(1:3, 4)));
```

```
% Combine partial Jacobians into the full Jacobian matrix
Jv2 = simplify(expand([j1, j2, j3, j4, j5, j6]))
```

```
if (Jv1 == Jv2)
    disp("The Matrices got from both the methods are same")
else
    disp("The Matrices got from both the methods are not same")
end
```

```
The Matrices got from both the methods are same
```

c)

ith column of J is:

$$J_i = \begin{bmatrix} {}^0Z_{i-1} \times ({}^0P_n - {}^0P_{i-1}) \\ {}^0Z_{i-1} \end{bmatrix}$$

$$* \quad J = \begin{bmatrix} J_v \\ J_w \end{bmatrix}$$

$$* \quad J_w = [{}^0Z_{i-1}]$$

```
Jw = [z00 z01 z02 z03 z04 z05]
```

Jw =

$$\begin{pmatrix} 0 & \sin(\theta_1(t)) & \sin(\theta_1(t)) & \sin(\theta_1(t)) & \cos(\theta_1(t))\sin(\sigma_2) & \cos(\theta_5(t))\sin(\theta_1(t)) - \cos(\theta_1(t))\sin(\theta_5(t))\cos(\sigma_2) \\ 0 & \sigma_1 & \sigma_1 & \sigma_1 & \sin(\theta_1(t))\sin(\sigma_2) & -\cos(\theta_1(t))\cos(\theta_5(t)) - \sin(\theta_1(t))\sin(\theta_5(t))\cos(\sigma_2 \\ 1 & 0 & 0 & 0 & -\cos(\sigma_2) & -\sin(\theta_5(t))\sin(\sigma_2) \end{pmatrix}$$

where

$$\sigma_1 = -\cos(\theta_1(t))$$

$$\sigma_2 = \theta_2(t) + \theta_3(t) + \theta_4(t)$$

d)

We have calculated J_v in parts (a) and (b), and also J_w in part (c).

The overall Jacobian Matrix J;

$$J = \begin{bmatrix} J_v \\ J_w \end{bmatrix}.$$

```
J = simplify([Jv1 ; Jw])
```

```
J =
```

$$\left(\begin{array}{l} 133\cos(\theta_1(t)) - 425\cos(\theta_2(t))\sin(\theta_1(t)) + \frac{1831\cos(\theta_1(t))\cos(\theta_5(t))}{20} - \frac{1961\cos(\theta_2(t))\cos(\theta_3(t))\sin(\theta_1(t}{5} \\ 133\sin(\theta_1(t)) + \frac{1831\cos(\theta_5(t))\sin(\theta_1(t))}{20} + 425\cos(\theta_1(t))\cos(\theta_2(t)) + \frac{1961\cos(\theta_1(t))\cos(\theta_2(t))\cos(\theta_3(t))}{5} \end{array} \right.$$

where

$$\sigma_1 = -\cos(\theta_1(t))$$

$$\sigma_2 = 8500\sin(\theta_2(t)) + \sigma_{11} - 1950\cos(\sigma_{12}) + 7844\sin(\sigma_9) - \sigma_{10}$$

$$\sigma_3 = \sigma_{11} - 1950\cos(\sigma_{12}) + 7844\sin(\sigma_9) - \sigma_{10}$$

$$\sigma_4 = \frac{1831\sin(\sigma_{13})}{40}$$

$$\sigma_5 = \frac{1831\sin(\sigma_{14})}{40}$$

$$\sigma_6 = \frac{1961\cos(\sigma_9)}{5}$$

$$\sigma_7 = \frac{195\sin(\sigma_{12})}{2}$$

$$\sigma_8 = 1950\cos(\sigma_{12}) - \sigma_{11} + \sigma_{10}$$

$$\sigma_9 = \theta_2(t) + \theta_3(t)$$

$$\sigma_{10} = \frac{1831\cos(\sigma_{13})}{2}$$

$$\sigma_{11} = \frac{1831\cos(\sigma_{14})}{2}$$

$$\sigma_{12} = \theta_2(t) + \theta_3(t) + \theta_4(t)$$

$$\sigma_{13} = \theta_2(t) + \theta_3(t) + \theta_4(t) - \theta_5(t)$$

$$\sigma_{14} = \theta_2(t) + \theta_3(t) + \theta_4(t) + \theta_5(t)$$

e)

```
simplify(det(J))
```

ans =

$$-\frac{33337\sin(\theta_5(t))\left(2125\sin(\theta_2(t)-\theta_3(t))-1961\sin(\theta_2(t)+2\,\theta_3(t))+1961\sin(\theta_2(t))-\frac{975\cos(\theta_2(t)+\theta}{2}\right.}{2}$$

To identify the singular points of a mechanism, we analyze the determinant of its Jacobian matrix. When the determinant equals zero, it indicates that the Jacobian has become singular, meaning it has lost its full rank.

First Case:

$\frac{33337}{2}\sin(\theta_5)$ = 0;

θ_5 = 0, $+n\pi$, $-n\pi$ where n is any postive real integer number

Second Case:

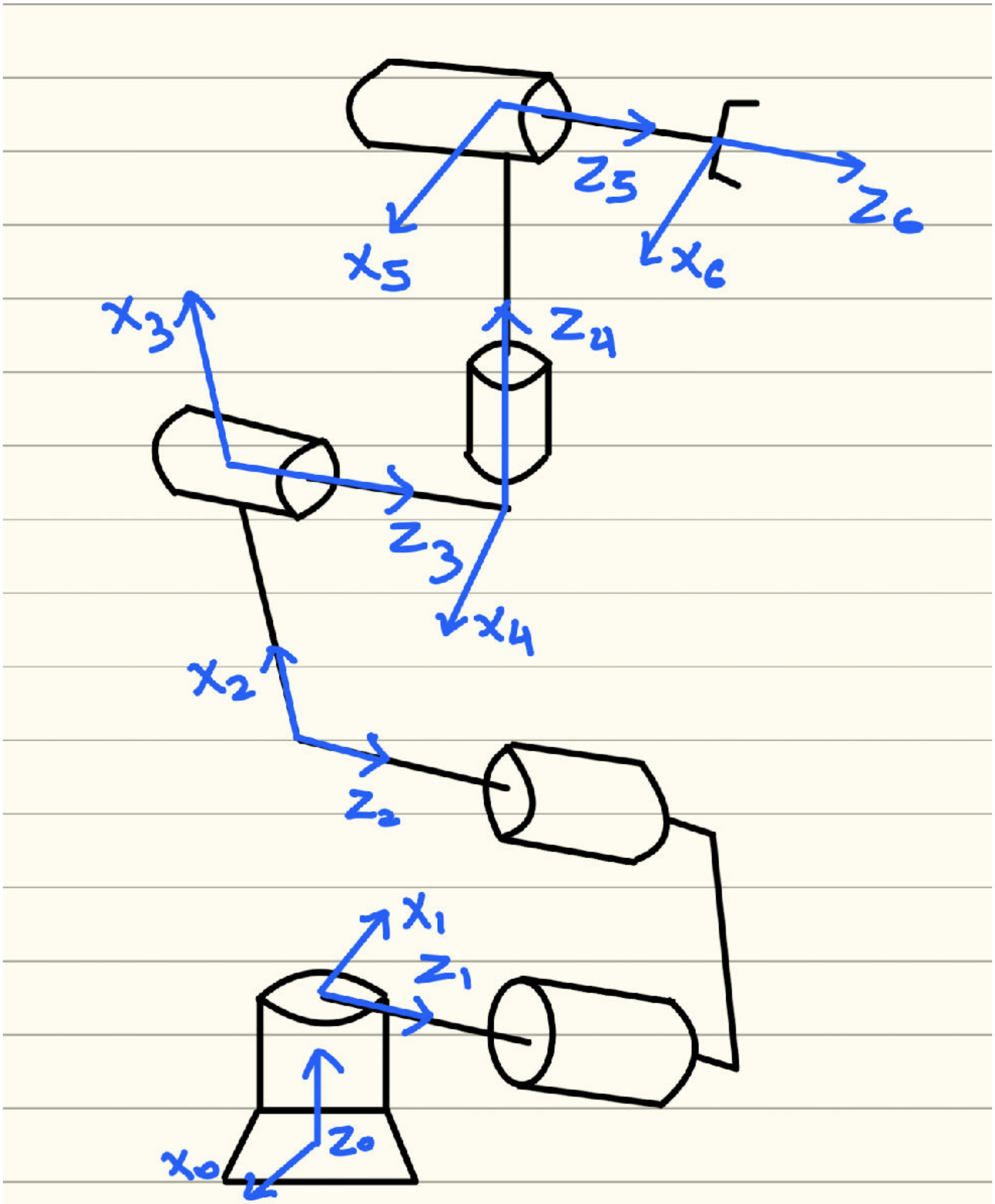
$$\left(2125\sin(\theta_2-\theta_3)-1961\sin(\theta_2+2\,\theta_3)+\frac{975\cos(\theta_2+2\,\theta_3+\theta_4)}{2}-\frac{975\cos(\theta_2+\theta_4)}{2}-2125\sin(\theta_2+\theta_3)+1961\sin(\theta_2)\right)$$

= 0;

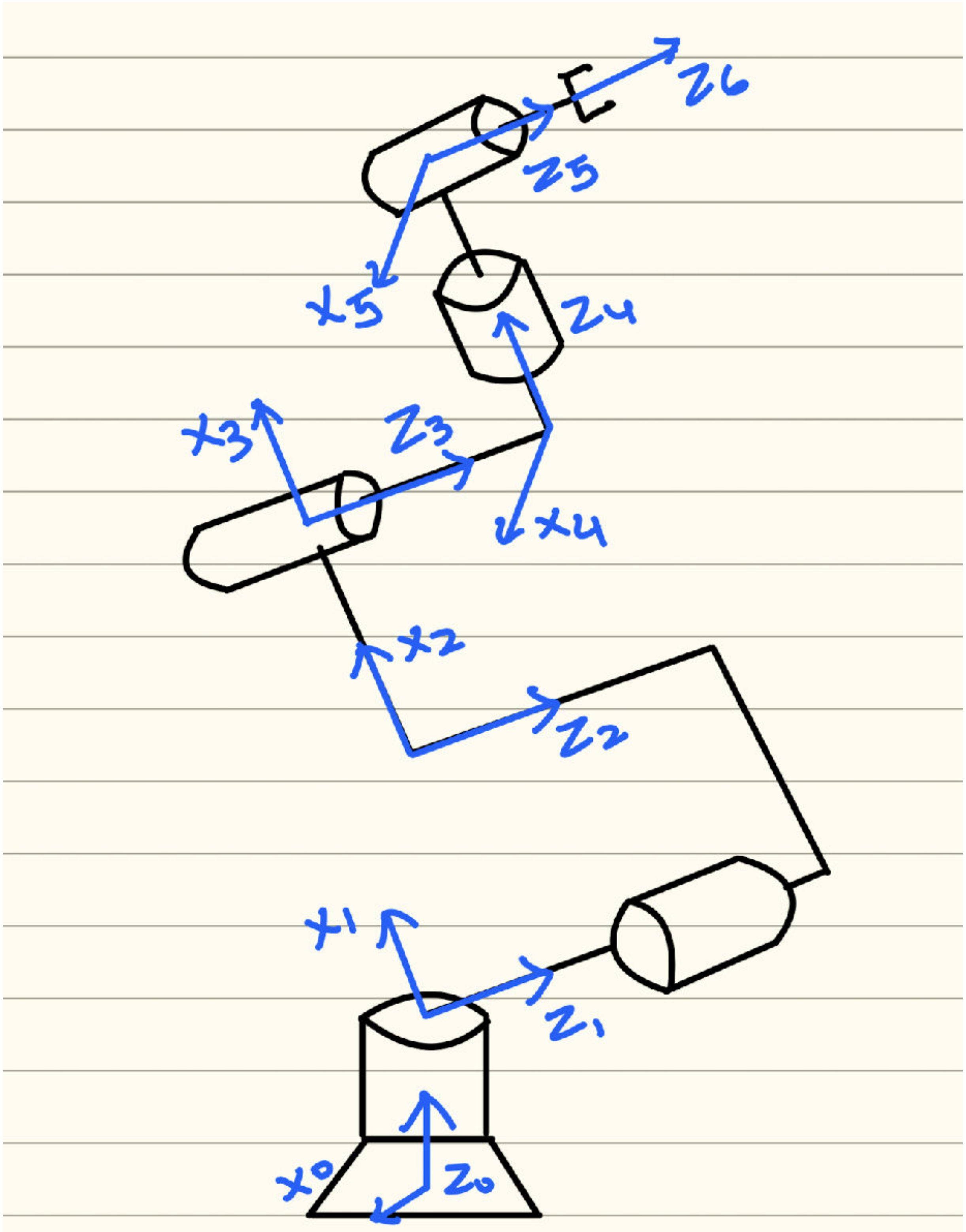
If θ_3 equals 0, all the terms cancel each other out, resulting in 0. Therefore, the arm attains a singular configuration when either θ_5 equals 0 or θ_3 equals 0.

f)

In the initial scenario described, when θ_5 equals zero, the alignment of z5 with z3 results in collinearity, preventing the robot from attaining velocity along the directions of z3, z5, and z6.



In the second scenario depicted below, when $\theta_3 = 0$, z2 and z3 align in a straight line, causing the robot to be unable to attain velocity in the z2 and z3 directions.



Z6
Z5
X5
Z4
X3
Z3
X4
X2
Z2
X1
Z1
X0
Z0

Introduction to Robotics - L1

Huzaifah and Asghar

QUESTION 4:

Answer the following questions individually:

Problem 4
CLO2-C2

5+10 points

(a) How many hours did each of you spend on this homework and specifically state your contribution in this homework assignment? Answer as accurately as you can, as this will be used to structure next year's class.

Asghar:
I spent 12 hours on this homework, spending half the time in-person on campus and other half on a discord call with Huzaifah solving the homework.

Huzaifah:
I spent 12 hours on this homework, with half the time in-person on campus and other half on a discord call with Asghar solving this homework

(b) Do you have any specific advice for students attempting this homework next year?

My advice for students next year is to thoroughly review concepts in velocity kinematics and manipulator Jacobians before attempting this homework. Understanding how to calculate angular and linear velocities, derive Jacobian matrices, and analyze singular configurations is crucial.

(c) **This question has been revised compared to the previous homework assignments.** Each group member is to provide their reflections as answers to each of the following questions. You are expected to be precise in your responses.

1. Explain each of the outcomes, stated at the beginning of this document, in your own words.
2. Why is it important for you to achieve each of the outcomes in relation to understanding or building any robot?
3. What do you currently understand about content related to these outcomes? Do you have unanswered questions?
4. Have you achieved these outcomes? What went wrong? How will you enable yourself to achieve these outcomes? What could you do to know more or enhance your skills in this context?

Asghar's Response:

1. **Determine the angular velocities of rigid bodies in motion**: This outcome means that after completing the homework, we are able to calculate how fast a solid object is rotating. This is typically measured in radians per second.

   Determine the Jacobian of manipulators: The Jacobian is a mathematical function that helps us understand how small changes in the joint angles can affect the position and orientation of the end-effector (the tool or hand at the end of a robotic arm). After this homework, we are able to calculate this

Jacobian for a given robotic manipulator.

Determine the singularities of manipulators, allowing singularity decoupling: In robotics, a singularity refers to a configuration of the robot where the control of the end-effector's motion may be lost, or where the robot may exert an infinite amount of force. After this homework, we are able to identify these singularities. Singularity decoupling is a method to separate the position and orientation control of a robot manipulator, which can help in avoiding singularities.

2. In robotics, grasping the concept of angular velocities is key to controlling and manipulating robotic systems effectively. It's all about understanding how changes in joint angles affect the position and orientation of the end-effector. This knowledge comes in handy across various fields like vehicle dynamics, power systems, and even human motion analysis, helping us design robots that are not only efficient but also safe to work with. Similarly, the Jacobian matrix plays a crucial role in controlling manipulators. It helps us map velocities between different spaces, like between joints and Cartesian coordinates, which is super important for tasks that require precise control over the robot's movement, such as welding or cutting. And let's not forget about dealing with singularities - those tricky situations where robots lose some of their motion abilities. Understanding singularities and how to handle them is essential for making sure our robots work reliably and safely. So, learning about all these concepts not only helps us build better robots but also ensures they operate smoothly in real-world environments.

3. I have a solid understanding of the concepts related to velocity kinematics and manipulator Jacobians. However, I still have some unanswered questions regarding practical implementation and real-world applications of these concepts.
4. While I have achieved a good grasp of the theoretical aspects, I need to enhance my practical skills by working on more robotics problems, collaborating with peers, and seeking guidance from professors. Continuous practice and exploration of real-world examples will help me further strengthen my skills in this context.

Huzaifah's Response in relation to (1) and (2):

Determine the angular velocities of rigid bodies in motion: Angular velocity is a measure of the rate of change of an angle with respect to time. In the context of robotics, it is often used to describe the rotational speed of a rigid body such as a robot arm or a joint. This is crucial in understanding and controlling the motion of robots. For instance, in a robotic arm, each joint may rotate at a certain angular velocity, which in turn influences the overall movement of the arm. With the help of practice due to this homework, I am able to calculate the angular velocities of such rigid bodies in motion, which is a fundamental aspect of robot kinematics.

Determine the Jacobian of manipulators: The Jacobian matrix is a mathematical construct that provides a map between the velocities in the joint space (the speeds at which the joints are moving) and the velocities in the Cartesian space (the speed at which the end-effector is moving in 3D space). It is a key concept in the control of robotic manipulators, as it allows us to understand how changes in the joint angles will affect the position and orientation of the end-effector. WIth the help of practice due to this homework, I am able to calculate the Jacobian for a given robotic manipulator, which is a critical step in designing control systems for robots.

Determine the singularities of manipulators, allowing singularity decoupling: Singularities in robotics refer to configurations where the robot loses its ability to move in certain directions or where the control of the end-effector's motion may be lost. They are usually associated with the Jacobian matrix becoming rank-deficient, meaning that it loses one or more of its rows or columns of linear independence. Singularity decoupling is a method that separates the position and orientation control of a robot manipulator, which can help in avoiding singularities. With the help of this homework, I am able to identify these singularities and apply singularity decoupling techniques to avoid them, which is crucial for safe and effective robot operation.

Achieving these outcomes is crucial for effectively designing and controlling robots. Understanding frame configurations and velocities facilitates accurate motion planning and control, while Jacobian analysis aids in optimizing robot performance and avoiding singularities.

3. I have a solid grasp of the theoretical aspects of velocity kinematics and manipulator Jacobians. However, I still have questions regarding their practical implementation and real-world applications.

4. While I have achieved a good understanding of the theoretical concepts, I need to enhance my practical skills through more problem-solving practice, collaboration with peers, and seeking guidance from professors. Engaging in hands-on projects and exploring real-world examples will further deepen my knowledge and skills in this area.