MTRN4230 - Project 1

Weichen Tie (z5308889)

T2 July 2024

Contents

1 Part A: Dynamic forward kinematics

You do not need to include anything in your report for this practical part of the assessment.

2 Part B: UR5e modelling

The DH table for the UR5e robot arm is as provided:

	theta (rad)	**a (m)**	**d (m)**	**alpha (rad)**
Joint 1	0	0	0.1625	$\pi/2$
Joint 2	0	-0.425	0	0
Joint 3	0	-0.3922	0	0
Joint 4	0	0	0.1333	$\pi/2$
Joint 5	0	0	0.0997	$-\pi/2$
Joint 6	0	0	0.0996	0

Table 1: The DH table for the UR5e robot arm

The home joint configuration (in degrees): [0.00, -75.00, 90.00, -105.00, -90.00, 0]

2.1 Manual Calculation of Forward Kinematic Solutions

2.1.1 Resultant Matrix and Output Pose

The resultant matrix derived for the home joint configuration is (in millimeters and radians):

$$ {}^{0}T_{6} = \begin{bmatrix} 0 & 1.00 & 0 & -588.53 \\ 1.00 & 0 & 0 & -133.30 \\ 0 & 0 & -1.00 & 371.91 \\ 0 & 0 & 0 & 1.00 \end{bmatrix} $$

The resultant posed derived for the home joint configuration is (in millimeters and radians):

$$ \begin{bmatrix} -588.5342 & -133.3000 & 371.9096 & 3.1416 & 0 & 1.5708 \end{bmatrix} $$

However upon cross inspection with the simulation, the 5th joint had an angle of -3.1416, but this is equivalent as it is a phase difference of 2 π.

2.1.2 Full Written Working

From Table 1, we can derive the following DH table for the home joint configuration:

	theta (rad)	**a (m)**	**d (m)**	**alpha (rad)**
Joint 1	0	0	0.1625	1.5708
Joint 2	1.3439	-0.425	0	0
Joint 3	1.5708	-0.3922	0	0
Joint 4	-1.8326	0	0.1333	1.5708
Joint 5	-1.5708	0	0.0997	-1.5708
Joint 6	0	0	0.0996	0

Table 2: The DH table for the UR5e robot arm at home joint configuration

From first principles, the homogenous transformation matrix (${}^{n-1}T_{n}$) can be derived as follows:

$$
\begin{aligned}
{}^{n-1}T_n &= Rot_{z,\theta_n} Trans_{z,d_n} Trans_{x,a_n} Rot_{x,\alpha_n} \\
&= \begin{bmatrix} \cos(\theta_n) & -\sin(\theta_n) & 0 & 0 \\ \sin(\theta_n) & \cos(\theta_n) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_n \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \cos(\alpha_n) & -\sin(\alpha_n) & 0 \\ 0 & \sin(\alpha_n) & \cos(\alpha_n) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos(\theta_n) & -\sin(\theta_n)\cos(\alpha_n) & \sin(\theta_n)\sin(\alpha_n) & a_n\cos(\theta_n) \\ \sin(\theta_n) & \cos(\theta_n)\cos(\alpha_n) & -\cos(\theta_n)\sin(\alpha_n) & a_n\sin(\theta_n) \\ 0 & \sin(\alpha_n) & \cos(\alpha_n) & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{1}
$$

Substituting each parameter with its corresponding joint values in the DH table in Table 2 to the joint-to-joint transformation matrix in Equation (1) will yield us the following matrices:

$$
{}^{0}T_1 = \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & 0 & -1.00 & 0 \\ 0 & 1.00 & 0 & 162.50 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix}
$$

$$
{}^{1}T_2 = \begin{bmatrix} 0.26 & -0.97 & 0 & -110.00 \\ 0.97 & 0.26 & 0 & -410.52 \\ 0 & 0 & 1.00 & 0 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{2}T_3 = \begin{bmatrix} 0 & -1.00 & 0 & 0 \\ 1.00 & 0 & 0 & -392.20 \\ 0 & 0 & 1.00 & 0 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{3}T_4 = \begin{bmatrix} -0.26 & 0 & -0.97 & 0 \\ -0.97 & 0 & 0.26 & 0 \\ 0 & 1.0000 & 0 & 133.30 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{4}T_5 = \begin{bmatrix} 0 & 0 & 1.00 & 0 \\ -1.00 & 0 & 0 & 0 \\ 0 & -1.00 & 0 & 99.70 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{5}T_6 = \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & 1.00 & 0 & 0 \\ 0 & 0 & 1.00 & 99.60 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

And since coordinate frames can be compounded through the relationship ${}^{A}T_C = {}^{A}T_B {}^{B}T_C$, we

can derive the resultant homogenous matrix ${}^0T_6$,

$$
\begin{aligned}
{}^0T_6 &= {}^0T_1\,{}^1T_2\,{}^2T_3\,{}^3T_4\,{}^4T_5\,{}^5T_6 \\
{}^0T_6 &= \begin{bmatrix} 0 & 1.00 & 0 & -588.53 \\ 1.00 & 0 & 0 & -133.30 \\ 0 & 0 & -1.00 & 371.91 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
\end{aligned}
$$

To get the pose, we realise that the resultant homogenous matrix takes the form of:

$$
\left[\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right]
$$

So our final joint positions in millimeters are,

$$[-588.5342, -133.3000, 371.9096]$$

And our roll, pitch and yaw values in radians respectively are can be calculated using Matlab's tr2rpy function,

```
rpy = tr2rpy(R);
% rpy = [3.1416, 0, 1.5708]
```

And thus, our final pose will be,

$$[-588.5342, -133.3000, 371.9096, 3.1416, 0, 1.5708]$$

2.1.3 Intermediate Matrices

The intermediate matrices that transforms between coordinate frames from the base frame can be calculated with the help of joint-to-joint matrices from 2.1.2.

$$
{}^{0}T_{1} = \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & 0 & -1.00 & 0 \\ 0 & 1.00 & 0 & 162.50 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{0}T_{2} = {}^{0}T_{1}\,{}^{1}T_{2} = \begin{bmatrix} 0.26 & 0.97 & 0 & -110.00 \\ 0 & 0 & -1.00 & 0 \\ -0.97 & 0.26 & 0 & 573.02 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{0}T_{3} = {}^{0}T_{2}\,{}^{2}T_{3} = \begin{bmatrix} 0.97 & -0.26 & 0 & -488.83 \\ 0 & 0 & -1.00 & 0 \\ 0.26 & 0.97 & 0 & 471.51 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{0}T_{4} = {}^{0}T_{3}\,{}^{3}T_{4} = \begin{bmatrix} 0 & 0 & -1.00 & -488.83 \\ 0 & -1.00 & 0 & -133.30 \\ -1.00 & 0 & 0 & 471.51 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{0}T_{5} = {}^{0}T_{4}\,{}^{4}T_{5} = \begin{bmatrix} 0 & 1.00 & 0 & -588.53 \\ 1.00 & 0 & 0 & -133.30 \\ 0 & 0 & -1.00 & 471.51 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

$$
{}^{0}T_{6} = {}^{0}T_{5}\,{}^{5}T_{6} = \begin{bmatrix} 0 & 1.00 & 0 & -588.53 \\ 1.00 & 0 & 0 & -133.30 \\ 0 & 0 & -1.00 & 371.91 \\ 0 & 0 & 0 & 1.00 \end{bmatrix}
$$

2.1.4 Explanation of the Meaning of Calculated Matrices

2.2 Model of UR5e Robotic Arm using RVC Toolbox

Using the Matlab and RVC toolbox, we can use the following code to model the UR5e robot in the home position,

```
jointConfiguration = deg2rad([0.00, -75.00, 90.00, -105.00, -90.00, 0.00]);
L(1) = Link([0, 0.1625, 0, pi/2]); % Link 1
L(2) = Link([0, 0, -0.425, 0]); % Link 2
L(3) = Link([0, 0, -0.3922, 0]); % Link 3
L(4) = Link([0, 0.1333, 0, pi/2]); % Link 4
L(5) = Link([0, 0.0997, 0, -pi/2]); % Link 5
L(6) = Link([0, 0.0996, 0, 0]); % Link 6
% Creating the robot
robot = SerialLink(L, 'name', 'UR5e');
robot.teach(jointConfiguration)
```

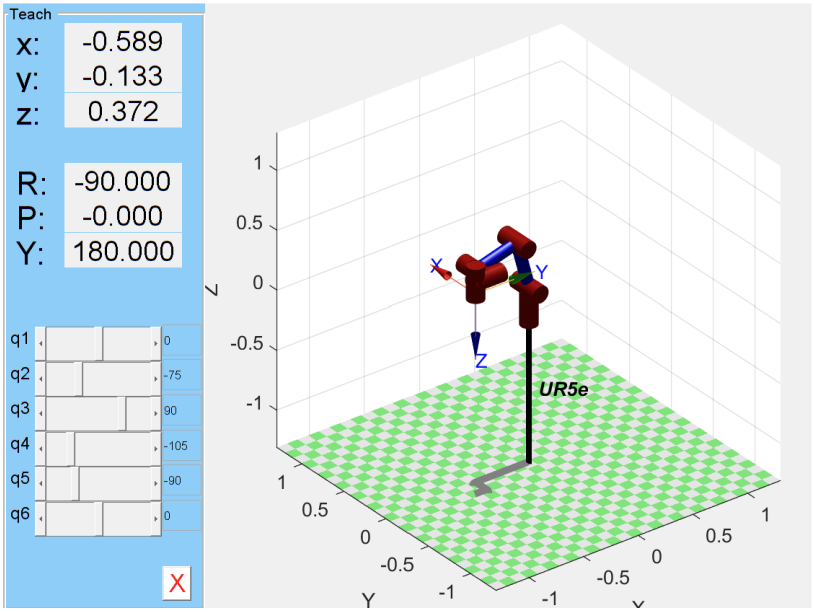

Figure 1: UR5e robot modelled in Matlab at the home position

2.2.1 Forward Kinematic Conversion to Attain Pose with Angles in RPY

By replacing the last code line with fkine(robot, jointConfiguration), we obtain the following matrix result,

$$\begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ 1 & 0 & 0 & -0.1333 \\ 0 & 0 & -1 & 0.3719 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
fkine(robot, jointConfiguration)
```

```
ans =
         0         1         0   -0.5885
         1         0         0   -0.1333
         0         0        -1    0.3719
         0         0         0         1
```

Figure 2: Matlab fkine result

The only difference between the Matlab result with the manually calculated result in section 2.1 is the final column's values being 1000 times smaller due to using meters instead of millimeters.

Once again, to get the pose, we realise that the matrix takes the form of:

$$\left[\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

So our final joint positions in meters are,

$$[-0.5885, -0.1333, 0.3719]$$

And our roll, pitch and yaw values in radians respectively are can be calculated using Matlab's tr2rpy function,

```
rpy = tr2rpy(R);
% rpy = [3.1416, 0, 1.5708]
```

And thus, our final pose will be,

$$[-0.5885, -0.1333, 0.3719, 3.1416, 0, 1.5708]$$

Which matches our manually calculated results when converted to millimeters.

2.3 Validation of Calculations

2.3.1 Screenshot Showing Pose Including the Rotation in RPY Representation

The following screenshot shows the pose of the UR5e robot in the virtual simulation with a rotation vector.

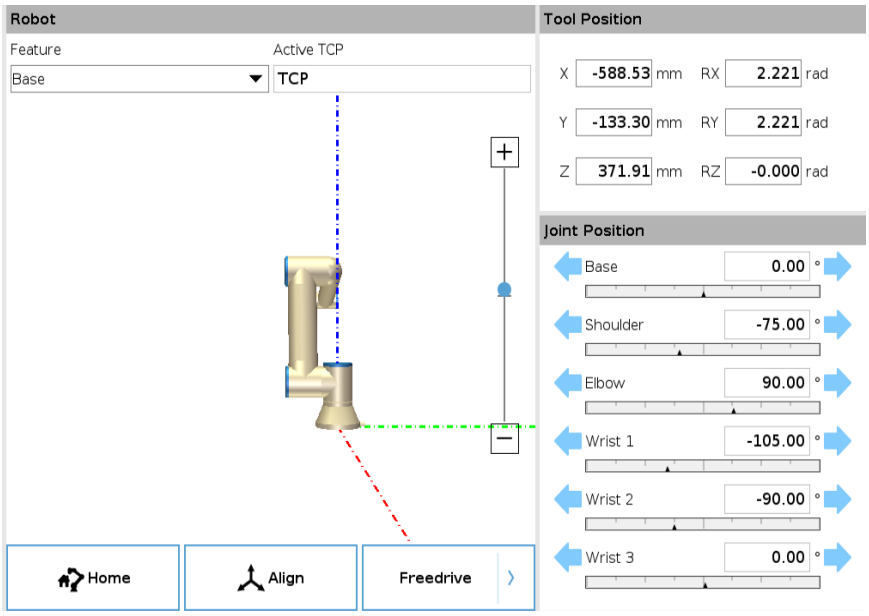

Figure 3: UR5e robot in the home position

When this rotation vector is then converted into RPY, we will get the following angles,

$$\begin{bmatrix} 3.1416 & 0 & 1.5708 \end{bmatrix}$$

Which validates the calculations performed in previous sections.

3 Part C: Robot Speed Limits

3.1 Approach to Calculation in Matlab

A method of relating the joint positions $q_1, q_2, q_3, q_4, q_5, q_6$ and joint velocities $\dot{q_1}, \dot{q_2}, \dot{q_3}, \dot{q_4}, \dot{q_5}, \dot{q_6}$ to the end effector's linear and angular velocities, $\dot{x}, \dot{y}, \dot{z}, \dot{\theta_1}, \dot{\theta_2}, \dot{\theta_3}$ respectively is through the Jacobian matrix **J** where,

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\theta_1} \\ \dot{\theta_2} \\ \dot{\theta_3} \end{pmatrix} = \mathbf{J}_{q_1,q_2,q_3,q_4,q_5,q_6} \begin{pmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \\ \dot{q_4} \\ \dot{q_5} \\ \dot{q_6} \end{pmatrix}$$

We first define the following DH Table for the UR5e robot.

	theta (rad)	**a (m)**	**d (m)**	**alpha (rad)**
Joint 1	q_1	0	0.1625	$\pi/2$
Joint 2	q_2	-0.425	0	0
Joint 3	q_3	-0.3922	0	0
Joint 4	q_4	0	0.1333	$\pi/2$
Joint 5	q_5	0	0.0997	-$\pi/2$
Joint 6	q_6	0	0.0996	0

Table 3: The DH table for the UR5e robot arm

We can define the parameters of each link in the link array using 'Links' in Matlab's RVC toolbox and then construct the articulated robot using 'SerialLink' the link array.

With the robot constructed, we will now iterate through each set of joint positions and velocities and obtaining the Jacobian matrix using the 'jacob0' function on the robot at the specified joint positions. By multiplying this Jacobian matrix with the corresponding joint velocities, it is possible to derive the linear velocities $\dot{x}$, $\dot{y}$ and $\dot{z}$ for the end effector.

It is then possible to determine the magnitude of this velocity simply through the equation,

$$|v| = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2}$$

We then update the value maximum velocity accordingly.

The Matlab code is shown below,

```
function maxLinearVelocity = calculateMaxLinearVelocity(jointPositions,jointVelocities)
    % Defining the link array in accordance with the DH table
    L(1) = Link([0, 0.1625, 0, pi/2]); % Link 1
    L(2) = Link([0, 0, -0.425, 0]); % Link 2
    L(3) = Link([0, 0, -0.3922, 0]); % Link 3
    L(4) = Link([0, 0.1333, 0, pi/2]); % Link 4
    L(5) = Link([0, 0.0997, 0, -pi/2]); % Link 5
    L(6) = Link([0, 0.0996, 0, 0]); % Link 6
    % Creating the robot
    robot = SerialLink(L, 'name', 'Articulated');
    maxLinearVelocity = 0;
    % Iterate through the jointPositions and jointVelocities arrays.
   for i = 1:length(jointPositions)
        % Calculate the Jacobian matrix
        jacobian = jacob0(robot, jointPositions(i,:));
        % Calculate tool/end effector linear velocity and angular velocities
        toolVelocity = jacobian * jointVelocities(i,:)';
        % Tool/end effector linear velocities encoded in first 3 rows of
        % vector
        linearVelocities = toolVelocity(1:3);
        % Calculate the magnitude of this velocity
        dotProduct = sum(linearVelocities .* linearVelocities);
        magnitude =sqrt(dotProduct);
        % Update the maxLinearVelocity value
        maxLinearVelocity = max(maxLinearVelocity, magnitude);
    end
end
```

3.2 Jacobian Calculation for the First Location

The velocity of the end effector at the first position can also be manually calculated similarly where its corresponding joint position can be read directly from the UR5e robot.

$$\begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix} = \begin{bmatrix} 0 & -1.124 & 1.803 & -2.250 & -1.571 & 0 \end{bmatrix}$$

We first derive the transformation matrices for each joint position ($^0T_1$, $^0T_2$, $^0T_3$, $^0T_4$, $^0T_5$, $^0T_6$) using the joint-to-joint transformation matrix,

$$^{n-1}T_n = \begin{bmatrix} \cos(q_n) & -\sin(q_n)\cos(\alpha_n) & \sin(q_n)\sin(\alpha_n) & a_n\cos(q_n) \\ \sin(q_n) & \cos(q_n)\cos(\alpha_n) & -\cos(q_n)\sin(\alpha_n) & a_n\sin(q_n) \\ 0 & \sin(\alpha_n) & \cos(\alpha_n) & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Therefore we get the following intermediate matrices,

$$
{}^0T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0.1625 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_2 = {}^0T_1\,{}^1T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0.1625 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.4319 & 0.9019 & 0 & -0.1835 \\ -0.9019 & 0.4319 & 0 & 0.3833 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} 0.4319 & 0.9019 & 0 & -0.1835 \\ 0 & 0 & -1 & 0 \\ -0.9019 & 0.4319 & 0 & 0.5458 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_3 = {}^0T_2\,{}^2T_3 = \begin{bmatrix} 0.4319 & 0.9019 & 0 & -0.1835 \\ 0 & 0 & -1 & 0 \\ -0.9019 & 0.4319 & 0 & 0.5458 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.2301 & -0.9732 & 0 & 0.0902 \\ 0.9732 & -0.2301 & 0 & -0.3817 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} 0.7784 & -0.6278 & 0 & -0.4888 \\ 0 & 0 & -1 & 0 \\ 0.6278 & 0.7784 & 0 & 0.2996 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_4 = {}^0T_3\,{}^3T_4 = \begin{bmatrix} 0.7784 & -0.6278 & 0 & -0.4888 \\ 0 & 0 & -1 & 0 \\ 0.6278 & 0.7784 & 0 & 0.2996 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.6278 & 0 & -0.7783 & 0 \\ -0.7783 & 0 & 0.6278 & 0 \\ 0 & 1 & 0 & 0.1333 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} 0 & 0 & -1 & -0.4888 \\ 0 & -1 & 0 & -0.1333 \\ -1 & 0 & 0 & 0.2996 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_5 = {}^0T_4\,{}^4T_5 = \begin{bmatrix} 0 & 0 & -1 & -0.4888 \\ 0 & -1 & 0 & -0.1333 \\ -1 & 0 & 0 & 0.2996 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0.0997 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ 1 & 0 & 0 & -0.1333 \\ 0 & 0 & -1 & 0.2996 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_6 = {}^0T_5\,{}^5T_6 = \begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ 1 & 0 & 0 & -0.1333 \\ 0 & 0 & -1 & 0.2996 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0.0996 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} 0 & -1 & 0 & -0.5885 \\ -1 & 0 & 0 & -0.1333 \\ 0 & 0 & -1 & 0.2000 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

Since these matrices takes the form of,

$$^0T_n(q_1, q_2, ..., q_n) =^0 T_n(\mathbf{q}) = \begin{bmatrix} ^0R_n(\mathbf{q}) & ^0o_n(\mathbf{q}) \\ 0 & 1 \end{bmatrix}$$

And the i^{th} column of the Jacobian matrix for an articulated robot takes the form of,

$$J_i = \begin{bmatrix} ^0\mathbf{z}_{i-1} \times (^0\mathbf{o}_n - ^0\mathbf{o}_{i-1}) \\ ^0\mathbf{z}_{i-1} \end{bmatrix}$$

The resulting Jacobian matrix will be:

$$\begin{bmatrix} 0.1333 & -0.0375 & 0.3458 & 0.0996 & 0 & 0 \\ -0.5885 & 0 & 0 & 0 & -0.0996 & 0 \\ 0 & -0.5885 & -0.4050 & -0.0997 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & -1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

4 Part D: Robot Singularities

4.1 Determine the DH matrix

	theta (rad)	**a (m)**	**d (m)**	**alpha (rad)**
Joint 1	q_1	1	0	0
Joint 2	q_2	1	0	0
Joint 3	q_3	1	0	0

Table 4: The DH table for the 3-Link Robot

4.2 Calculate the Jacobian

To calculate the Jacobian that relates the joint velocities to linear velocities of the manipulator, we need to first derive the homogenous transfromation matrix ${}^0T_3$ where,

$$ {}^0T_3 = {}^0T_1 \, {}^1T_2 \, {}^2T_3 $$

and,

$$ {}^{n-1}T_n = \begin{bmatrix} \cos(q_n) & -\sin(q_n)\cos(\alpha_n) & \sin(q_n)\sin(\alpha_n) & a_n\cos(q_n) \\ \sin(q_n) & \cos(q_n)\cos(\alpha_n) & -\cos(q_n)\sin(\alpha_n) & a_n\sin(q_n) \\ 0 & \sin(\alpha_n) & \cos(\alpha_n) & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

By substituting the DH parameters in table 4 into ${}^{n-1}T_n$ we get,

$$ {}^{n-1}T_n = \begin{bmatrix} \cos(q_n) & -\sin(q_n) & 0 & \cos(q_n) \\ \sin(q_n) & \cos(q_n) & 0 & \sin(q_n) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

which can be simplified to a 2D transformation matrix by removing the third row and column,

$$ {}^{n-1}T_n = \begin{bmatrix} \cos(q_n) & -\sin(q_n) & \cos(q_n) \\ \sin(q_n) & \cos(q_n) & \sin(q_n) \\ 0 & 0 & 1 \end{bmatrix} $$

We can now derive the full 2D homogenous transformation matrix ${}^0T_3$ by chaining transformation matrices.

$$ \begin{aligned} {}^0T_3 &= {}^0T_1 \, {}^1T_2 \, {}^2T_3 \\ &= \begin{bmatrix} \cos(q_1) & -\sin(q_1) & \cos(q_1) \\ \sin(q_1) & \cos(q_1) & \sin(q_1) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(q_2) & -\sin(q_2) & \cos(q_2) \\ \sin(q_2) & \cos(q_2) & \sin(q_2) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(q_3) & -\sin(q_3) & \cos(q_3) \\ \sin(q_3) & \cos(q_3) & \sin(q_3) \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos(q_1+q_2+q_3) & -\sin(q_1+q_2+q_3) & \cos(q_1+q_2+q_3)+\cos(q_1+q_2)+\cos(q_1) \\ \sin(q_1+q_2+q_3) & \cos(q_1+q_2+q_3) & \sin(q_1+q_2+q_3)+\sin(q_1+q_2)+\sin(q_1) \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} $$

And since the forward kinematics solution for a 2D transformation matrix takes the form of,

$$\left[\begin{array}{cc|c} & R & T \\ \hline 0 & 0 & 1 \end{array}\right]$$

We can derive that,

$$\mathbf{T} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(q_1 + q_2 + q_3) + \cos(q_1 + q_2) + \cos(q_1) \\ \sin(q_1 + q_2 + q_3) + \sin(q_1 + q_2) + \sin(q_1) \end{bmatrix}$$

then,

$$\begin{aligned} \frac{\delta x}{\delta q_1} &= -\sin(q_1 + q_2 + q_3) - \sin(q_1 + q_2) - \sin(q_1) \\ \frac{\delta x}{\delta q_2} &= -\sin(q_1 + q_2 + q_3) - \sin(q_1 + q_2) \\ \frac{\delta x}{\delta q_3} &= -\sin(q_1 + q_2 + q_3) \\ \frac{\delta y}{\delta q_1} &= \cos(q_1 + q_2 + q_3) + \cos(q_1 + q_2) + \cos(q_1) \\ \frac{\delta y}{\delta q_2} &= \cos(q_1 + q_2 + q_3) + \cos(q_1 + q_2) \\ \frac{\delta y}{\delta q_3} &= \cos(q_1 + q_2 + q_3) \end{aligned}$$

So the Jacobian matrix ($\mathbf{J}_v$) that relates joint velocity $\begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix}$ to spacial velocities $\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$ takes the form of,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \mathbf{J}_v \begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix}$$

where,

$$\mathbf{J}_v = \begin{bmatrix} \frac{\delta x}{\delta q_1} & \frac{\delta x}{\delta q_2} & \frac{\delta x}{\delta q_3} \\ \frac{\delta y}{\delta q_1} & \frac{\delta y}{\delta q_2} & \frac{\delta y}{\delta q_3} \end{bmatrix}$$

$$= \begin{bmatrix} -\sin(q_1 + q_2 + q_3) - \sin(q_1 + q_2) - \sin(q_1) & -\sin(q_1 + q_2 + q_3) - \sin(q_1 + q_2) & -\sin(q_1 + q_2 + q_3) \\ \cos(q_1 + q_2 + q_3) + \cos(q_1 + q_2) + \cos(q_1) & \cos(q_1 + q_2 + q_3) + \cos(q_1 + q_2) & \cos(q_1 + q_2 + q_3) \end{bmatrix}$$

The Jacobian that relates the joint velocities with angular velocities can be derived as follows,

$$\mathbf{R} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} = \begin{bmatrix} \cos(q_1 + q_2 + q_3) & -\sin(q_1 + q_2 + q_3) \\ \sin(q_1 + q_2 + q_3) & \cos(q_1 + q_2 + q_3) \end{bmatrix}$$

where, by inspection,

$$\begin{aligned} & \theta = q_1 + q_2 + q_3 \\ \therefore \quad & \omega = \dot{\theta} = \dot{q_1} + \dot{q_2} + \dot{q_3} \\ \therefore \quad & \omega = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix} \end{aligned}$$

and hence,

$$\mathbf{J}_\omega = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

Therefore the full Jacobian matrix is as follows,

$$\mathbf{J} = \begin{bmatrix} -\sin(q_1+q_2+q_3) - \sin(q_1+q_2) - \sin(q_1) & -\sin(q_1+q_2+q_3) - \sin(q_1+q_2) & -\sin(q_1+q_2+q_3) \\ \cos(q_1+q_2+q_3) + \cos(q_1+q_2) + \cos(q_1) & \cos(q_1+q_2+q_3) + \cos(q_1+q_2) & \cos(q_1+q_2+q_3) \\ 1 & 1 & 1 \end{bmatrix}$$

4.3 For what value(s) is the manipulator at a singularity?

To calculate the values of the manipulator at a singularity, we check the values for which the Jacobian determinant equals 0.

$$\det(J) = 0$$

To simplify the calculation process, let,

$$\begin{aligned} a &= \sin(q_1+q_2+q_3) \\ b &= \sin(q_1+q_2) \\ c &= \sin(q_1) \\ d &= \cos(q_1+q_2+q_3) \\ e &= \cos(q_1+q_2) \\ f &= \cos(q_1) \end{aligned}$$

So that our Jacobian matrix is,

$$\mathbf{J} = \begin{bmatrix} -a-b-c & -a-b & -a \\ d+e+f & d+e & d \\ 1 & 1 & 1 \end{bmatrix}$$

And as such, solving for singularity positions through the Jacobian determinant $\det(\mathbf{J}) = 0$ equates to,

$$\begin{aligned} \det(\mathbf{J}) = 0 &= (-a-b-c)[(d+e)-(d)] - (-a-b)[(d+e+f)-(d)] + (-a)[(d+e+f)-(d+e)] \\ &= -e(a+b+c) + (a+b)(e+f) - af \\ &= -ae - be - ce + ae + af + be + bf - af \\ &= bf - ce \\ &= \sin(q_1+q_2)\cos(q_1) - \sin(q_1)\cos(q_1+q_2) \\ &= \sin(q_2) \qquad \because \sin(\alpha-\beta) = \sin(\alpha)\cos(\beta) - \sin(\beta)\cos(\alpha) \end{aligned}$$

Therefore the singularities exist whenever $\sin(q_2) = 0$ or in other words,

$$q_2 = n\pi, \quad n \in \mathbb{Z}$$

This occurs whenever link 2 is colinear with link 1.

4.4 What motion is restricted at this singularity?

4.5 What type of singularity is experienced?

5 Part E: SCARA Robot Inverse kinematics

	theta (rad)	**a (m)**	**d (m)**	**alpha (rad)**
Joint 1	θ_1	L_1	d_1	0
Joint 2	θ_2	L_2	0	0
Joint 3	0	0	d_3	0
Joint 4	θ_4	0	d_4	0

Table 5: The DH table for the UR5e robot arm at home joint configuration