Project 1

ROBOT KINEMATICS

Darshan Komala Sreeramu

z5610741

University of New South Wales

MTRN4230: Robotics
Term 2, 2025

TABLE OF CONTENTS

1. Part A: Dynamic forward kinematics (With a twist)

Demonstration done in Week 8 lab session.

2. Part B: UR5e modelling

For convention, we will look at links from 0 to n and the joints from 1 to n. We look at corresponding joint frames from 0 to n, with the 0^{th} frame being of the base and n^{th} being of the end-effector TCP(not the last joint).

Home Joint Configuration: [0.00, −75.00, −105.00, 90.00, 0.00]

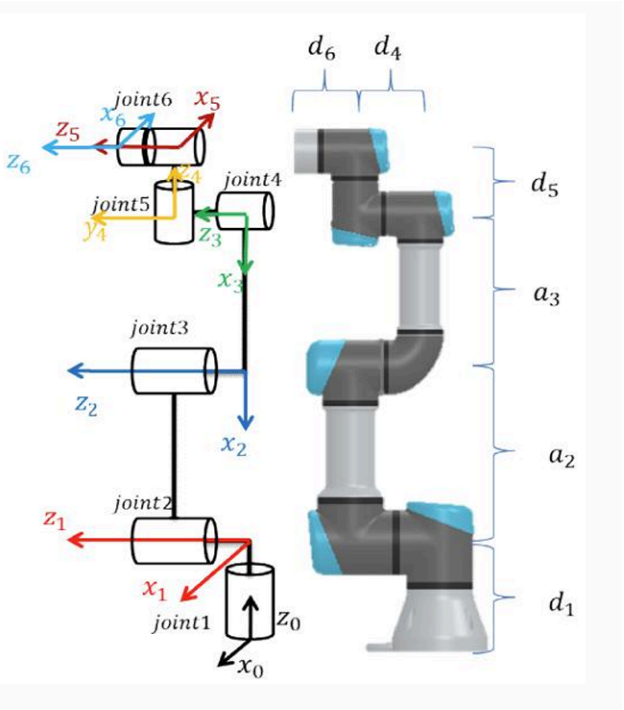


Figure 2.1 Fully functional UR5e

2.1. Modelling a new arm

Taking the broken elbow joint into consideration, we model a new arm with only five joints. To do this, we assume that the elbow doesn't exist anymore. We replace the elbow joint(joint 3) and the links it connects with a single new link that connects joints 2 and 4. The joints 2 and 4 are still in the same configuration, and the new link becomes a hypotenuse to the 2 collapse links that form a triangle with the elbow at 90°. We will build a new DH-table with this new structure in mind. The new arm will have only 5 degrees of freedom.

Figure 2.2 Configuration of the arm with new link

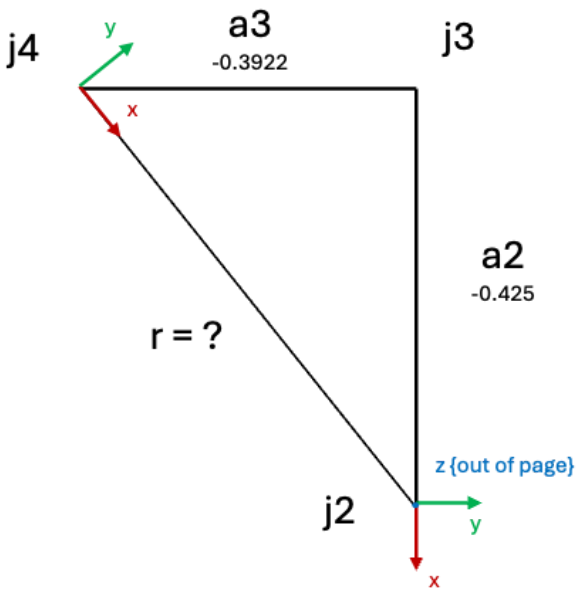


Figure 2.3 Geometry to collapse the elbow and its two links

Looking at Figure 2.2 for the new link structure and following the conventions for axes used in Figure 2.2 and Figure 2.3, we will build a new DH table of five joints.

2.2. Building a new DH table

Below is the new DH table for the 5-joint arm. The rotations and translations in the transformations are detailed in the next section.

Joint *i*	θ_i **(deg)**	$\mathbf{d_i}$ **(m)**	$\mathbf{a_i}$ **(m)**	$\boldsymbol{\alpha_i}$ **(rad)**
1	q_1	0.1625	0	$\frac{\pi}{2}$
2	$42.702 + q_2$	0	–0.5783	0
3	$47.298 + q_3$	0.1333	0	$\frac{\pi}{2}$
4	q_4	0.0997	0	$-\frac{\pi}{2}$
5	q_5	0.0996	0	0

Table 2.1 DH table for the new 5-joint arm

2.3. Joint transformations for Home position

Now that we have a new DH table, we need to create all the joint-to-joint transformations again.

Each homogeneous frame transform is given by:

$$ {}^{i-1}T_i(q_i) = \begin{bmatrix} \cos q_i & -\sin q_i \cos \alpha_i & \sin q_i \sin \alpha_i & a_i \cos q_i \\ \sin q_i & \cos q_i \cos \alpha_i & -\cos q_i \sin \alpha_i & a_i \sin q_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

- For the first homogeneous transformation, from joint 1 to 2:
    - Actuation about base by q_1.
    - Translate along z_0 by $d_1 = 0.1625$ m.
    - The origins are already coinciding.
    - Tilt the new z-axis by $\alpha_1 = +\pi/2$ to get frame 1.

Hence, i = 1 ($\theta_1 = q_1$, $d_1 = 0.1625$, $a_1 = 0$, $\alpha_1 = \pi/2$)

$$ {}^{0}T_{1} = \begin{bmatrix} \cos q_1 & 0 & \sin q_1 & 0 \\ \sin q_1 & 0 & -\cos q_1 & 0 \\ 0 & 1 & 0 & 0.1625 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Substituting $q_1 = 0$.

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

- For the second homogeneous transformation, from joint 2 to 3:
    - Twist along z_1 by 42.702^0 to align x-axes and also actuation by q_2.
    - X-axis is already aligned to the new plane.
    - Translate along the new x-axis by $a_2 = -0.5783$ m.
    - The z-axes are already aligned at the origin.

Hence, i = 2 ($\theta_2 = 42.702 + q_2$, $d_2 = 0$, $a_2 = -0.5783$, $\alpha_2 = 0$)

$$ {}^{1}T_{2} = \begin{bmatrix} \cos(42.702 + q_2) & -\sin(42.702 + q_2) & 0 & -0.5783\cos(42.702 + q_2) \\ \sin(42.702 + q_2) & \cos(42.702 + q_2) & 0 & -0.5783\sin(42.702 + q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Substituting $q_2 = -75$.

$$ {}^{1}T_{2} = \begin{bmatrix} 0.8453 & 0.5343 & 0 & -0.4888 \\ -0.5343 & 0.8453 & 0 & 0.3090 \\ 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} $$

- For the third homogeneous transformation, from joint 3 to 4:
    - Twist along z_2 by 47.298^0 to align x-axes and also actuation by q_3.
    - Translate along z_2 by $d_3 = 0.1333$ m.
    - The origins are already coinciding.
    - Tilt the new z-axis by $\alpha_3 = +\pi/2$ to get frame 3.

Hence, i = 3 (θ_3 = 47.298 + q_3, d_3 = 0.1333, a_3 = 0, α_3 = π/2)

$$ {}^{2}T_{3} = \begin{bmatrix} \cos(47.298 + q_3) & 0 & \sin(47.298 + q_3) & 0 \\ \sin(47.298 + q_3) & 0 & -\cos(47.298 + q_3) & 0 \\ 0 & 1 & 0 & 0.1333 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Substituting q_3 = -105.

$$ {}^{2}T_{3} = \begin{bmatrix} 0.5343 & 0.0000 & -0.8453 & 0 \\ -0.8453 & 0.0000 & -0.5343 & 0 \\ 0 & 1.0000 & 0.0000 & 0.1333 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} $$

- For the fourth homogeneous transformation, from joint 4 to 5:
    - Actuation about z_3 by q_4.
    - Translate along z_3 by d_4 = 0.0997 m.
    - The origins are already coinciding.
    - Tilt the new z-axis by α_4 = -π/2 to get frame 4.

Hence, i = 4 (θ_4 = q_4, d_4 = 0.0997, a_4 = 0, α_4 = -π/2)

$$ {}^{3}T_{4} = \begin{bmatrix} \cos q_4 & 0 & -\sin q_4 & 0 \\ \sin q_4 & 0 & \cos q_4 & 0 \\ 0 & -1 & 0 & 0.0997 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Substituting q_4 = 90.

$$ {}^{3}T_{4} = \begin{bmatrix} 0.0000 & -0.0000 & -1.0000 & 0 \\ 1.0000 & 0.0000 & 0.0000 & 0 \\ 0 & -1.0000 & 0.0000 & 0.0997 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} $$

- For the fifth homogeneous transformation, from joint 5 to end-effector:
    - Actuation about z_4 by q_4.
    - Translate along z_4 by d_5 = 0.0996 m.
    - The axes are already at the same center and the z-axes coincide.

Hence, i = 5 ($\theta_5 = q_5$, $d_5 = 0.0996$, $a_5 = 0$, $\alpha_5 = 0$)

$$^4T_5 = \begin{bmatrix} \cos q_5 & -\sin q_5 & 0 & 0 \\ \sin q_5 & \cos q_5 & 0 & 0 \\ 0 & 0 & 1 & 0.0996 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Substituting $q_5 = 0$.

$$^4T_5 = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.0996 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix}$$

Then, we find the cumulative transformation matrices between the base to every joint. We can chain the transformations to get the end-effector matrix:

$$^0T_n = {}^0T_1\,{}^1T_2 \cdots {}^{n-1}T_n$$

For the transformation from base to joint 1, we have already calculated $^0T_1$.

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

Moving on,

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

$$^0T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0.1625 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.8453 & 0.5343 & 0 & -0.4888 \\ -0.5343 & 0.8453 & 0 & 0.3090 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.8453 & 0.5343 & 0 & -0.4888 \\ 0 & 0 & -1 & 0 \\ -0.5343 & 0.8453 & 0 & 0.4715 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

$$^0T_3 = \begin{bmatrix} 0.8453 & 0.5343 & 0 & -0.4888 \\ 0 & 0 & -10 & 0 \\ -0.5343 & 0.8453 & 0 & 0.4715 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5343 & 0 & -0.8453 & 0 \\ -0.8453 & 0 & -0.5343 & 0 \\ 0 & 1 & 0 & 0.1333 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 & -0.4888 \\ 0 & -1 & 0 & -0.1333 \\ -1 & 0 & 0 & 0.4715 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^0T_4 = {}^0T_3\,{}^3T_4$$

$$^0T_4 = \begin{bmatrix} 0 & 0 & -1 & -0.4888 \\ 0 & -1 & 0 & -0.1333 \\ -1 & 0 & 0 & 0.4715 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 & 0 \\ 1 & 0 & 00 & 0 \\ 0 & -1 & 0 & 0.0997 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ -1 & 0 & 0 & -0.1333 \\ 0 & 0 & 1 & 0.4715 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^0T_5 = {}^0T_4\,{}^4T_5$$

$$^0T_5 = \begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ -1 & 0 & 0 & -0.1333 \\ 0 & 0 & 1 & 0.4715 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 1.0000 & 0.0996 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ -1 & 0 & 0 & -0.1333 \\ 0 & 0 & 1 & 0.5711 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This is the final transformation matrix obtained from base to end-effector:

$$^0T_5 = \begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ -1 & 0 & 0 & -0.1333 \\ 0 & 0 & 1 & 0.5711 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.4. Extraction of Position and Orientation from the Transformation Matrix

Given,

$$^0T_5 = \begin{bmatrix} 0 & 1 & 0 & -0.5885 \\ -1 & 0 & 0 & -0.1333 \\ 0 & 0 & 1 & 0.5711 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From the definition of homogeneous transformation matrix, we extract pose using r_{14}, r_{24} and r_{34}.

P = [x, y, z] = [r_{14}, r_{24}, r_{34}] = [-0.5885, -0.1333, 0.5711]

Also using the rotation sub-matrix,

$$R = \begin{bmatrix} r11 & r12 & r13 \\ r21 & r22 & r23 \\ r31 & r32 & r33 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We use the ZYX convention,

Roll, r = atan2(r_{32} , r_{33}) = atan2(0, 1) = 0 rad.

Pitch, p = atan2($-r_{31}$, $\sqrt{r_{32}^2 + r_{33}^2}$) = atan2(0, $\sqrt{0^2 + 1^2}$) = 0 rad.

Yaw, y = atan2(r_{21}, r_{11}) = atan2(-1, 0) = $\frac{-\pi}{2}$ = -0.5 rad.

But, the Universal Robots controller uses the range [0, 2π] to represent all angles, while mathematically, we typically use [–π, +π] for atan2(). Since 2π is literally a 360^0 rotation, they are geometrically and logically the same to the arm controller. So, we'll transform the obtained yaw value into the range used by the arm controller.

Yaw, y = 2π - atan2(-1, 0) = $\frac{3\pi}{2}$ (270^o) = 4.7124 rad.

Final end-effector pose:
[x, y, z, r, p, y] = [-0.5885, -0.1333, 0.5711, 0, 0, 4.7124].

2.5. Model of Broken UR5e using RVC toolbox

The new DH table was used to construct a new arm model using the RVC toolbox in MATLAB. We then perform the forward kinematic conversion(using the fkine function) to get the pose with the angles in RPY.

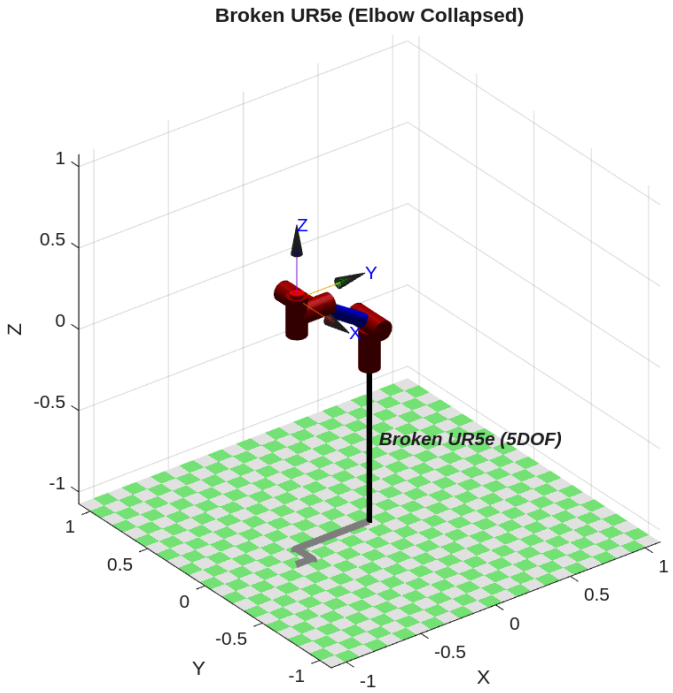


Figure 2.4 Robot constructed with the new DH table using RVC

Above is the constructed robot in RVC and below is the final transformation matrix obtained using the fkine function, and also the extracted position and orientation values(in RPY).

```
Command Window
>> Project1PartB
Final transformation matrix 0T5:
    6.123e-17          1-2.833e-16    -0.5885
           -1 6.123e-17 8.577e-33    -0.1333
    2.592e-32 2.833e-16          1     0.5711
            0          0          0          1
Position(m) = [-0.5885  -0.1333  0.5711]
RPY(rad)    = [0.000  -0.000  4.712]
>>
```

Figure 2.5 Final Transformation Matrix and extracted pose values

A lot of the elements in the matrix are of order smaller than 10^{-16}, which make no significant numerical impact to the solution, and hence holds with the manual calculations done in Section 2.4.

Below is a screenshot of the workspace for the same. For the complete code, refer to Appendix A.

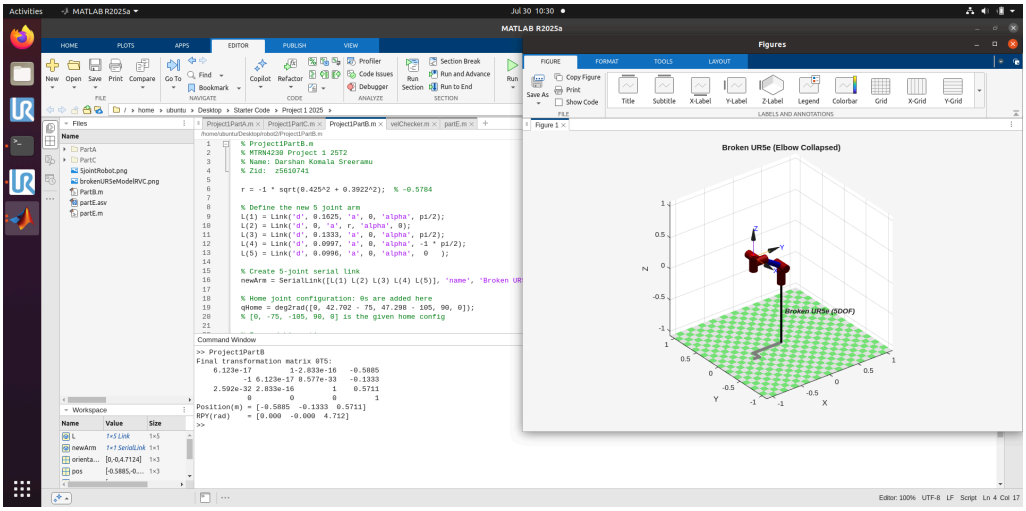


Figure 2.6 Part B MATLAB workspace

2.6. Validating Results

To validate our results from manual calculations as well as the RVC outputs, we set the UR5e to the home joint configuration in the simulator.

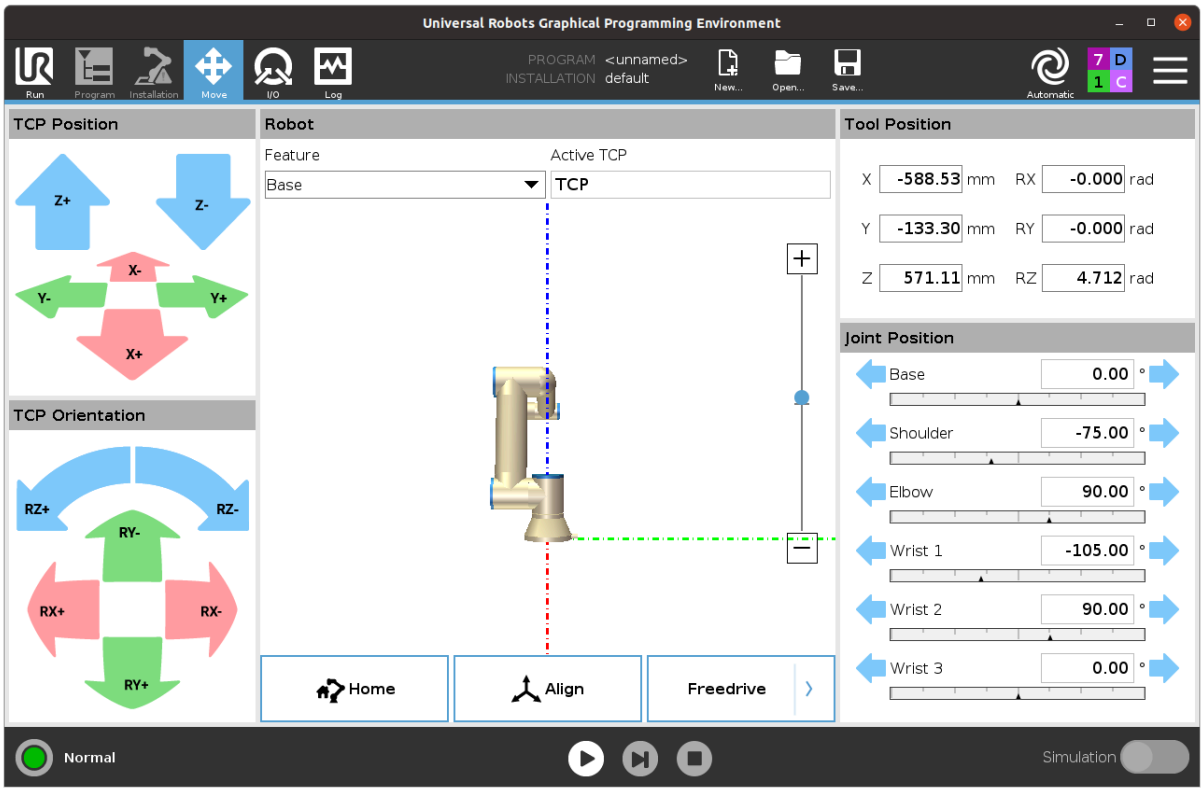


Figure 2.7 Simulator showing pose for the home joint configuration

We just set the elbow to 90^0. From the above figure, it's evident that all the calculations match and hold up with the simulator.

This leads us to conclude that the new 5-joint arm was modelled properly for the given constraints of the UR5e. But since it still has only 5 degrees of freedom, it can only hold up to the UR5e for only certain configurations, of which the home joint configuration was one of them.

3. Part C: Robot Joint Speeds

3.1. Execution in MATLAB

Datasets for end-effector velocities and joint positions with the suffix ‘1’ have been used for the program execution.

Below is the function ‘calculateMaxJointVelocity’:

buntu/Desktop/Starter Code/Project 1 2025/PartC/Project1PartC.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% You must implement the following function
% Return the max joint angular velocity value
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: Darshan Komala Sreeramu
% Zid: z5610741
function maxJointVel = calculateMaxJointVelocity(joint_poses, TCP_vels)
    % Converting to N×6 from 6xN
    if size(TCP_vels, 1) == 6 && size(TCP_vels, 2) == size(joint_poses, 1) % Also verify if datasets have equal entries
        TCP_vels = TCP_vels';
    end

    % Define the unbroken arm
    Links(1) = Link('d', 0.1625, 'a', 0, 'alpha', pi/2);
    Links(2) = Link('d', 0, 'a', -0.425, 'alpha', 0);
    Links(3) = Link('d', 0, 'a', -0.3922, 'alpha', 0);
    Links(4) = Link('d', 0.1333, 'a', 0, 'alpha', pi/2);
    Links(5) = Link('d', 0.0997, 'a', 0, 'alpha', -pi/2);
    Links(6) = Link('d', 0.0996, 'a', 0, 'alpha', 0);
    arm = SerialLink(Links, 'name', 'UR5e_unbroken');

    numOfEntries = size(joint_poses, 1);
    jointVelo = zeros(numOfEntries, 6); % Make empty initial array

    % Loop through all data to find joint velocities
    for i = 1:numOfEntries
        q = joint_poses(i, :);  % (Dim: 1×6)
        Vtcp = TCP_vels(i, :)';  % already in base frame(Dim: 6x1 after transpose)

        % Base-frame Jacobian
        J = arm.jacob0(q);  % (Dim: 6×6)

        % joint velocity using the inverse(pinv instead of inv in case matrix isn't square)
        dq = pinv(J) * Vtcp;  % (Dim: 6×1)

        jointVelo(i, :) = dq';
    end

    % Search and store index of max mod value
    maxVal = max(abs(jointVelo), [], 'all');
    maxIndex = find(abs(jointVelo) == maxVal, 1, 'first');
    [sampleIndex, jointIndex] = ind2sub(size(jointVelo), maxIndex);

    % Extract the signed value at that point
    peakVel = jointVelo(maxIndex);

    % Print out max vel, which joint, and what index in dataset
    fprintf("Max joint velocity = %+0.4f rad/s on joint %d at sample %d.\n", peakVel, jointIndex, sampleIndex);

    maxJointVel = peakVel;
end
```

Figure 3.1 calculateMaxJointVelocity function

Upon executing the above after building on the provided code stub, we get:

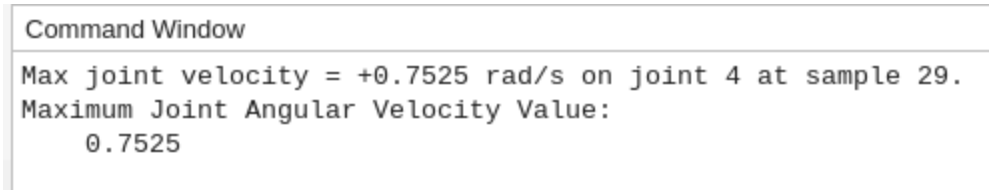


Figure 3.2 Output for dataset with suffix 1

From the output Command Window, we can clearly see that the maximum(absolute) joint velocity achieved during the movej sequence for the used dataset is 0.7525 rad/s, found on Joint 4.

We simply took the absolute values of all entries in the joint-velocity matrix, used the max() function to get the linear index of the largest magnitude, and then mapped that index to its column number, which is in turn, the joint number.

Below is a screenshot of the MATLAB window after execution:

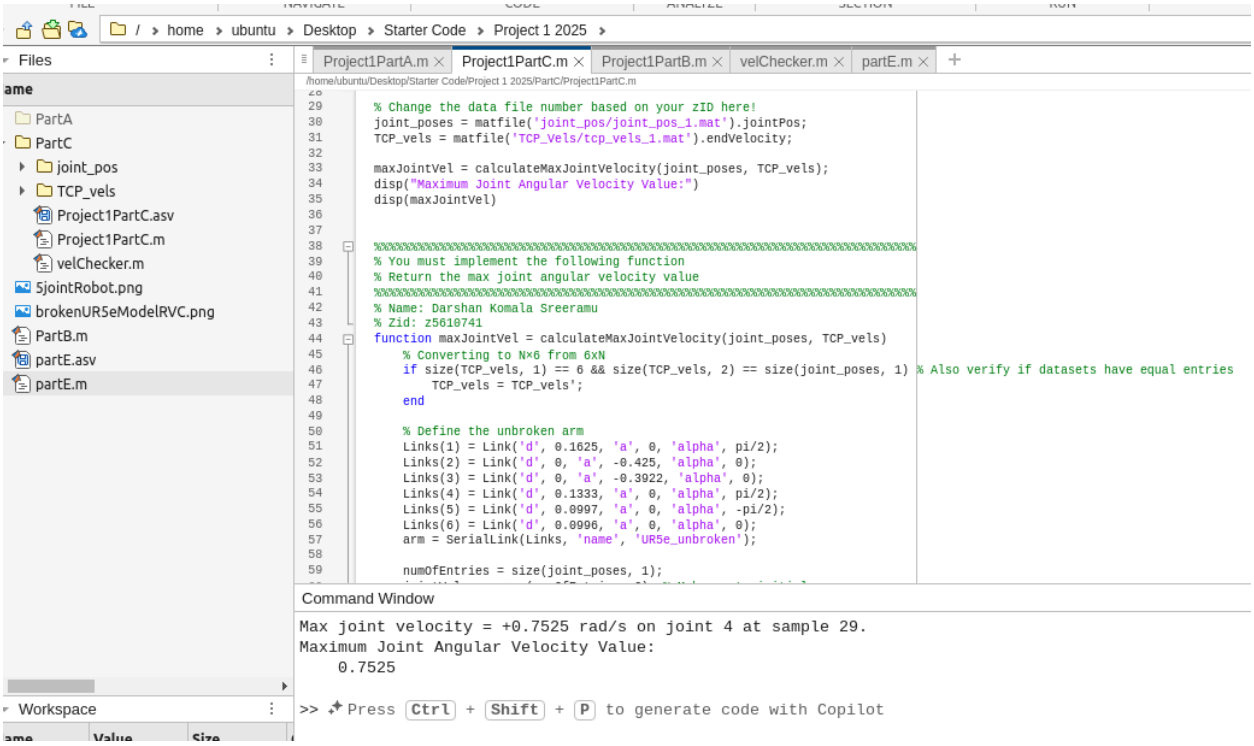


Figure 3.3 Screenshot of the MATLAB workspace

3.2. Approach of the Solution

To calculate the maximum joint velocity achieved from the given files, the core idea is to use the Jacobian to relate joint velocities to end-effector velocities.

$$\begin{bmatrix} \vec{v} \\ \vec{\omega} \end{bmatrix} = J(q) \cdot \dot{q}$$

Where,

- $q \in \mathbb{R}^6$ are the joint angles
- $\dot{q} \in \mathbb{R}^6$ are the joint angular velocities we wish to find
- $V \in \mathbb{R}^6$ is the measured end-effector twist $[v_x, v_y, v_z, \omega_x, \omega_y, \omega_z]^\top$
- $J(q) \in \mathbb{R}^{6\times6}$ is the **geometric Jacobian**, a function of q

To build the Jacobian, we first define the DH parameters for the robot. For each joint, we form the homogeneous transform matrix.

$$^{i-1}T_i(q_i) = \begin{bmatrix} \cos q_i & -\sin q_i \cos \alpha_i & \sin q_i \sin \alpha_i & a_i \cos q_i \\ \sin q_i & \cos q_i \cos \alpha_i & -\cos q_i \sin \alpha_i & a_i \sin q_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We then accumulate ^0T_j for j = 1 to 6. And, at each step, we extract:

➔ z_j = the joint's rotation axis (third column of the rotation submatrix)
➔ o_j = the joint's origin (translation column)

Then, for each joint j, the Jacobian column is

$$J_j = \begin{bmatrix} \mathbf{z}_j \times (\mathbf{o}_6 - \mathbf{o}_{j-1}) \\ \mathbf{z}_j \end{bmatrix}$$

where o_6 is the end-effector position in the base frame. Stacking all six columns gives us the J(q).

In this problem, we are given the end-effector velocities. So, we find the joint velocities after finding the inverse Jacobian.

$$\dot{q} = J(q)^{-1} \cdot \begin{bmatrix} \vec{v} \\ \vec{\omega} \end{bmatrix}$$

We do this for all entries and store the resultants. Then, we search for the peak absolute value of joint velocities by reading through the resultant joint velocities.

3.3. Calculating the homogeneous transformation

We start with the frame by frame DH transforms after defining the DH table for the arm.

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

For an 6-DOF serial manipulator, the homogeneous transform from base to end-effector is given by:

$$^{0}T_{6} \;=\; {}^{0}T_{1}\,{}^{1}T_{2}\cdots{}^{5}T_{6}$$

Where,

$$^{i-1}T_{i}(q_i) = \begin{bmatrix} \cos q_i & -\sin q_i \cos\alpha_i & \sin q_i \sin\alpha_i & a_i \cos q_i \\ \sin q_i & \cos q_i \cos\alpha_i & -\cos q_i \sin\alpha_i & a_i \sin q_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Using this, we recursively calculate and end up with the homogeneous transformation from base to end-effector.

Initial joint position: [0.0020, -1.3125, 1.5758, -1.8479, -1.5657, 3.1668].

We plug in these values as the DH parameter theta.

Starting off from the base, we substitute DH parameters to find $^{0}T_{1}$:

$$^{0}T_{1} = \begin{bmatrix} 0.999998 & 0 & 0.002000 & 0 \\ 0.002000 & 0 & -1.000000 & 0 \\ 0 & 1.000000 & 0 & 0.1625 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Do the same for $^{1}T_{2}$:

$$ {}^{1}T_{2} = \begin{bmatrix} 0.2554 & 0.9668 & 0 & -0.1086 \\ -0.9668 & 0.2554 & 0 & 0.4109 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Same for ${}^{2}T_{3}$:

$$ {}^{2}T_{3} = \begin{bmatrix} -0.0050 & -1.0000 & 0 & 0.0020 \\ 1.0000 & -0.0050 & 0 & -0.3922 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Same for ${}^{3}T_{4}$:

$$ {}^{3}T_{4} = \begin{bmatrix} -0.2742 & 0 & -0.9617 & 0 \\ -0.9617 & 0 & 0.2742 & 0 \\ 0 & 1.0000 & 0 & 0.1333 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Same for ${}^{4}T_{5}$:

$$ {}^{4}T_{5} = \begin{bmatrix} 0.0051 & 0 & 0.99999 & 0 \\ -0.99999 & 0 & 0.0051 & 0 \\ 0 & -1.0000 & 0 & 0.0997 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Same for ${}^{5}T_{6}$:

$$ {}^{5}T_{6} = \begin{bmatrix} -0.99975 & 0.02205 & 0 & 0 \\ -0.02205 & -0.99975 & 0 & 0 \\ 0 & 0 & 1 & 0.0996 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Then we get started with chaining the matrices to get to the end-effector.

$$ {}^{0}T_{6} \;=\; {}^{0}T_{1}\,{}^{1}T_{2} \cdots {}^{5}T_{6} $$

We already have,

$$
{}^0T_1 = \begin{bmatrix} 0.999998 & 0 & 0.002000 & 0 \\ 0.002000 & 0 & -1.000000 & 0 \\ 0 & 1.000000 & 0 & 0.1625 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

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

$$
{}^0T_2 = \begin{bmatrix} 0.999998 & 0 & 0.002000 & 0 \\ 0.0020 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0.1625 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.2554 & 0.9668 & 0 & -0.1086 \\ -0.9668 & 0.2554 & 0 & 0.4109 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.2554 & 0.9668 & 0.0020 & -0.1086 \\ 0.0005 & 0.0019 & -1.0000 & -0.0002 \\ -0.9668 & 0.2554 & 0 & 0.5734 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

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

$$
{}^0T_3 = \begin{bmatrix} 0.2554 & 0.9668 & 0.0020 & -0.1086 \\ 0.0005 & 0.0019 & -1 & -0.0002 \\ -0.9668 & 0.2554 & 0 & 0.5734 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.0050 & -1 & 0 & 0.0020 \\ 1 & -0.0050 & 0 & -0.3922 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.9655 & -0.2603 & 0.0020 & -0.4872 \\ 0.0019 & -0.0005 & -1.0000 & -0.0010 \\ 0.2603 & 0.9655 & 0 & 0.4713 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_4 = {}^0T_3 \, {}^3T_4
$$

$$
{}^0T_4 = \begin{bmatrix} 0.9655 & -0.2603 & 0.0020 & -0.4872 \\ 0.0019 & -0.0005 & -1.0000 & -0.0010 \\ 0.2603 & 0.9655 & 0 & 0.4713 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.2742 & 0 & -0.9617 & 0 \\ -0.9617 & 0 & 0.2742 & 0 \\ 0 & 1.0000 & 0 & 0.1333 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -0.0138 & 0.0020 & -0.9999 & -0.4870 \\ -0.0000 & -1.0000 & -0.0020 & -0.1343 \\ -0.9999 & 0.0000 & 0.0138 & 0.4713 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_5 = {}^0T_4 \, {}^4T_5
$$

$$
{}^0T_5 = \begin{bmatrix} -0.0138 & 0.0020 & -0.9999 & -0.4870 \\ -0.0000 & -1.0000 & -0.0020 & -0.1343 \\ -0.9999 & 0.0000 & 0.0138 & 0.4713 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.0051 & 0 & 0.99999 & 0 \\ -0.99999 & 0 & 0.0051 & 0 \\ 0 & -1.0000 & 0 & 0.0997 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -0.0021 & 0.9999 & -0.0138 & -0.5867 \\ 1.0000 & 0.0020 & -0.0051 & -0.1345 \\ -0.0051 & -0.0138 & -0.9999 & 0.4727 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^0T_6 = {}^0T_5 \, {}^5T_6
$$

$$
{}^0T_6 = \begin{bmatrix} -0.0021 & 0.9999 & -0.0138 & -0.5867 \\ 1.0000 & 0.0020 & -0.0051 & -0.1345 \\ -0.0051 & -0.0138 & -0.9999 & 0.4727 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.99975 & 0.02205 & 0 & 0 \\ -0.02205 & -0.99975 & 0 & 0 \\ 0 & 0 & 1 & 0.0996 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -0.0231 & -0.9996 & -0.0138 & -0.5880 \\ -0.9997 & 0.0232 & -0.0051 & -0.1350 \\ 0.0054 & 0.0137 & -0.9999 & 0.3731 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

The final homogeneous transformation matrix from base to end-effector:

$$
{}^0T_6 = \begin{bmatrix} -0.0231 & -0.9996 & -0.0138 & -0.5880 \\ -0.9997 & 0.0232 & -0.0051 & -0.1350 \\ 0.0054 & 0.0137 & -0.9999 & 0.3731 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

Since we have all the transformation matrices, we can now proceed with calculating the Jacobian.

3.4. Calculating the Jacobian

Extracting intermediate o and z:

i	o_i (m)	z_i
0	$\begin{bmatrix}0\\0\\0\end{bmatrix}$	base–axis: $\begin{bmatrix}0\\0\\1\end{bmatrix}$
1	$\begin{bmatrix}0\\0\\0.1625\end{bmatrix}$	$\begin{bmatrix}0.002\\-1\\0\end{bmatrix}$
2	$\begin{bmatrix}-0.1086\\-0.0002\\0.5734\end{bmatrix}$	$\begin{bmatrix}0.002\\-1\\0\end{bmatrix}$
3	$\begin{bmatrix}-0.4872\\-0.001\\0.4713\end{bmatrix}$	$\begin{bmatrix}0.002\\-1\\0\end{bmatrix}$
4	$\begin{bmatrix}-0.4870\\-0.1343\\0.4713\end{bmatrix}$	$\begin{bmatrix}-0.9999\\-0.002\\0.0138\end{bmatrix}$
5	$\begin{bmatrix}-0.5867\\-0.1345\\0.4727\end{bmatrix}$	$\begin{bmatrix}0.9999\\0.0051\\-0.0138\end{bmatrix}$
6	$\begin{bmatrix}-0.5880\\-0.1350\\0.3731\end{bmatrix}$	$\begin{bmatrix}-0.0138\\-0.0051\\-0.9999\end{bmatrix}$

For a revolute joint j, the j[th] column is:

$$J_j = \begin{bmatrix} J_{1:3,j} \\ J_{4:6,j} \end{bmatrix} = \begin{bmatrix} z_j \times (o_6 - o_{j-1}) \\ z_j \end{bmatrix}$$

We now plug in o_i and z_i for each joint and column.

- **Column 1(j = 1):**

$$o_6 - o_0 = \begin{bmatrix} -0.8172 \\ -0.2329 \\ 0.0628 \end{bmatrix}, \quad z_1 = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$$

$$z_1 \times (o_6 - o_0) = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 0 & -1 & 0 \\ -0.8172 & -0.2329 & 0.0628 \end{vmatrix} = \begin{bmatrix} 0.1349 \\ -0.5880 \\ 0 \end{bmatrix}$$

So,

$$J_1 = \begin{bmatrix} 0.1349 \\ -0.5880 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

- **Column 2(j = 2):**

$$o_6 - o_1 = \begin{bmatrix} -0.8172 \\ -0.2329 \\ 0.0628 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0.1625 \end{bmatrix} = \begin{bmatrix} -0.8172 \\ -0.2329 \\ -0.0997 \end{bmatrix}, \quad z_2 = \begin{bmatrix} 0.0020 \\ -1.0000 \\ 0 \end{bmatrix}$$

$$z_2 \times (o_6 - o_1) = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 0.0020 & -1.0000 & 0 \\ -0.8172 & -0.2329 & -0.0997 \end{vmatrix} \approx \begin{bmatrix} -0.2106 \\ -0.0004 \\ -0.5883 \end{bmatrix}$$

So,

$$J_2 = \begin{bmatrix} -0.2106 \\ -0.0004 \\ -0.5883 \\ 0.0019 \\ -0.9999 \\ 0 \end{bmatrix}$$

- **Column 3(j = 3):**

$$o_6 - o_2 = \begin{bmatrix} -0.8172 \\ -0.2329 \\ 0.0628 \end{bmatrix} - \begin{bmatrix} -0.1086 \\ -0.0002 \\ 0.5734 \end{bmatrix} = \begin{bmatrix} -0.7086 \\ -0.2327 \\ -0.5106 \end{bmatrix}, \quad z_3 = \begin{bmatrix} 0.0020 \\ -1.0000 \\ 0 \end{bmatrix}$$

$$z_3 \times (o_6 - o_2) = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 0.0020 & -1.0000 & 0 \\ -0.7086 & -0.2327 & -0.5106 \end{vmatrix} \approx \begin{bmatrix} 0.2003 \\ 0.0004 \\ -0.4797 \end{bmatrix}$$

So,

$$J_3 = \begin{bmatrix} 0.2003 \\ 0.0004 \\ -0.4797 \\ 0.0019 \\ -0.9999 \\ 0 \end{bmatrix}$$

- **Column 4(j = 4):**

$$o_6 - o_3 = \begin{bmatrix} -0.8172 \\ -0.2329 \\ 0.0628 \end{bmatrix} - \begin{bmatrix} -0.4872 \\ -0.0010 \\ 0.4713 \end{bmatrix} = \begin{bmatrix} -0.3300 \\ -0.2319 \\ -0.4085 \end{bmatrix}, \quad z_4 = \begin{bmatrix} -0.9999 \\ -0.0020 \\ 0.0138 \end{bmatrix}$$

$$z_4 \times (o_6 - o_3) \approx \begin{bmatrix} 0.0982 \\ 0.0002 \\ -0.1010 \end{bmatrix}$$

So,

$$J_4 = \begin{bmatrix} 0.0982 \\ 0.0002 \\ -0.1010 \\ 0.00199 \\ -0.9999 \\ 0.0 \end{bmatrix}$$

- **Column 5(j = 5):**

$$o_6 - o_4 = \begin{bmatrix} -0.8172 \\ -0.2329 \\ 0.0628 \end{bmatrix} - \begin{bmatrix} -0.4870 \\ -0.1343 \\ 0.4713 \end{bmatrix} = \begin{bmatrix} -0.3302 \\ -0.0986 \\ -0.4085 \end{bmatrix}, \quad z_5 = \begin{bmatrix} -0.0138 \\ -0.0051 \\ -0.9999 \end{bmatrix}$$

$$z_5 \times (o_6 - o_4) \approx \begin{bmatrix} 0.0002 \\ -0.0996 \\ 0.0005 \end{bmatrix}$$

So,

$$J_5 = \begin{bmatrix} 0.0002 \\ -0.0996 \\ 0.0005 \\ -0.9999 \\ -0.0019 \\ 0.01380 \end{bmatrix}$$

- **Column 6(j = 6):**

$$o_6 - o_5 = \begin{bmatrix} -0.8172 \\ -0.2329 \\ 0.0628 \end{bmatrix} - \begin{bmatrix} -0.5867 \\ -0.1345 \\ 0.4727 \end{bmatrix} = \begin{bmatrix} -0.2305 \\ -0.0984 \\ -0.4099 \end{bmatrix}, \quad z_6 = \begin{bmatrix} -0.0138 \\ -0.0051 \\ -0.9999 \end{bmatrix}$$

$$z_6 \times (o_6 - o_5) \approx \begin{bmatrix} (-0.0051)(-0.4099) - (-0.9999)(-0.0984) \\ (-0.9999)(-0.2305) - (-0.0138)(-0.4099) \\ (-0.0138)(-0.0984) - (-0.0051)(-0.2305) \end{bmatrix} \approx \begin{bmatrix} 0.0000 \\ 0.0000 \\ 0.0000 \end{bmatrix}$$

So,

$$J_6 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -0.0138 \\ -0.0051 \\ -0.9999 \end{bmatrix}$$

Putting these six columns side by side, we get the final Jacobian matrix.

$$J(q_0) = \begin{bmatrix} 0.1349 & -0.2106 & 0.2002 & 0.0982 & 0.0002 & 0 \\ -0.5880 & -0.0004 & 0.0004 & 0.0002 & -0.0996 & 0 \\ 0 & -0.5883 & -0.4797 & -0.1010 & 0.0005 & 0 \\ 0 & 0.0020 & 0.0020 & 0.0020 & -0.9999 & -0.0138 \\ 0 & -0.9999 & -0.9999 & -0.9999 & -0.0020 & -0.0051 \\ 1 & 0 & 0 & 0 & 0.0138 & -0.9999 \end{bmatrix}$$

This is the final Jacobian for the initial position given for the suffix 1.

4. Part D: Robot Singularities

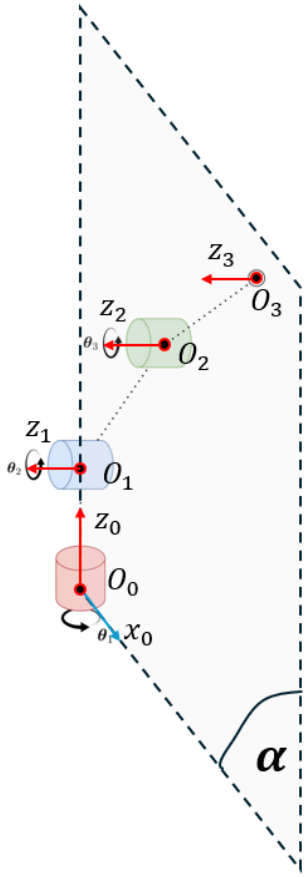


Figure 4.1 3D Robotic Arm to analyze

We look at the new robot configuration and to determine the Singularity positions, we make a new DH table, calculate the new Jacobian and apply the condition det(J) = 0 to find the positions.

4.1. The New DH Table

The figure is a classic planar 3-R manipulator. We now calculate the DH parameters from joint 1 to joint 3.

- The joint variable for each revolute joint is its rotation about z. These are the three independent DOF θ_1, θ_2, θ_3.
- Each joint axis lies in the same plane α, so there is no offset along the previous z-axis ($d_i = 0$).
- All links are unit-length bars, exactly 1 m.
- z_0 is vertical, pointing along the plane α. z_1, z_2, z_3 are all pointing perpendicular to the plane α.

Joint *i*	θ_i	**d_i (m)**	**a_i (m)**	**α_i (rad)**
1	θ_1	0	0	$\frac{\pi}{2}$
2	θ_2	0	1	0
3	θ_3	0	1	0

Table 4.1 New DH table for Part D

4.2. Calculating the new Transformation Matrices

Now that we have a new DH table, we need to create all the joint-to-joint transformations again. We number joints from 1 to 3 and their corresponding frames from 0 to 3(end-effector).

Each homogeneous frame transform is given by:

$$^{i-1}T_i(q_i) = \begin{bmatrix} \cos q_i & -\sin q_i \cos \alpha_i & \sin q_i \sin \alpha_i & a_i \cos q_i \\ \sin q_i & \cos q_i \cos \alpha_i & -\cos q_i \sin \alpha_i & a_i \sin q_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For joint 1 to 2:

$$^0T_1 = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 & 0 \\ \sin\theta_1 & 0 & -\cos\theta_1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then moving from joint 2 to 3:

$$^1T_2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & \cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & \sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Then moving from joint 3 to end-effector:

$$
{}^{2}T_{3}\begin{bmatrix}\cos\theta_3 & -\sin\theta_3 & 0 & \cos\theta_3\\ \sin\theta_3 & \cos\theta_3 & 0 & \sin\theta_3\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

Then, we find the cumulative transformation matrices between the base to every joint. We can chain the transformations to get the end-effector matrix:

$$
{}^{0}T_{3} \;=\; {}^{0}T_{1}\,{}^{1}T_{2}\,{}^{2}T_{3}
$$

$$
{}^{0}T_{3} = \begin{bmatrix}\cos\theta_1 & 0 & \sin\theta_1 & 0\\ \sin\theta_1 & 0 & -\cos\theta_1 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}\cos\theta_2 & -\sin\theta_2 & 0 & \cos\theta_2\\ \sin\theta_2 & \cos\theta_2 & 0 & \sin\theta_2\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}\cos\theta_3 & -\sin\theta_3 & 0 & \cos\theta_3\\ \sin\theta_3 & \cos\theta_3 & 0 & \sin\theta_3\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

$$
{}^{0}T_{3} = \begin{bmatrix}\cos\theta_1\cos(\theta_2+\theta_3) & -\cos\theta_1\sin(\theta_2+\theta_3) & \sin\theta_1 & \cos\theta_1\left[\cos(\theta_2+\theta_3)+\cos\theta_2\right]\\ \sin\theta_1\cos(\theta_2+\theta_3) & -\sin\theta_1\sin(\theta_2+\theta_3) & -\cos\theta_1 & \sin\theta_1\left[\cos(\theta_2+\theta_3)+\cos\theta_2\right]\\ \sin(\theta_2+\theta_3) & \cos(\theta_2+\theta_3) & 0 & \sin(\theta_2+\theta_3)+\sin\theta_2\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

This is of form.

$$
{}^{0}T_{3} = \begin{bmatrix}r_{11} & r_{12} & r_{13} & p_x\\ r_{21} & r_{22} & r_{23} & p_y\\ r_{31} & r_{32} & r_{33} & p_z\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

Where the position vector of the end effector is:

$$
\vec{o_3} = \begin{bmatrix}x\\ y\\ z\end{bmatrix} = \begin{bmatrix}\cos\theta_1\left[\cos(\theta_2+\theta_3)+\cos\theta_2\right]\\ \sin\theta_1\left[\cos(\theta_2+\theta_3)+\cos\theta_2\right]\\ \sin(\theta_2+\theta_3)+\sin\theta_2\end{bmatrix}
$$

4.3. Computing the Jacobian Matrix

The Jacobian matrix relates joint velocities to the linear and angular velocities of the end-effector.

$$
\begin{bmatrix}\vec{v}\\ \vec{\omega}\end{bmatrix} = J(\theta_1,\theta_2,\theta_3)\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\\ \dot{\theta}_3\end{bmatrix}
$$

To find the Jacobian, we extract points and axes to then compute with the below equations.

$$J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix}$$

Here, all are revolute joints. Hence, each column of the Jacobian is:

$$\text{Linear part:} \quad J_v^{(i)} = \mathbf{z}_{i-1} \times (\mathbf{o}_3 - \mathbf{o}_{i-1})$$

$$\text{Angular part:} \quad J_\omega^{(i)} = \mathbf{z}_{i-1}$$

So,

$$J_v = \begin{bmatrix} \mathbf{z}_0 \times (\mathbf{o}_3 - \mathbf{o}_0) & \mathbf{z}_1 \times (\mathbf{o}_3 - \mathbf{o}_1) & \mathbf{z}_2 \times (\mathbf{o}_3 - \mathbf{o}_2) \end{bmatrix}$$

$$J_\omega = \begin{bmatrix} \mathbf{z}_0 & \mathbf{z}_1 & \mathbf{z}_2 \end{bmatrix}$$

Extracting origins:

$$\mathbf{o}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{o}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{o}_2 = \begin{bmatrix} \cos\theta_1 \cos\theta_2 \\ \sin\theta_1 \cos\theta_2 \\ \sin\theta_2 \end{bmatrix}$$

$$\mathbf{o}_3 = \begin{bmatrix} \cos\theta_1 \left(\cos\theta_2 + \cos\theta_2 \cos\theta_3 - \sin\theta_2 \sin\theta_3\right) \\ \sin\theta_1 \left(\cos\theta_2 + \cos\theta_2 \cos\theta_3 - \sin\theta_2 \sin\theta_3\right) \\ \sin\theta_2 + \sin\theta_2 \cos\theta_3 + \cos\theta_2 \sin\theta_3 \end{bmatrix}$$

Extracting rotation axes:

$$\mathbf{z}_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\mathbf{z}_1 = \begin{bmatrix} \sin\theta_1 \\ -\cos\theta_1 \\ 0 \end{bmatrix}$$

$$\mathbf{z}_2 = \begin{bmatrix} \sin\theta_1 \\ -\cos\theta_1 \\ 0 \end{bmatrix}$$

With these, we can now make the final Jacobian, after simplifying:

$$J = \begin{bmatrix} -(\cos\theta_2 + \cos(\theta_2+\theta_3))\sin\theta_1 & -(\sin\theta_2 + \sin(\theta_2+\theta_3))\cos\theta_1 & -\sin(\theta_2+\theta_3)\cos\theta_1 \\ (\cos\theta_2 + \cos(\theta_2+\theta_3))\cos\theta_1 & -(\sin\theta_2 + \sin(\theta_2+\theta_3))\sin\theta_1 & -\sin(\theta_2+\theta_3)\sin\theta_1 \\ 0 & \cos\theta_2 + \cos(\theta_2+\theta_3) & \cos(\theta_2+\theta_3) \\ 0 & \sin\theta_1 & \sin\theta_1 \\ 0 & -\cos\theta_1 & -\cos\theta_1 \\ 1 & 0 & 0 \end{bmatrix}$$

4.4. Singularity Analysis

A joint configuration is singular if the determinant of the Jacobian for the joint configuration values is zero. Here, the robot may lose a Degree of Freedom or move rapidly out of control. Near singularities, a unique solution for Inverse Kinematics won't exist.

Let's divide the Jacobian into two parts and look at the Linear velocity sub-matrix, J_v.

$$J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix}$$

$$J_v = \begin{bmatrix} -(\cos\theta_2 + \cos(\theta_2+\theta_3))\sin\theta_1 & -(\sin\theta_2 + \sin(\theta_2+\theta_3))\cos\theta_1 & -\sin(\theta_2+\theta_3)\cos\theta_1 \\ (\cos\theta_2 + \cos(\theta_2+\theta_3))\cos\theta_1 & -(\sin\theta_2 + \sin(\theta_2+\theta_3))\sin\theta_1 & -\sin(\theta_2+\theta_3)\sin\theta_1 \\ 0 & \cos\theta_2 + \cos(\theta_2+\theta_3) & \cos(\theta_2+\theta_3) \end{bmatrix}$$

$$J_\omega = \begin{bmatrix} 0 & \sin\theta_1 & \sin\theta_1 \\ 0 & -\cos\theta_1 & -\cos\theta_1 \\ 1 & 0 & 0 \end{bmatrix}$$

- **Looking at J_ω**

  We can see that the second and third columns are identical. Therefore, these two columns are always linearly dependent. This implies that the rank of J_ω is 2 at most.

This means that this robot arm design inherently provides only 2 independent rotational degrees of freedom for the end-effector. It cannot achieve arbitrary 3D orientations because the last two of its joint axes are always parallel. This is a characteristic of the robot's design, not a configuration-dependent singularity that causes a "loss" of an already existing degree of freedom.

- **Looking at J_v:**

  We look for values where $det(J_v) = 0$, i.e. let's find values for which J_v becomes linearly dependent.

  Finding the determinant of J_v,

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - ceg - bdi - afh$$

  Using the above equation and simplifying the equation using MATLAB's Symbolic Math Toolbox, we get:

$$\det(J_v) = -(\cos\theta_3 + 1)\,(\sin(\theta_2 + \theta_3) - \sin\theta_2)$$

  Setting the expression to 0 to find singularities:

$$(\cos\theta_3 + 1)\,(\sin(\theta_2 + \theta_3) - \sin\theta_2) = 0$$

  Condition 1: $\cos\theta_3 + 1 = 0$

$$\cos\theta_3 = -1$$

  This happens when $\theta_3 = \pi + 2n\pi$, where n is any integer.

  Physically, we can see it as when $\theta_3 = +180^\circ / -180^\circ$, the second and third links of the robot arm become collinear, but pointed in opposite directions (folded back on themselves). This condition signifies a Wrist singularity. The robot loses the ability to move its end-effector in certain translational directions, as the degrees of freedom provided by θ_2 and θ_3 become redundant for specific motions. The robot's end-effector cannot move (or push/pull) effectively in the direction parallel to the now-aligned links.

This is also a Shoulder singularity, since the end-effector(wrist center) is on the rotation axis of the base. Any base rotation would make no difference, i.e., loss of X-Y translational control.

Condition 2: $\sin(\theta_2 + \theta_3) - \sin\theta_2 = 0$

$\sin(\theta_2 + \theta_3) = \sin\theta_2$

This is true when either:

- ➔ $\theta_3 = n\pi$

  Physically, link 2 and 3 align in the same line. The center of joint 2, center of joint 3 and the wrist center are in a single line. This is an Elbow singularity. The end-effector is at its furthest point along this line relative to joint 2. The robot loses the ability to effectively translate outward and inward along that same axis by independently controlling θ_2 and θ_3.

- ➔ $2\theta_2 + \theta_3 = \pi + 2n\pi$

  In this configuration, the end-effector's position is directly along the rotation axis of the base frame. This is a Shoulder singularity. Any base rotation would make no difference, i.e., loss of X-Y translational control.

Hence, we have three singularity configurations, and all the common types too(wrist, shoulder, elbow). These configurations lead to a loss of one or more degrees of freedom, restricting the end-effector’s ability to move in certain directions.

The MATLAB code to verify the same is given in Appendix B, along with the visualization of the determinant function, keeping one angle constant.

5. Part E: Kinematic Decoupling & Inverse Kinematics

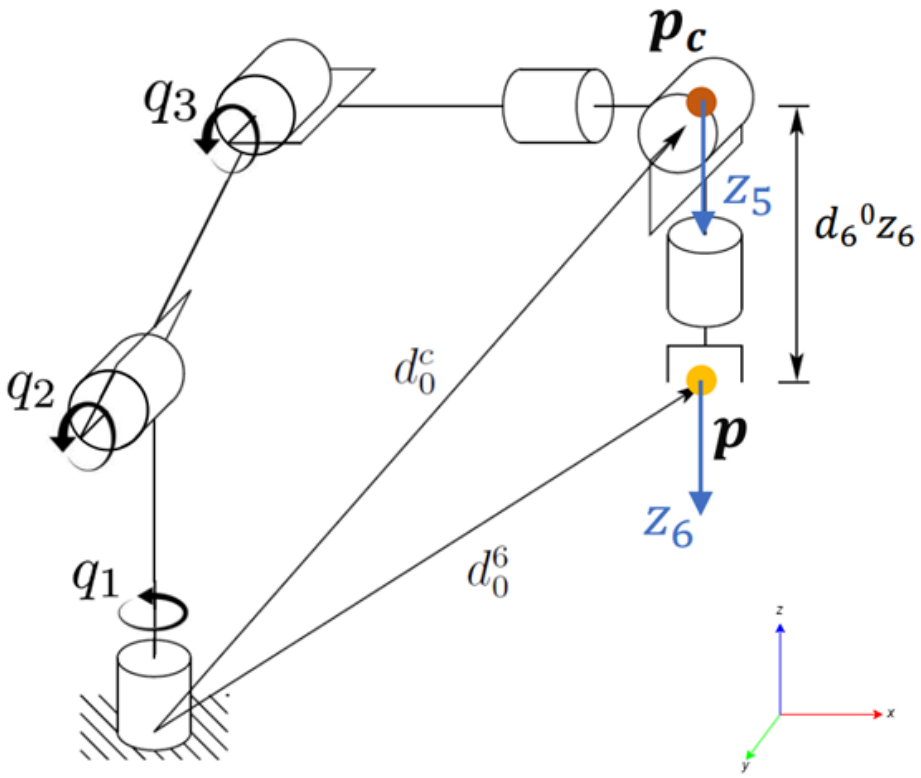


Figure 5.1 Robotic arm for inverse kinematics

The goal is to find the joint angles, given the end-effector pose. Using Kinematic Decoupling, we break the above problem into two parts:

1. Three link positioning (Position control): First solve q_1, q_2, q_3 for position of wrist center. We find this analytical solution in this section.
2. Wrist alignment (Orientation control): Then solve q_4, q_5, q_6 for needed orientation of the end-effector.

5.1. Looking at the diagram

- p (Yellow): End-effector position.
- p_c (Orange): Wrist center (decoupling point), where the axes of the last three joints intersect at a point. Function of q_1, q_2, q_3. It is the elbow-independent position. Solving for p_c gives the first three joint angles. Decoupling the wrist simplifies calculations and allows for more efficient and precise control of the manipulator.

- $d_0^6 z_6$: Last DH offset, i.e. constant vector of length d_6 along the tool axis z_6 expressed in the base frame. Magnitude gives the fixed distance from the wrist center frame{5} to the end-effector frame{6}. Decouples orientation from positioning.
- The vectors d_0^c and d_0^6 are simply the base to wrist centre and base to end-effector distance vectors. z_0^6 is the last column of the rotation matrix of the robot from base to end-effector.

5.2. Given assumptions

- The distance between joints 1 and 2 is zero. So, the shoulder sits exactly above the base axis.
- The distance between q_2 and q_3(shoulder to elbow) is α_1.
- The distance between q_3 and p_c(elbow to wrist center) is α_2.

5.3. Analytical Inverse Solution for position

The aim is to find the analytical equations for joint angles q_1, q_2 and q_3 such that

$$\mathbf{p}_c = \begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix}$$

- **Decoupling into planar geometry**

  Set the horizontal and vertical reach of the wrist center.

  Horizontal projection: $r = \sqrt{x_c^2 + y_c^2}$ – (1)

  Vertical reach: $s = z_c$ – (2)

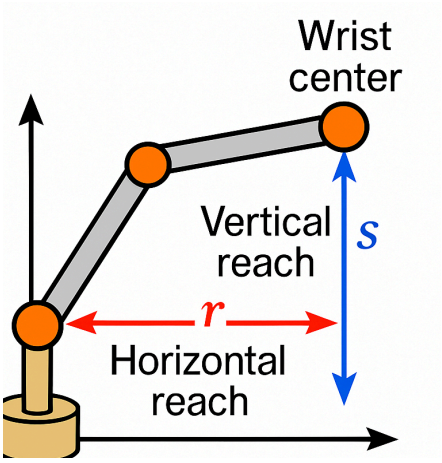


Figure 5.2 Reaches in the non-wrist arm

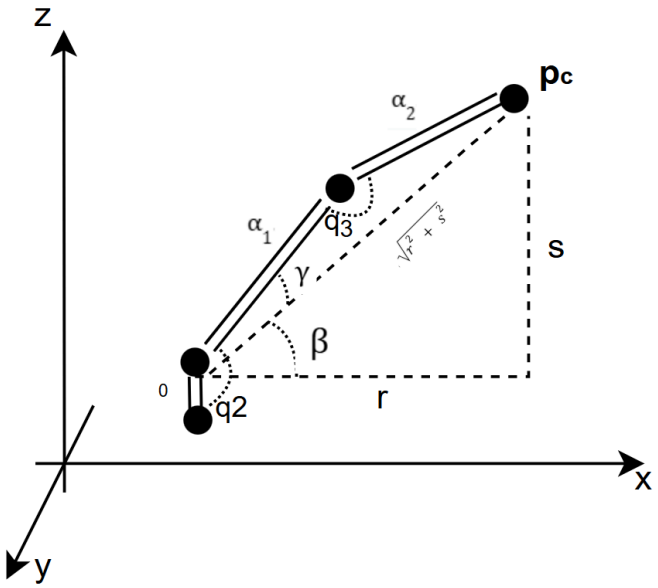


Figure 5.3 Geometry of the non-wrist arm

From the given figures:

Base Yaw (q_1) in only x-y plane: $q_1 = arctan\left(\frac{y_c}{x_c}\right)$ – (3)

Coming to the elbow triangle(links that q_3 connects). Using the law of cosines on the triangle (α_1, α_2, $\sqrt{r^2 + s^2}$):

$$D = \frac{r^2 + s^2 - \alpha_1^2 - \alpha_2^2}{2\alpha_1\alpha_2} = \cos q_3, \ |D| \leq 1 \quad \text{– (4)}$$

Now, Elbow angle,

$$q_3 = arctan\left(\frac{\pm\sqrt{1-D^2}}{D}\right) \quad -(5)$$

Here, + means elbow down and – means elbow up.

Moving on to Shoulder angle, subtracting angles after the inner opposite angle rule:

$q_2 = \beta - \gamma$, γ is the inner angle opposite α_2. β is the line-of-sight angle.

$$q_2 = arctan\left(\frac{s}{r}\right) - arctan\left(\frac{\alpha_2 \sin q_3}{\alpha_1 + \alpha_2 \cos q_3}\right) \quad -(6)$$

These three equations give both elbow-down and elbow-up inverse kinematic branches.

$$q_1 = arctan\left(\frac{y_c}{x_c}\right)$$

$$q_2 = arctan\left(\frac{s}{r}\right) - arctan\left(\frac{\alpha_2 \sin q_3}{\alpha_1 + \alpha_2 \cos q_3}\right)$$

$$q_3 = arctan\left(\frac{\pm\sqrt{1-D^2}}{D}\right), \quad \text{where } D = \frac{r^2 + s^2 - \alpha_1^2 - \alpha_2^2}{2\alpha_1\alpha_2},\ |D| \leq 1$$

Here is the output of the implementation of the same math in MATLAB. For the same code, refer to Appendix C.

```
Command Window
>> partE
D = 0.8000  (|D|<=1? true)

Elbow-DOWN solution:
  q1 =  35.538°   q2 =   6.503°   q3 =  36.870°
  FK check  (x,z) = (1.749, 0.800)

Elbow-UP   solution:
  q1 =  35.538°   q2 =  43.373°   q3 = -36.870°
  FK check  (x,z) = (1.749, 0.800)
>>
```

Figure 5.4 MATLAB output for the inverse kinematics problem

References

The MathWorks Inc. (2024). *MATLAB* (Version R2024a). Natick, MA: The MathWorks Inc.

Corke, P. I. (2017). *Robotics, vision and control: Fundamental algorithms in MATLAB* (2nd ed.). Springer.

The MathWorks Inc. (2024). *Symbolic Math Toolbox* (Version R2024a). Natick, MA: The MathWorks Inc.

JGraph Ltd. (n.d.). *diagrams.net*. Retrieved from https://www.diagrams.net/

Appendices

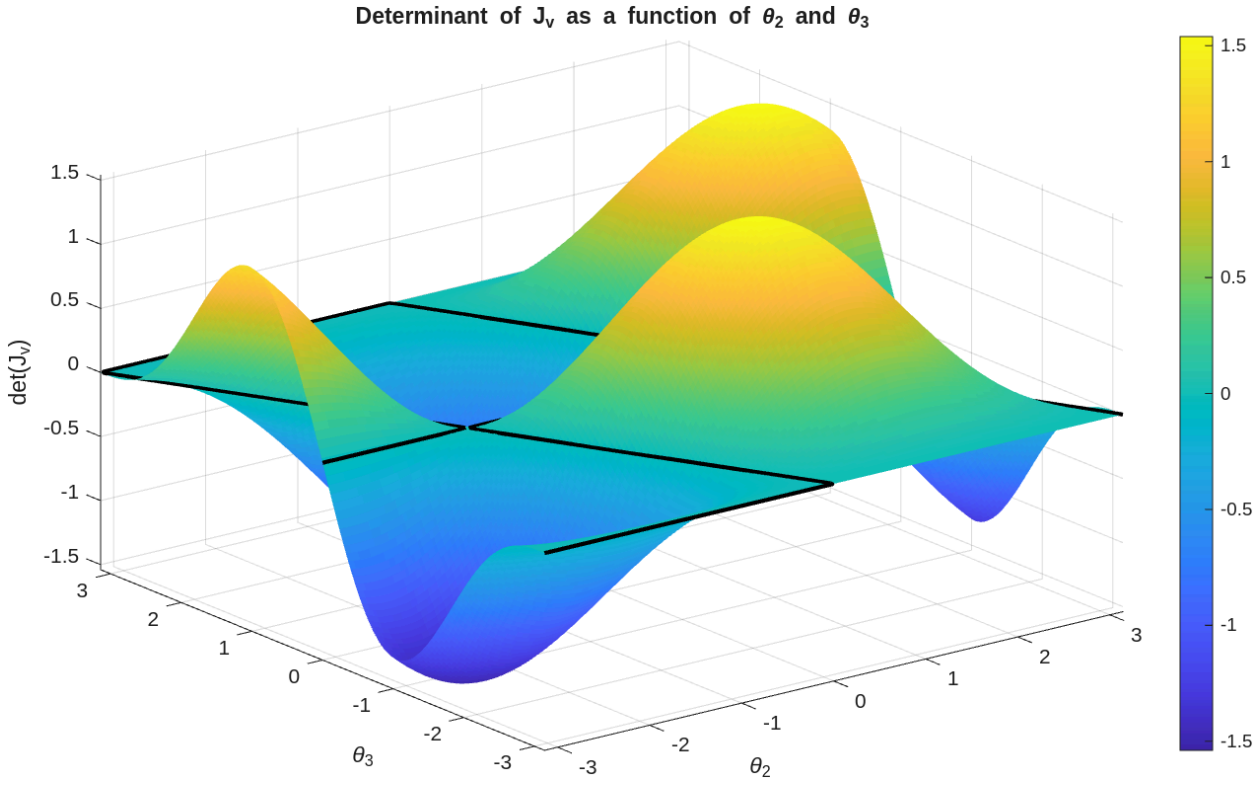
1. Appendix A: MATLAB code for Part B

```
% Project1PartB.m
% MTRN4230 Project 1 25T2
% Name: Darshan Komala Sreeramu
% Zid:  z5610741
r = -1 * sqrt(0.425^2 + 0.3922^2);  % -0.5783
% Define the new 5 joint arm
L(1) = Link('d', 0.1625, 'a', 0, 'alpha', pi/2);
L(2) = Link('d', 0, 'a', r, 'alpha', 0);
L(3) = Link('d', 0.1333, 'a', 0, 'alpha', pi/2);
L(4) = Link('d', 0.0997, 'a', 0, 'alpha', -1 * pi/2);
L(5) = Link('d', 0.0996, 'a', 0, 'alpha',  0   );
% Create 5-joint serial link
newArm = SerialLink([L(1) L(2) L(3) L(4) L(5)], 'name', 'Broken UR5e (5DOF)');
% Home joint configuration: θs are added here
qHome = deg2rad([0, 42.702 - 75, 47.298 - 105, 90, 0]);
% [0, -75, -105, 90, 0] is the given home config
% Forward kinematics
T = newArm.fkine(qHome);
disp('Final transformation matrix 0T5:');
disp(T);
pos = transl(T);
orientation = tr2rpy(T, 'zyx');
% Range conversion from [-pi, pi] to [0, 2pi]
orientation(3) = mod(orientation(3), 2*pi);
fprintf('Position(m) = [%.4f  %.4f  %.4f]\n', pos);
fprintf('RPY(rad)    = [%.3f  %.3f  %.3f]\n', orientation);
% Plot the arm in RVC
figure;
newArm.plot(qHome);
title('Broken UR5e (Elbow Collapsed)');
```

2. Appendix B: MATLAB code for Part D and Visualization

```
% Project1PartD.m

% MTRN4230 Project 1 25T2

% Name: Darshan Komala Sreeramu

% Zid:  z5610741

syms theta1 theta2 theta3 real

A = cos(theta2) + cos(theta2 + theta3);

B = sin(theta2) + sin(theta2 + theta3);

C = sin(theta2 + theta3);

s1 = sin(theta1);

c1 = cos(theta1);

% Define the jacobian

Jv = [ -A*s1, -B*c1, -C*c1;

      A*c1, -B*s1, -C*s1;

      0,     A,     cos(theta2 + theta3) ];

disp(Jv);

% Compute and simplify the determinant

detJv = det(Jv);

detJvSimp = simplify(detJv);

disp(detJvSimp);

% Convert to a MATLAB function for plotting

theta1Val = pi/4;

detJvFunc = matlabFunction(subs(detJvSimp, theta1, theta1Val), 'Vars', [theta2,
theta3]);

% Create a grid for theta2 and theta3

[Theta2, Theta3] = meshgrid(linspace(-pi, pi, 200), linspace(-pi, pi, 200));

DetVals = detJvFunc(Theta2, Theta3);
```

```
% Plot the determinant

figure;

surf(Theta2, Theta3, DetVals, 'EdgeColor', 'none');

hold on;

% Plot where determinant is zero

contour3(Theta2, Theta3, DetVals, [0 0], 'k', 'LineWidth', 2);

xlabel('\theta_2');

ylabel('\theta_3');

zlabel('det(J_v)');

title('Determinant of J_v as a function of \theta_2 and \theta_3');

colorbar;

view(3);

grid on;

hold off;
```



3. Appendix C: MATLAB code for Part E

```
% Project1PartE.m

% MTRN4230 Project 1 25T2

% Name: Darshan Komala Sreeramu

% Zid:  z5610741

pc = [1.4; 1; 0.8];    % target wrist-centre

r = norm(pc(1:2));    % horizontal reach

s = pc(3);    % vertical reach

a1 = 1;

a2 = 1;   % link lengths

% Base yaw

q1 = atan2(pc(2), pc(1));

% Two-link geometry

D = (r^2 + s^2 - a1^2 - a2^2)/(2*a1*a2);    % cosine of q3

fprintf('D = %.4f  (|D|<=1? %s)\n', D, string(abs(D)<=1))

q3_down = atan2(+sqrt(1-D^2), D);  % elbow down

q3_up = atan2(-sqrt(1-D^2), D);  % elbow up

beta = atan2(s, r);

gammaD = atan2(a2*sin(q3_down), a1 + a2*cos(q3_down));

gammaU = atan2(a2*sin(q3_up), a1 + a2*cos(q3_up));

q2Down = beta - gammaD;

q2Up = beta - gammaU;

% Forward check

xFK = @(q2,q3) cos(q1) + cos(q1+q2) + cos(q1+q2+q3);

zFK = @(q2,q3) sin(q2) + sin(q2+q3);

disp(' ')

fprintf('Elbow-DOWN solution:\n')

fprintf('  q1 = %7.3f°   q2 = %7.3f°   q3 = %7.3f°\n',...
```

```
       rad2deg([q1 q2Down q3_down]))

fprintf('  FK check  (x,z) = (%.3f, %.3f)\n',...

       xFK(q2Down,q3_down), zFK(q2Down,q3_down))

disp(' ')

fprintf('Elbow-UP   solution:\n')

fprintf('  q1 = %7.3f°   q2 = %7.3f°   q3 = %7.3f°\n', rad2deg([q1 q2Up
q3_up]))

fprintf('  FK check  (x,z) = (%.3f, %.3f)\n', xFK(q2Up,q3_up), zFK(q2Up,q3_up))
```