

AuE 8220- Autonomy: Mobility and Manipulation

Homework 6: Jacobians (Related Design and Control Issues)

Authors: Tanmay Samak, Chinmay Samak, Riccardo Setti, Olamide Akinyele

Problem 1:

A: Planar RR Manipulator

1-A-i Isotropy Index

Concept & Results: We defined a finely spaced workspace grid for the manipulator, taking into account its minimum ($a_1 - a_2$) and maximum ($a_1 + a_2$) reach. We then computed the joint angles at each of the workspace grid points using inverse kinematics for elbow-up configuration:

$$\theta_2 = -\cos^{-1}\left(\frac{grid_x^2 + grid_y^2 - a_1^2 - a_2^2}{2 * a_1 * a_2}\right)$$

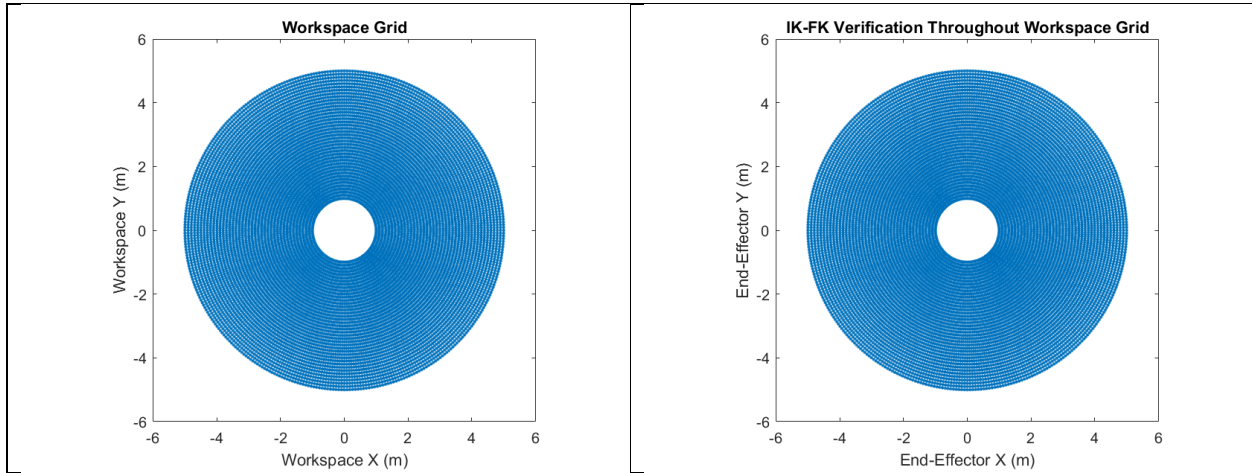
$$\theta_1 = \text{atan2}(grid_y, grid_x) - \text{atan2}(a_2 * \sin(\theta_2), (a_1 + a_2 * \cos(\theta_2)))$$

We then verified our approach by recovering the end-effector positions using forward kinematics:

$$x = a_1 * \cos(\theta_1) + a_2 * \cos(\theta_1 + \theta_2)$$

$$y = a_1 * \sin(\theta_1) + a_2 * \sin(\theta_1 + \theta_2)$$

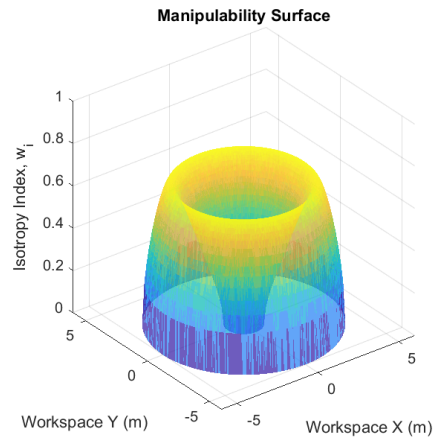
Following are the two plots:



Next, we computed the Jacobian matrix at each point in workspace grid and performed singular value decomposition (SVD) over it.

$$J = \begin{bmatrix} -a_1 * \sin(\theta_1) & -a_2 * \sin(\theta_1 + \theta_2) \\ a_1 * \cos(\theta_1) & a_2 * \cos(\theta_1 + \theta_2) \end{bmatrix}$$

We then computed the isotropy index at each point in workspace grid: $w_i = \frac{\sigma_{min}}{\sigma_{max}}$

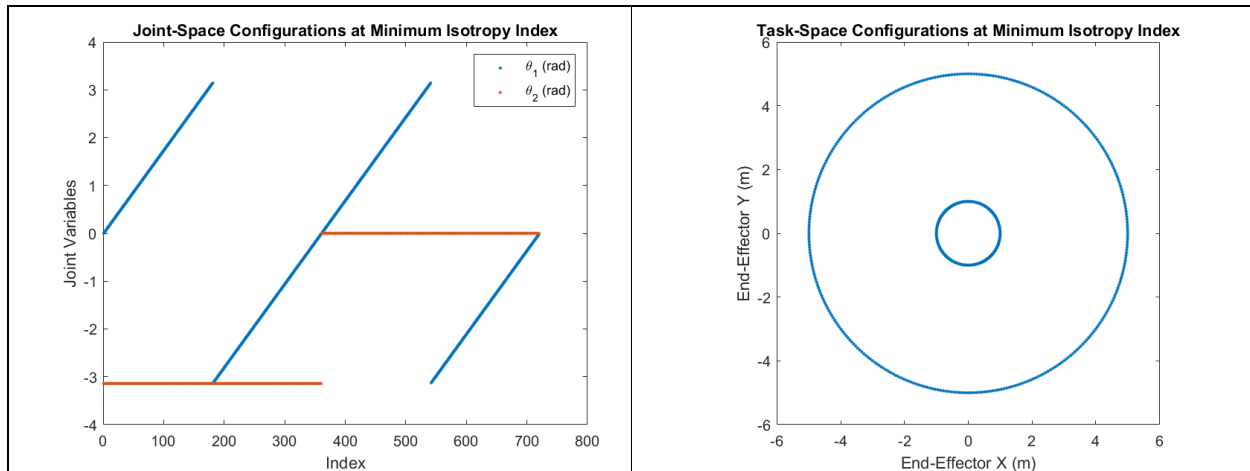


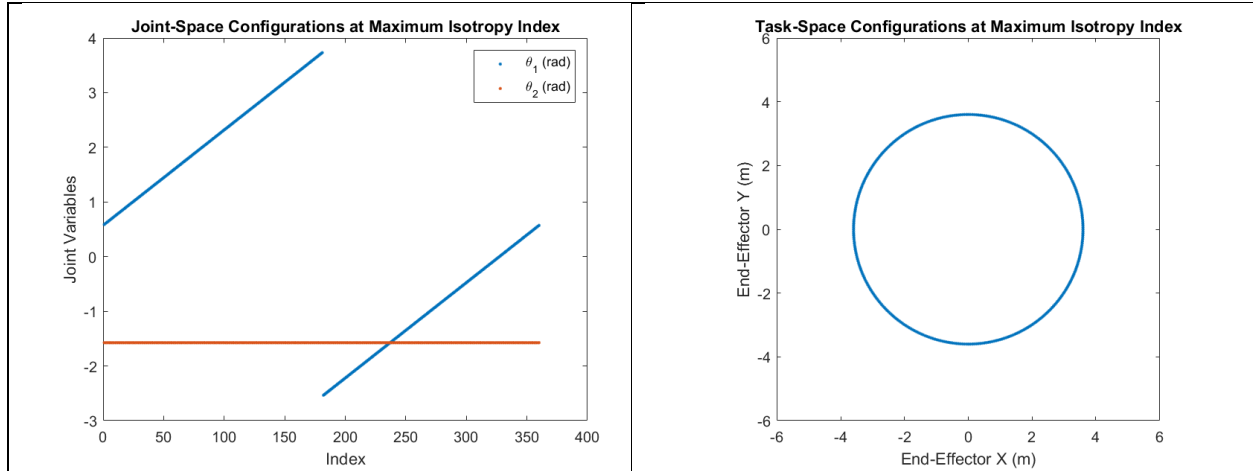
Intuitively, the plot above shows “degree of isotropy” for joint velocities (i.e., degree of equalness in traveling quickly in all possible directions). We measured the minimum and maximum values of isotropy index:

The minimum value of isotropy index is 0.0000.

The maximum value of isotropy index is 0.6667.

The isotropy index is minimum (zero) at points in workspace where the second link is either fully retracted (overlapping on the first link) or when it is fully outstretched. This is because the manipulator cannot move in the direction of link 2 anymore, it can only travel in a direction that is tangential to the direction of link 2. It is maximum (0.6667) somewhere between these two limits. We analyzed the joint-space as well as task-space coordinates at which the isotropy index was minimum and maximum:





MATLAB Code: P1_1.m lines 13-133

1-A-ii Yoshikawa's Measure of Manipulability

Concept & Results: We defined a finely spaced workspace grid for the manipulator, taking into account its minimum ($a_1 - a_2$) and maximum ($a_1 + a_2$) reach. We then computed the joint angles at each of the workspace grid points using inverse kinematics for elbow-up configuration:

$$\theta_2 = -\cos^{-1}\left(\frac{grid_x^2 + grid_y^2 - a_1^2 - a_2^2}{2 * a_1 * a_2}\right)$$

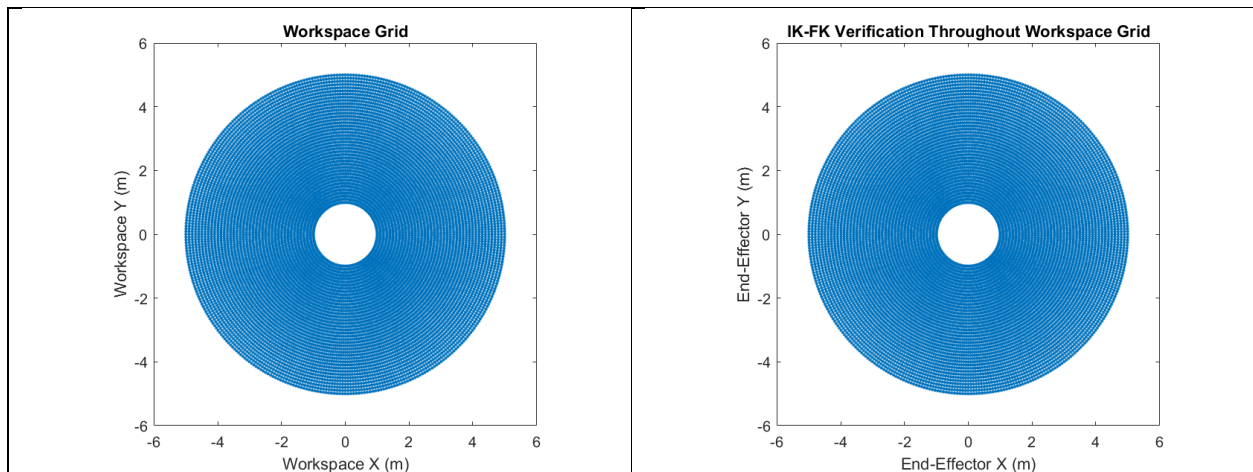
$$\theta_1 = \text{atan2}(grid_y, grid_x) - \text{atan2}(a_2 * \sin(\theta_2), (a_1 + a_2 * \cos(\theta_2)))$$

We then verified our approach by recovering the end-effector positions using forward kinematics:

$$x = a_1 * \cos(\theta_1) + a_2 * \cos(\theta_1 + \theta_2)$$

$$y = a_1 * \sin(\theta_1) + a_2 * \sin(\theta_1 + \theta_2)$$

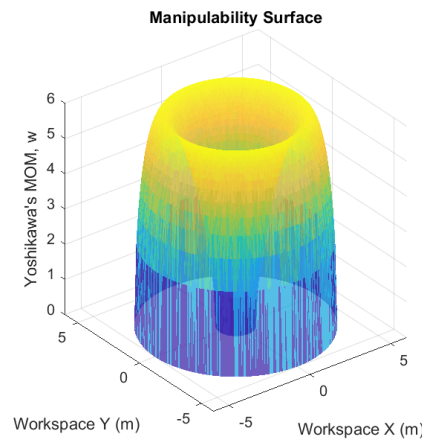
Following are the two plots:



Next, we computed the Jacobian matrix at each point in workspace grid and performed singular value decomposition (SVD) over it.

$$J = \begin{bmatrix} -a_1 * \sin(\theta_1) & -a_2 * \sin(\theta_1 + \theta_2) \\ a_1 * \cos(\theta_1) & a_2 * \cos(\theta_1 + \theta_2) \end{bmatrix}$$

We then computed the Yoshikawa's measure of manipulability (MOM) at each point in workspace grid:
 $w = \sqrt{\det(J * J^T)} = \det(\Sigma)$; where $U * \Sigma * V^T = SVD(J)$.

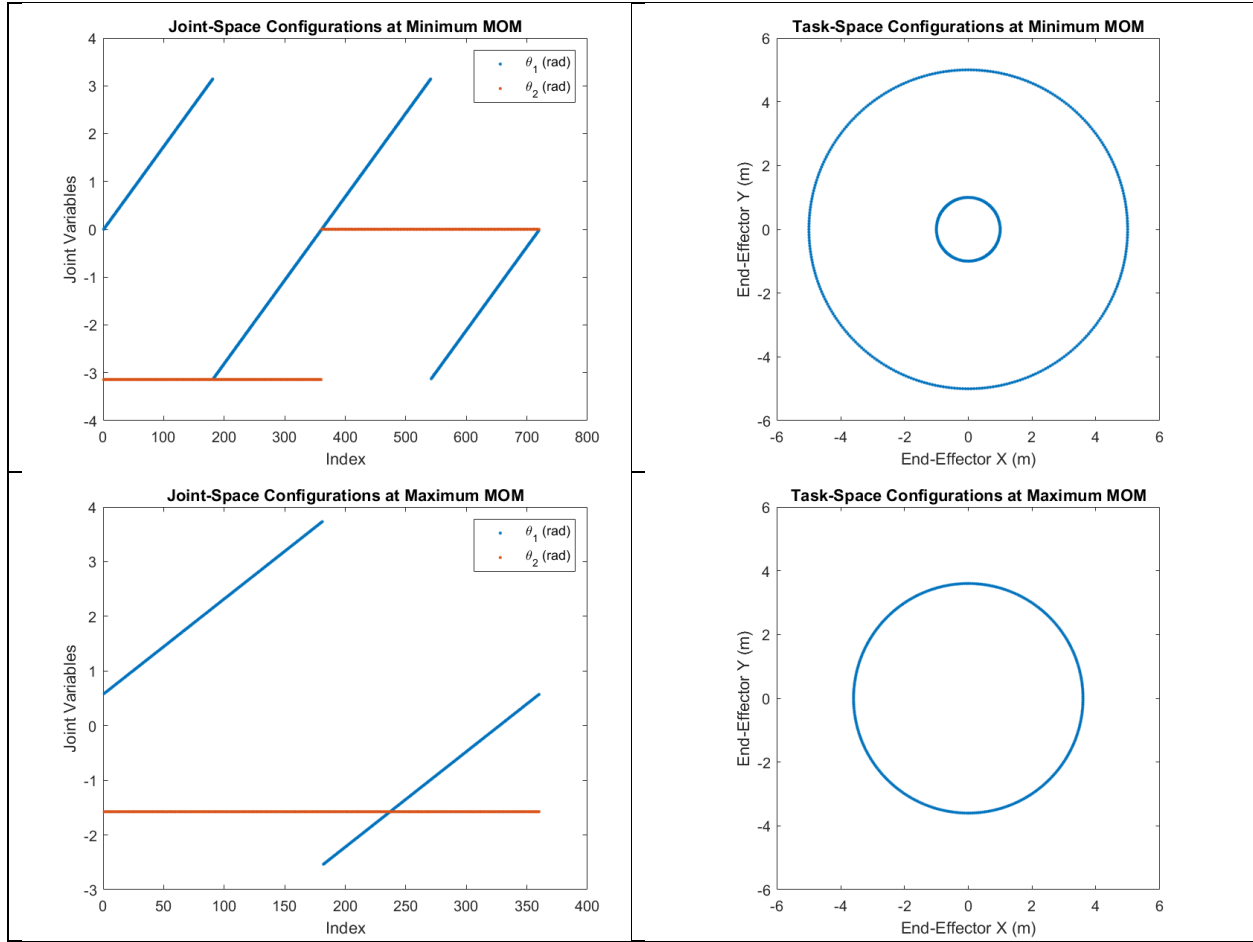


Intuitively, the plot above shows “combined degree of manipulability” for joint velocities (i.e., cumulative degree of quickness to travel in all possible directions), which is equivalent to the area/volume/hyper-volume of the manipulability ellipse/ellipsoid/hyper-ellipsoid at that point in workspace. We measured the minimum and maximum values of Yoshikawa's MOM:

The minimum value of MOM is 0.0000.

The maximum value of MOM is 6.0000.

Yoshikawa's MOM is minimum (zero) at points in workspace where the second link is either fully retracted (overlapping on the first link) or when it is fully outstretched. This is because the manipulator cannot move in the direction of link 2 anymore, it can only travel in a direction that is tangential to the direction of link 2 (as a result, the manipulability ellipse collapses to a line, and hence its area is zero). It is maximum (6) somewhere between these two limits. We analyzed the joint-space as well as task-space coordinates at which Yoshikawa's MOM was minimum and maximum:



MATLAB Code: P1_1.m lines 134-250

1-A-iii Manipulability Ellipsoids

Concept & Results: We defined a sparsely spaced workspace grid for the manipulator, taking into account its minimum ($a_1 - a_2$) and maximum ($a_1 + a_2$) reach. We then computed the joint angles at each of the workspace grid points using inverse kinematics for elbow-up configuration:

$$\theta_2 = -\cos^{-1}\left(\frac{grid_x^2 + grid_y^2 - a_1^2 - a_2^2}{2 * a_1 * a_2}\right)$$

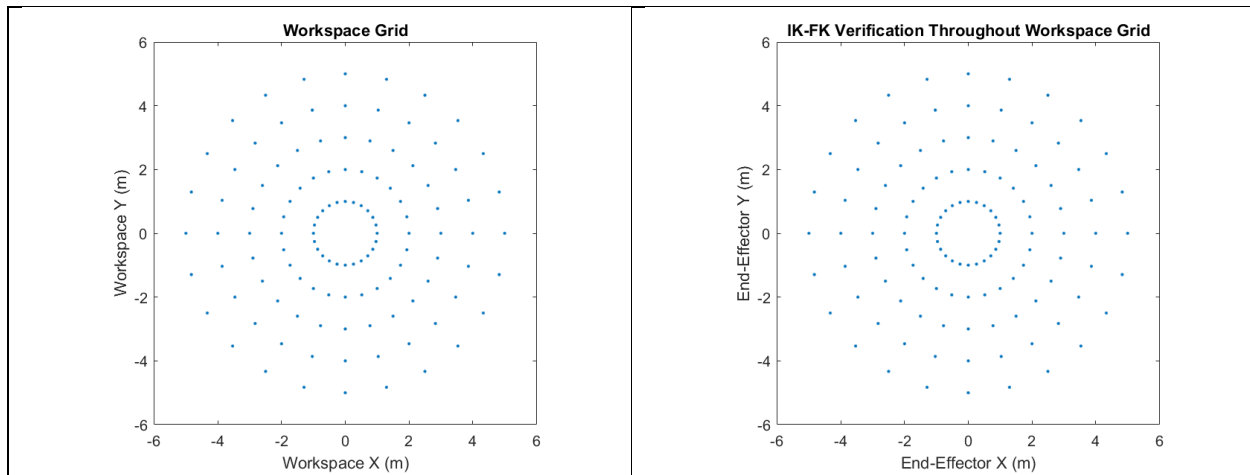
$$\theta_1 = \text{atan2}(grid_y, grid_x) - \text{atan2}(a_2 * \sin(\theta_2), a_1 + a_2 * \cos(\theta_2))$$

We then verified our approach by recovering the end-effector positions using forward kinematics:

$$x = a_1 * \cos(\theta_1) + a_2 * \cos(\theta_1 + \theta_2)$$

$$y = a_1 * \sin(\theta_1) + a_2 * \sin(\theta_1 + \theta_2)$$

Following are the two plots:



Next, we computed the Jacobian matrix at each point in workspace grid and performed singular value decomposition (SVD) over it.

$$J = \begin{bmatrix} -a_1 * \sin(\theta_1) & -a_2 * \sin(\theta_1 + \theta_2) \\ a_1 * \cos(\theta_1) & a_2 * \cos(\theta_1 + \theta_2) \end{bmatrix}$$

$$U * \Sigma * V^T = SVD(J)$$

We then computed and plotted the scaled manipulability ellipsoids at each point in the workspace grid:

The scale for manipulability ellipsoids is 1:15 m/s.



Intuitively, the plot above shows “degree of manipulability” for joint velocities (i.e., degree of quickness to travel in all possible directions).

MATLAB Code: P1_1.m lines 251-334

Problem 1:

B: Planar RP Manipulator

1-B-i Isotropy Index

Concept & Results: We defined a finely spaced workspace grid for the manipulator, taking into account its minimum ($d_{min} - d_{max}$) and maximum ($a_1 + a_2$) reach. We then computed the joint angles at each of the workspace grid points using inverse kinematics:

$$\theta_1 = \text{atan2}(\text{grid}_y, \text{grid}_x)$$

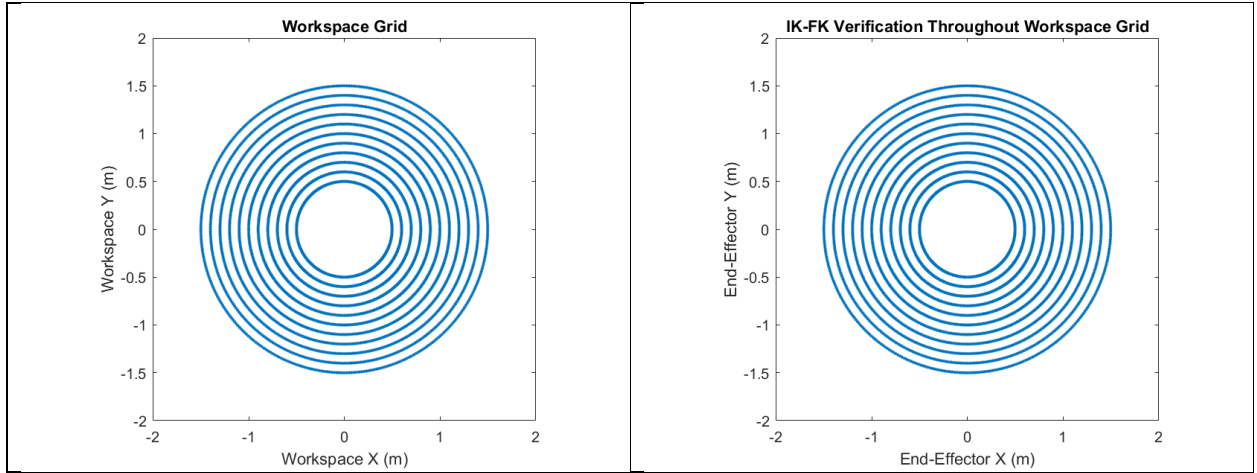
$$d_2 = \sqrt{\text{grid}_x^2 + \text{grid}_y^2}$$

We then verified our approach by recovering the end-effector positions using forward kinematics:

$$x = d_2 * \cos(\theta_1)$$

$$y = d_2 * \sin(\theta_1)$$

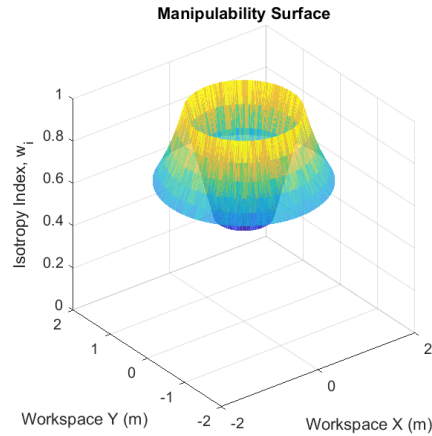
Following are the two plots:



Next, we computed the Jacobian matrix at each point in workspace grid and performed singular value decomposition (SVD) over it.

$$J = \begin{bmatrix} -d_2 * \sin(\theta_1) & \cos(\theta_1) \\ d_2 * \cos(\theta_1) & \sin(\theta_1) \end{bmatrix}$$

We then computed the isotropy index at each point in workspace grid: $w_i = \frac{\sigma_{min}}{\sigma_{max}}$

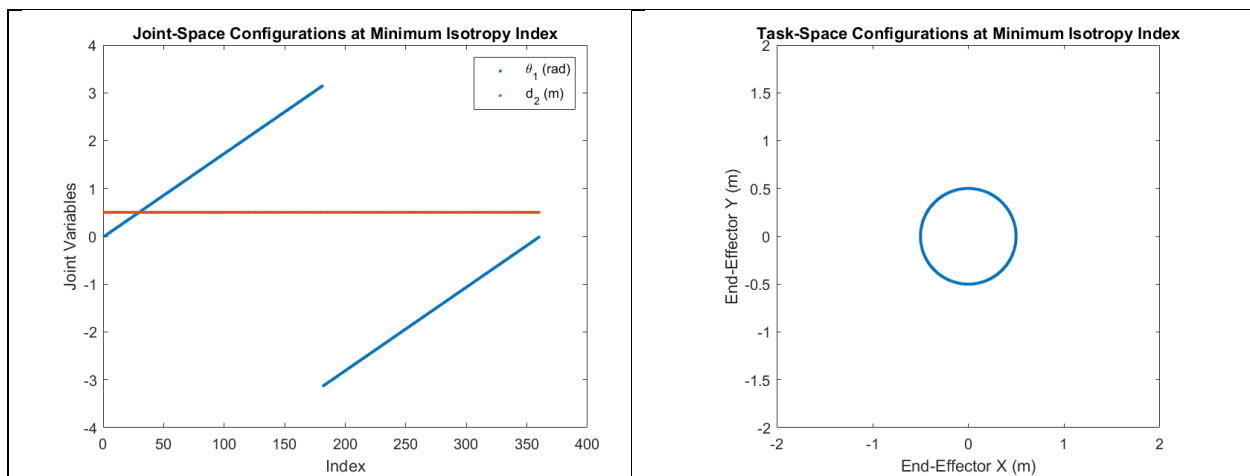


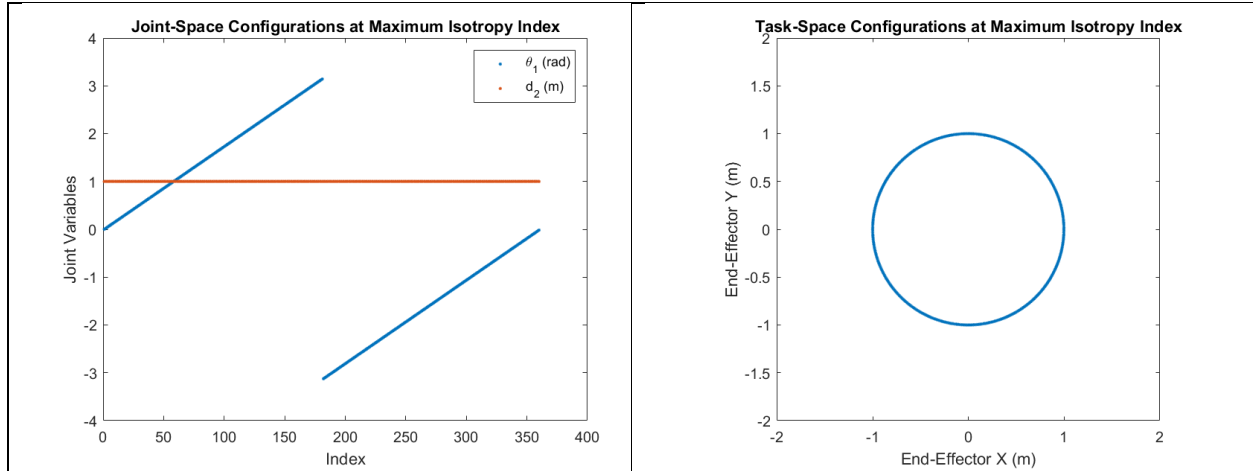
Intuitively, the plot above shows “degree of isotropy” for joint velocities (i.e., degree of equalness in traveling quickly in all possible directions). We measured the minimum and maximum values of isotropy index:

The minimum value of isotropy index is 0.5000.

The maximum value of isotropy index is 1.0000.

The isotropy index is minimum (0.5) at points in workspace where the prismatic joint is completely retracted. This is because the manipulator can move fairly quickly in the direction of prismatic joint, but it cannot travel as quickly in a direction that is tangential to the direction of prismatic joint. It is maximum when the prismatic joint is extended half-way. This is because the robot can travel equally quickly in all the directions (isotropy index = 1). We analyzed the joint-space as well as task-space coordinates at which the isotropy index was minimum and maximum:





MATLAB Code: P1_2.m lines 13-133

1-B-ii Yoshikawa's Measure of Manipulability

Concept & Results: We defined a finely spaced workspace grid for the manipulator, taking into account its minimum ($d_{min} - d_{max}$) and maximum ($a_1 + a_2$) reach. We then computed the joint angles at each of the workspace grid points using inverse kinematics:

$$\theta_1 = \text{atan2}(\text{grid}_y, \text{grid}_x)$$

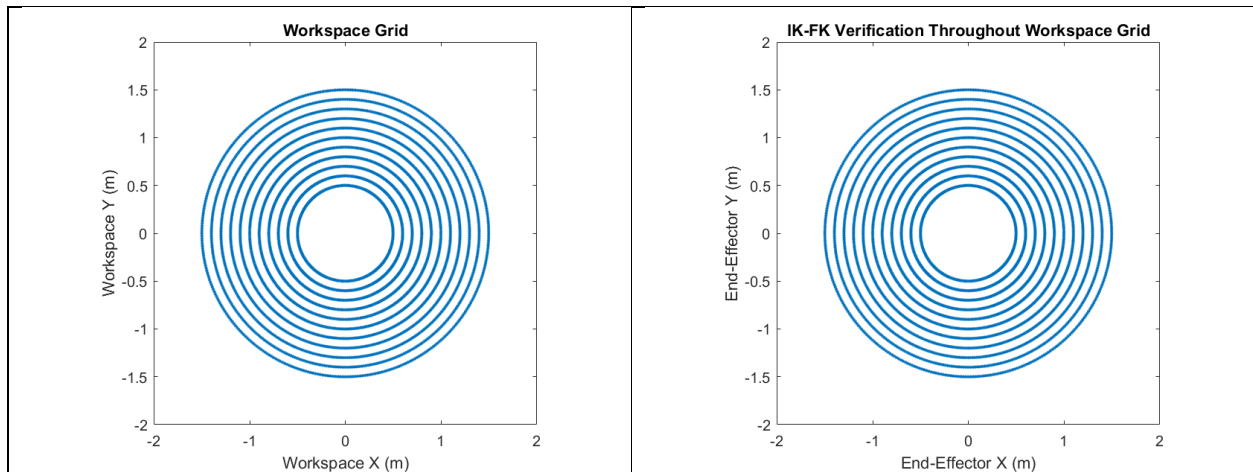
$$d_2 = \sqrt{\text{grid}_x^2 + \text{grid}_y^2}$$

We then verified our approach by recovering the end-effector positions using forward kinematics:

$$x = d_2 * \cos(\theta_1)$$

$$y = d_2 * \sin(\theta_1)$$

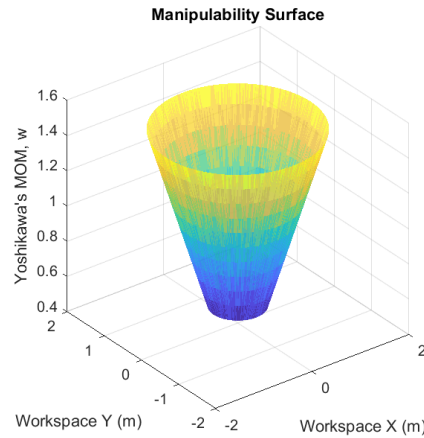
Following are the two plots:



Next, we computed the Jacobian matrix at each point in workspace grid and performed singular value decomposition (SVD) over it.

$$J = \begin{bmatrix} -d_2 * \sin(\theta_1) & \cos(\theta_1) \\ d_2 * \cos(\theta_1) & \sin(\theta_1) \end{bmatrix}$$

We then computed the Yoshikawa's measure of manipulability (MOM) at each point in workspace grid: $w = \sqrt{\det(J * J^T)} = \det(\Sigma)$; where $U * \Sigma * V^T = SVD(J)$.

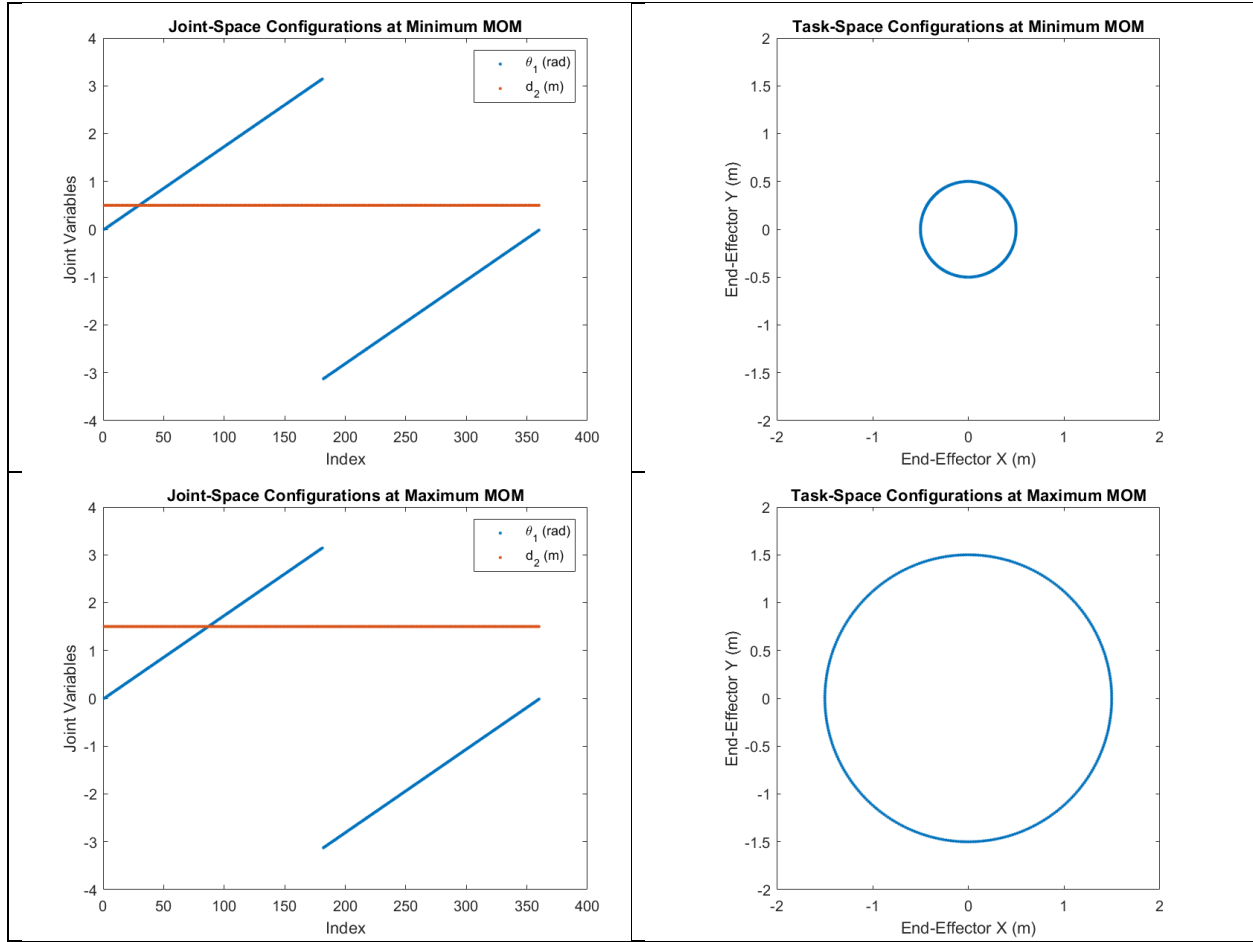


Intuitively, the plot above shows “combined degree of manipulability” for joint velocities (i.e., cumulative degree of quickness to travel in all possible directions), which is equivalent to the area/volume/hyper-volume of the manipulability ellipse/ellipsoid/hyper-ellipsoid at that point in workspace. We measured the minimum and maximum values of Yoshikawa's MOM:

The minimum value of MOM is 0.5000.

The maximum value of MOM is 1.5000.

Yoshikawa's MOM is minimum (0.5) at points in workspace where the prismatic joint is fully retracted. This is because the manipulator can move fairly quickly in the direction of prismatic joint, but it cannot travel as quickly in a direction that is tangential to the direction of prismatic joint. It is maximum when the prismatic joint is fully extended. This is because the manipulator is most “manipulable” in a direction that is tangential to the direction of prismatic joint. We analyzed the joint-space as well as task-space coordinates at which Yoshikawa's MOM was minimum and maximum:



MATLAB Code: P1_2.m lines 134-250

1-B-iii Manipulability Ellipsoids

Concept & Results: We defined a sparsely spaced workspace grid for the manipulator, taking into account its minimum ($d_{min} - d_{max}$) and maximum ($a_1 + a_2$) reach. We then computed the joint angles at each of the workspace grid points using inverse kinematics:

$$\theta_1 = \text{atan2}(\text{grid}_y, \text{grid}_x)$$

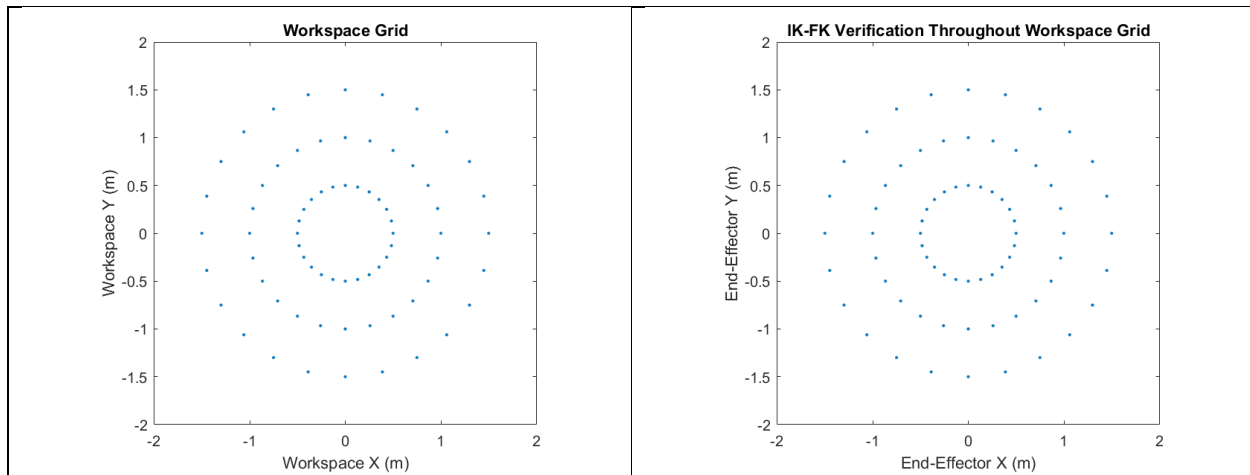
$$d_2 = \sqrt{\text{grid}_x^2 + \text{grid}_y^2}$$

We then verified our approach by recovering the end-effector positions using forward kinematics:

$$x = d_2 * \cos(\theta_1)$$

$$y = d_2 * \sin(\theta_1)$$

Following are the two plots:



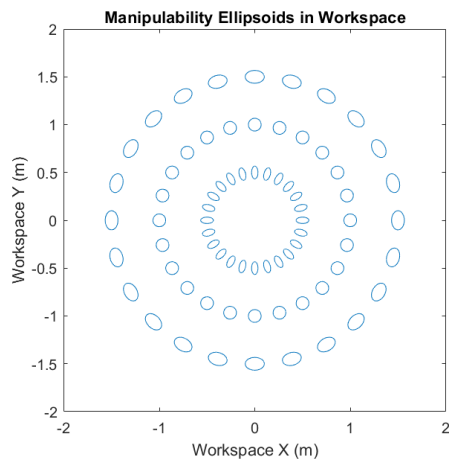
Next, we computed the Jacobian matrix at each point in workspace grid and performed singular value decomposition (SVD) over it.

$$J = \begin{bmatrix} -d_2 * \sin(\theta_1) & \cos(\theta_1) \\ d_2 * \cos(\theta_1) & \sin(\theta_1) \end{bmatrix}$$

$$U * \Sigma * V^T = SVD(J)$$

We then computed and plotted the scaled manipulability ellipsoids at each point in the workspace grid:

The scale for manipulability ellipsoids is 1:15 m/s.



Intuitively, the plot above shows “degree of manipulability” for joint velocities (i.e., degree of quickness to travel in all possible directions).

Note: Since we have a soft constraint (and not a hard constraint) on the prismatic joint that it can travel between 0.5 and 1.5, it cannot be accounted for in the degree of manipulability.

MATLAB Code: P1_2.m lines 251-334

Problem 2:

A: Circular Trajectory

2-A-i Joint-Space Closed-Loop Control

Concept:

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}_{2 \times 1} = \begin{bmatrix} \dot{\theta}_{1d} \\ \dot{\theta}_{2d} \end{bmatrix}_{2 \times 1} + \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}_{2 \times 2} \begin{bmatrix} \theta_{1d} - \theta_1 \\ \theta_{2d} - \theta_2 \end{bmatrix}_{2 \times 1}$$

Where $\begin{bmatrix} \dot{\theta}_{1d} \\ \dot{\theta}_{2d} \end{bmatrix}_{2 \times 1} = \begin{bmatrix} J^{-1} \end{bmatrix}_{2 \times 2} \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix}_{2 \times 1}$

$k_i = 1/\tau_i$
↑ pole ↑ time constant

→ of the form:

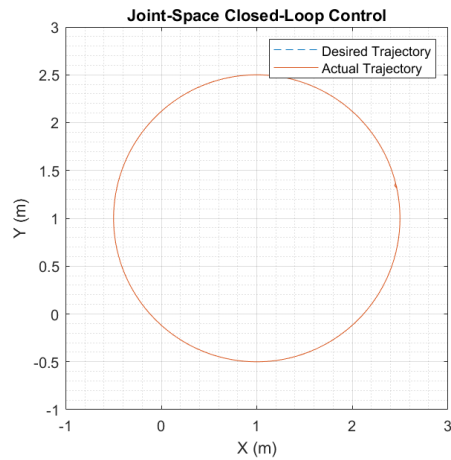
$$\dot{q} = \dot{q}_d + [k][q_d - q]$$
$$\Rightarrow \underbrace{\dot{q}_d - \dot{q}}_{\dot{q}_e} + [k] \underbrace{[q_d - q]}_{q_e} = 0$$
$$\Rightarrow \dot{q}_e + k q_e = 0$$

$\xrightarrow{\text{soln}} q_e(t) = q_e(0) e^{-[k]t}$

error @ t initial error decay with 't'

$\left. \begin{array}{l} @ t=0, q_e = q_e(0) \\ @ t=\infty, q_e = 0 \end{array} \right\} \begin{array}{l} \text{As "t" goes from 0 to } \infty, \\ \text{"q}_e\text{" goes from initial error to 0} \end{array}$

Results:



The end-effector has a small error initially, which is drawn very close to zero as it is controlled (with the implemented closed-loop controller in joint-space) to track the desired circular trajectory.

MATLAB Code: P2_1.m lines 40-65

2-A-ii Task-Space Closed-Loop Control

Concept:

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix}_{2 \times 1} = \begin{bmatrix} J^{-1} \end{bmatrix}_{2 \times 2} \left(\begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix}_{2 \times 1} + \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}_{2 \times 2} \begin{bmatrix} x_d - x \\ y_d - y \end{bmatrix}_{2 \times 1} \right)$$

where $\begin{bmatrix} x \\ y \end{bmatrix}_{2 \times 1} = \begin{bmatrix} FK(\theta_1, \theta_2) \end{bmatrix}_{2 \times 1}$

$k_i = 1/\tau_i$
 ↑ pole ↑ time constant

of the form:

$$J^{-1}(\dot{x}_d + [K][x_d - x]) = \ddot{q}$$

$$\Rightarrow \dot{x}_d + [K][x_d - x] = \underbrace{J\ddot{q}}_{\dot{x}}$$

$$\Rightarrow \underbrace{\dot{x}_d - \dot{x}}_{\dot{x}_e} + \underbrace{[K][x_d - x]}_{x_e} = 0$$

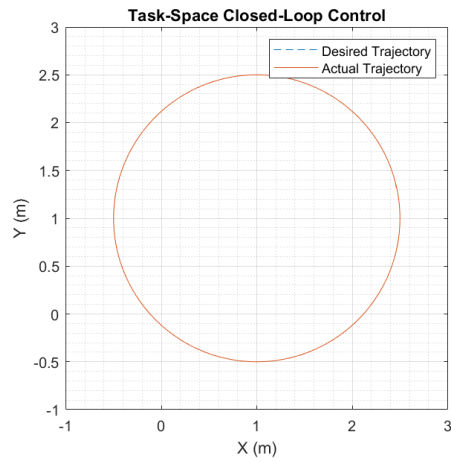
$$\Rightarrow \dot{x}_e + Kx_e = 0$$

solⁿ $x_e(t) = x_e(0) e^{-[K]t}$

error @ 't' initial error decay with 't'

$\left. \begin{array}{l} @t=0, x_e = x_e(0) \\ @t=\infty, x_e = 0 \end{array} \right\}$ As "t" goes from 0 to ∞ "x_e" goes from initial error to 0

Results:



The end-effector has a small error initially, which is drawn very close to zero as it is controlled (with the implemented closed-loop controller in task-space) to track the desired circular trajectory. It was noticed that the initial error in this case was less than that in case of joint-space closed-loop control, since the controller was directly acting on task-space error.

MATLAB Code: `P2_1.m` lines 66-91

Problem 2:

B: Elliptical Trajectory

2-B-i Joint-Space Closed-Loop Control

Concept:

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}_{2 \times 1} = \begin{bmatrix} \dot{\theta}_{1d} \\ \dot{\theta}_{2d} \end{bmatrix}_{2 \times 1} + \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}_{2 \times 2} \begin{bmatrix} (\theta_{1d} - \theta_1) \\ (\theta_{2d} - \theta_2) \end{bmatrix}_{2 \times 1}$$

where $\begin{bmatrix} \dot{\theta}_{1d} \\ \dot{\theta}_{2d} \end{bmatrix}_{2 \times 1} = \begin{bmatrix} J^{-1} \end{bmatrix}_{2 \times 2} \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix}_{2 \times 1}$

$k_i = 1/\tau_i$
↑ pole ↑ time constant

→ of the form:

$$\dot{q} = \dot{q}_d + [k][q_d - q]$$
$$\Rightarrow \underbrace{\dot{q}_d - \dot{q}}_{\dot{q}_e} + [k] \underbrace{[q_d - q]}_{q_e} = 0$$
$$\Rightarrow \dot{q}_e + k q_e = 0$$

solⁿ $\rightarrow q_e(t) = q_e(0) e^{-[k]t}$

error @ t initial error decay with 't'

$$\left. \begin{array}{l} @ t=0, q_e = q_e(0) \\ @ t=\infty, q_e = 0 \end{array} \right\} \begin{array}{l} \text{As "t" goes from 0 to } \infty, \\ \text{"q}_e\text{" goes from initial error to 0} \end{array}$$

Results:



The end-effector has a small error initially, which is drawn very close to zero as it is controlled (with the implemented closed-loop controller in joint-space) to track the desired elliptical trajectory.

MATLAB Code: P2_2.m lines 87-140

2-B-ii Task-Space Closed-Loop Control

Concept:

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix}_{2 \times 1} = \begin{bmatrix} J^{-1} \end{bmatrix}_{2 \times 2} \left(\begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix}_{2 \times 1} + \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}_{2 \times 2} \begin{bmatrix} x_d - x \\ y_d - y \end{bmatrix}_{2 \times 1} \right)$$

where $\begin{bmatrix} x \\ y \end{bmatrix}_{2 \times 1} = \begin{bmatrix} FK(\theta_1, \theta_2) \end{bmatrix}_{2 \times 1}$

$k_i = 1/\tau_i$
 ↑ pole ↑ time constant

of the form:

$$J^{-1}(\dot{x}_d + [k][x_d - x]) = \dot{q}$$

$$\Rightarrow \dot{x}_d + [k][x_d - x] = \underbrace{J\dot{q}}_{\dot{x}}$$

$$\Rightarrow \underbrace{\dot{x}_d - \dot{x}}_{\dot{x}_e} + \underbrace{[k][x_d - x]}_{x_e} = 0$$

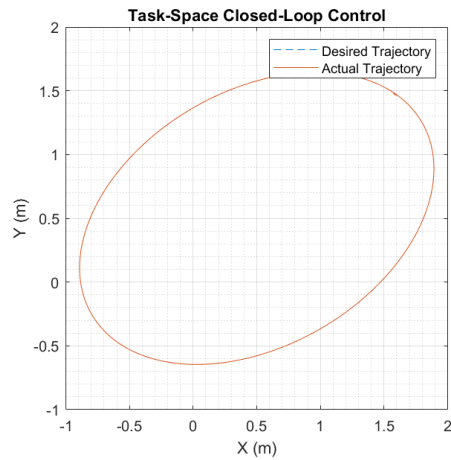
$$\Rightarrow \dot{x}_e + Kx_e = 0$$

solⁿ $x_e(t) = x_e(0) e^{-[k]t}$

error @ 't' initial error decay with 't'

$\left. \begin{array}{l} @t=0, x_e = x_e(0) \\ @t=\infty, x_e = 0 \end{array} \right\}$ As 't' goes from 0 to ∞ , x_e goes from initial error to 0

Results:

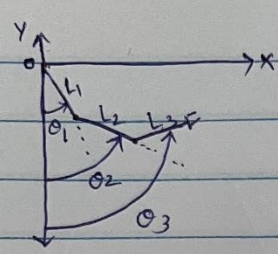


The end-effector has a small error initially, which is drawn very close to zero as it is controlled (with the implemented closed-loop controller in task-space) to track the desired elliptical trajectory. It was noticed that the initial error in this case was less than that in case of joint-space closed-loop control, since the controller was directly acting on task-space error.

MATLAB Code: `P2_2.m` lines 141-194

Problem 3:

Concept:



w.r.t +ve x axis
(Absolute)

(Relative)

$\theta_1' = \theta_1 + \frac{3\pi}{2}$ $\theta_2' = \theta_2 + \frac{3\pi}{2}$ $\theta_3' = \theta_3 + \frac{3\pi}{2}$ <p style="text-align: center;">offset</p>	$\theta_1'' = \theta_1'$ $\theta_2'' = \theta_2' - \theta_1'$ $\theta_3'' = \theta_3' - \theta_2'$
---	--

FK:

Pos: $x = L_1 \sin \theta_1 + L_2 \sin \theta_2 + L_3 \sin \theta_3$
 $y = -L_1 \cos \theta_1 - L_2 \cos \theta_2 - L_3 \cos \theta_3$

Vel: $\dot{x} = L_1 \cos \theta_1 \dot{\theta}_1 + L_2 \cos \theta_2 \dot{\theta}_2 + L_3 \cos \theta_3 \dot{\theta}_3$
 $\dot{y} = L_1 \sin \theta_1 \dot{\theta}_1 + L_2 \sin \theta_2 \dot{\theta}_2 + L_3 \sin \theta_3 \dot{\theta}_3$

$\dot{\mathbf{x}} = \mathbf{J} \dot{\mathbf{q}}$

A

B (i)

B (ii)

C

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} L_1 C_1 & L_2 C_2 & L_3 C_3 \\ L_1 S_1 & L_2 S_2 & L_3 S_3 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

2×1 2×3 3×1

$\dot{\mathbf{q}} = \text{pinv}(\mathbf{J}) \dot{\mathbf{x}} = \mathbf{J}^{\#} \dot{\mathbf{x}}$

B (i) $\dot{\theta}_1 = 0 \Rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} L_2 C_2 & L_3 C_3 \\ L_2 S_2 & L_3 S_3 \end{bmatrix} \begin{bmatrix} \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \Rightarrow \dot{\mathbf{q}} = \mathbf{J}^{-1} \dot{\mathbf{x}}, \dot{\mathbf{q}}(1) = 0$

B (ii) $\dot{\theta}_3 = -\dot{\theta}_2 \Rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} L_1 C_1 & (L_2 C_2 - L_3 C_3) \\ L_1 S_1 & (L_2 S_2 - L_3 S_3) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \Rightarrow \dot{\mathbf{q}} = \mathbf{J}^{-1} \dot{\mathbf{x}}, \dot{\mathbf{q}}(3) = -\dot{\mathbf{q}}(2)$

C $\dot{\mathbf{q}} = \mathbf{J}^{\#} \dot{\mathbf{x}} + \underbrace{[\mathbf{I} - \mathbf{J}^{\#} \mathbf{J}]}_{\text{null space filter}} \cdot (-\nabla V)$

where $(-\nabla V) = \begin{bmatrix} -\frac{\partial V}{\partial \theta_1} \\ -\frac{\partial V}{\partial \theta_2} \\ -\frac{\partial V}{\partial \theta_3} \end{bmatrix} = \begin{bmatrix} -k_1 \theta_1 \\ -k_2 \theta_2 \\ -k_3 \theta_3 \end{bmatrix}$

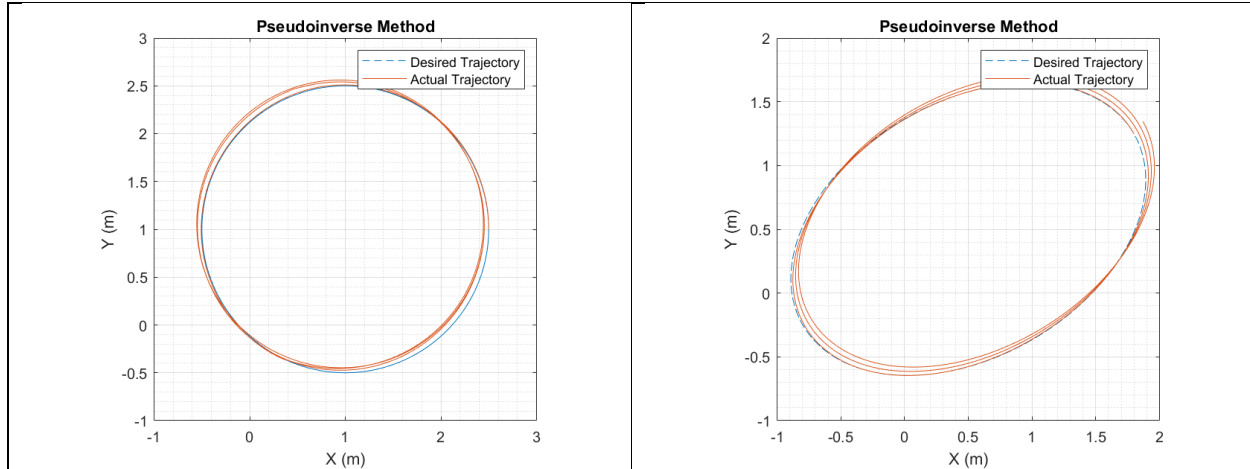
negative gradient of artificial potential function

where $V = \frac{k_1}{2} \theta_1^2 + \frac{k_2}{2} \theta_2^2 + \frac{k_3}{2} \theta_3^2$

artificial potential function

A: Pseudoinverse Method

Results:



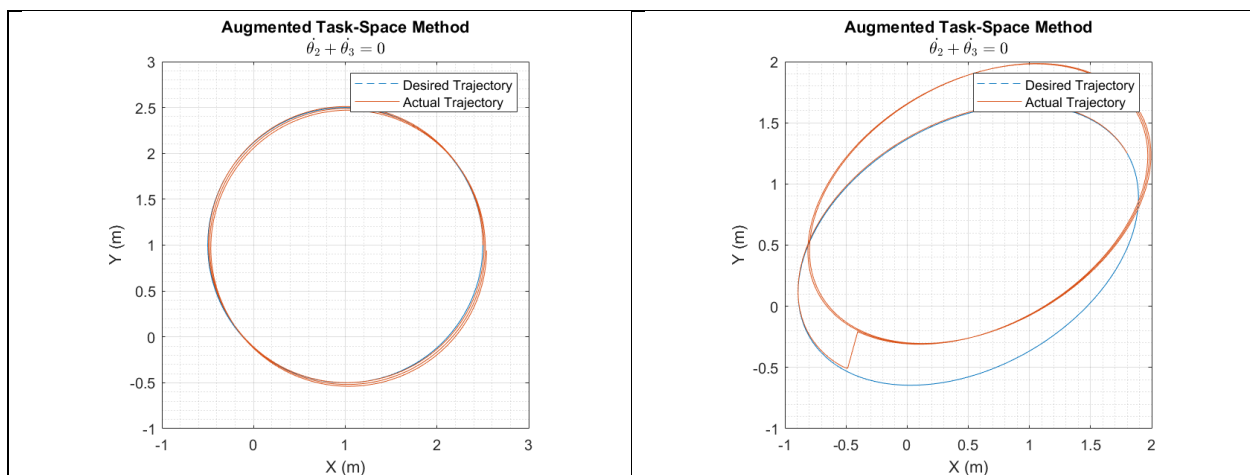
The manipulator was able to track the desired trajectory satisfactorily, but it kept drifting away from the desired trajectory continuously. However, no jittery motion was observed, the simulation went pretty smooth.

MATLAB Code: `P3_1.m`

B: Augmented-Task Space Method

3-B-i Constraint: $\dot{\theta}_2 + \dot{\theta}_3 = 0$

Results:



The manipulator was able to track the desired circular trajectory satisfactorily, but it kept drifting away from the desired trajectory continuously. However, in case of the elliptical trajectory, we observed jittery

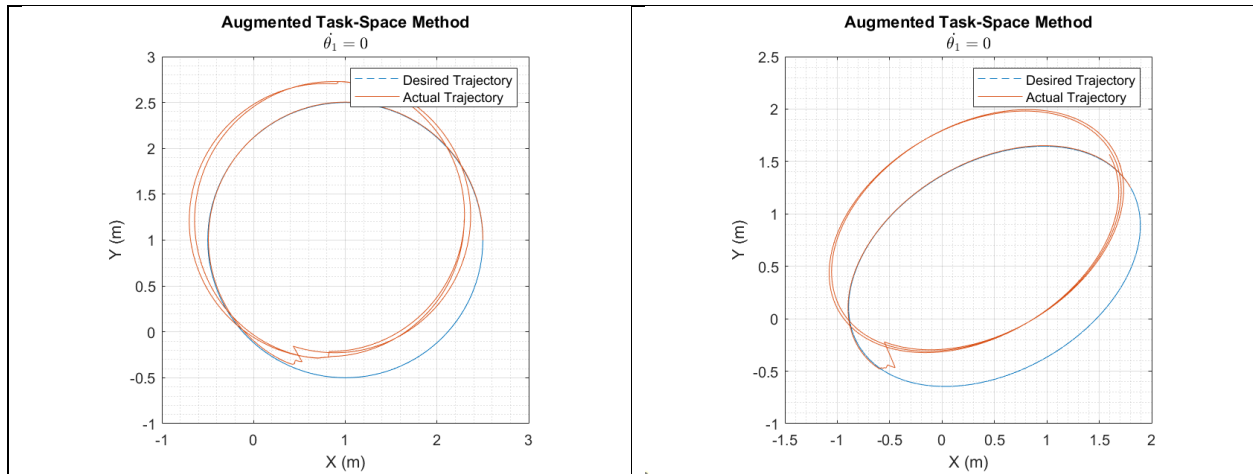
motion at point(s) where the imposed constraint could not be satisfied, which caused a major shift in the actual trajectory of the robot's end-effector.

Note: The point(s) violating the imposed constraints (for both the maneuvers) are dependent on the initial configuration of the robot. As a result, a different initial configuration than the one for which we have simulated for (refer lines 121 and 149 of MATLAB code) may change number and location of such point(s). We chose initial configurations such that the robot could track most of the desired trajectory continuously (luckily, we could find a configuration such that the robot could track the entire desired circular trajectory satisfactorily but couldn't find a similar configuration to track the elliptical trajectory without any jitters).

MATLAB Code: P3_2A.m

3-B-ii Constraint: $\dot{\theta}_1 = 0$

Results:



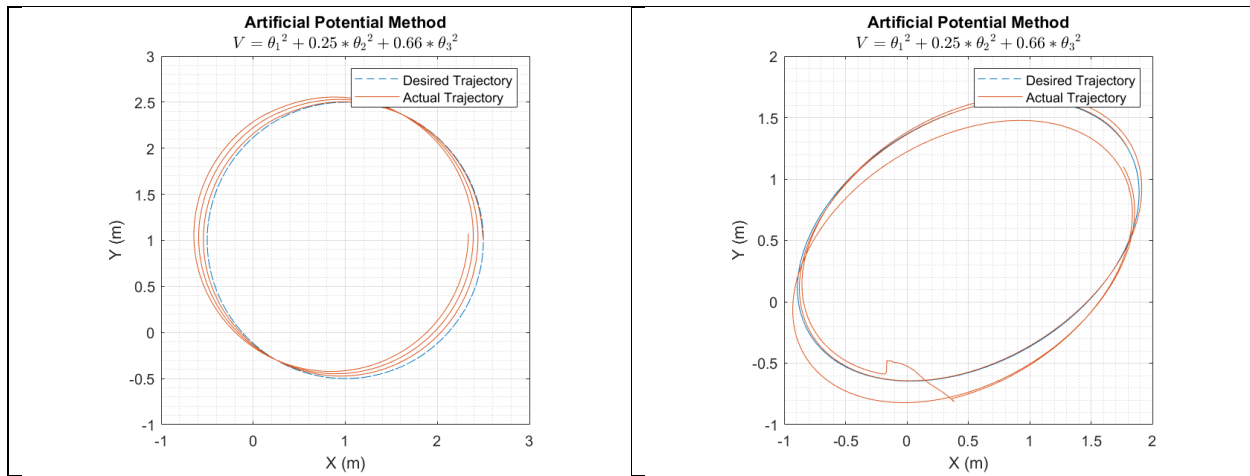
The manipulator faced constraint-related issues while trying to track the desired trajectories satisfactorily. Apart from the drifting behavior, we observed jittery motion at point(s) where the imposed constraint could not be satisfied, which caused a major shift in the actual trajectory of the robot's end-effector.

Note: The point(s) violating the imposed constraints (for both the maneuvers) are dependent on the initial configuration of the robot. As a result, a different initial configuration than the one for which we have simulated for (refer lines 121 and 149 of MATLAB code) may change number and location of such point(s). We chose initial configurations such that the robot could track most of the desired trajectory continuously (we could not find any configuration such that the robot could track the entire desired trajectories without any jitters).

MATLAB Code: P3_2B.m

C: Artificial Potential Method

Results:



The manipulator was able to track the desired circular trajectory satisfactorily, but it kept drifting away from the desired trajectory continuously. However, in case of the elliptical trajectory, after satisfactory tracking for one complete cycle, we observed jittery motion at point(s) where the imposed artificial-potential requirement could not be satisfied, which caused a significant shift in the actual trajectory of the robot's end-effector (during second tracking cycle).

Note: The point(s) violating the imposed artificial-potential requirement (for both the maneuvers) are dependent on the initial configuration of the robot. As a result, a different initial configuration than the one for which we have simulated for (refer lines 121 and 149 of MATLAB code) may change number and location of such point(s). We chose initial configurations such that the robot could track most of the desired trajectory continuously (luckily, we could find a configuration such that the robot could track the entire desired circular trajectory satisfactorily but couldn't find a similar configuration to track the elliptical trajectory without any jitters).

MATLAB Code: P3_3.m