KLE Technological University | Creating Value, Leveraging Knowledge

DEPARTMENT OF MATHEMATICS

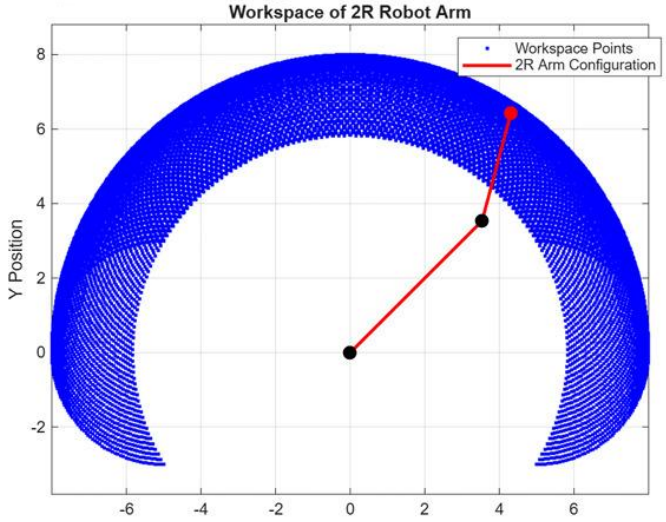
Mathematics Lab Assessment No. 4

Name: Anup Wadekar
Division: S
SRN:01FE24BAR008
Date: 04-12-2025

Question

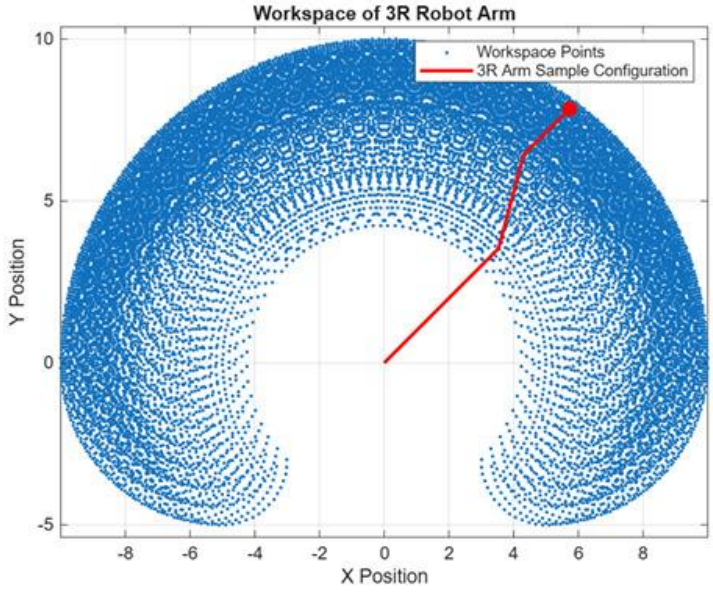
Problem on Robot 2R:

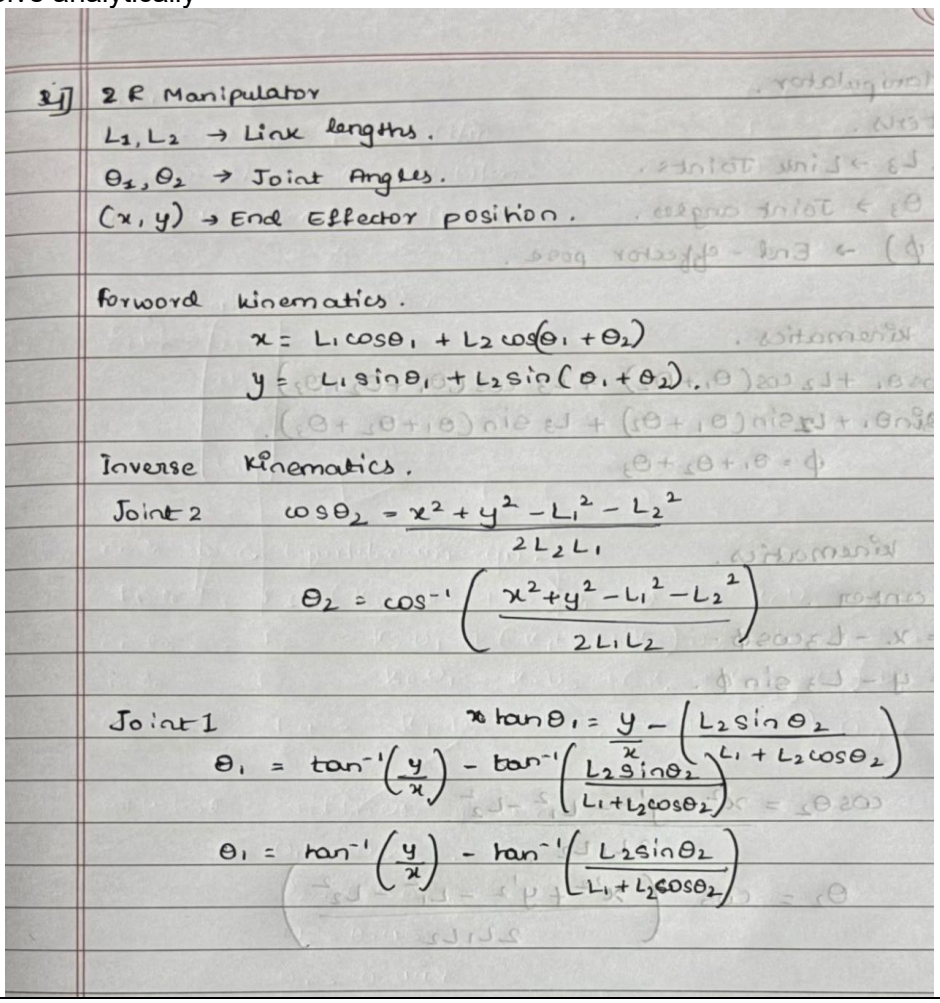
Planar Robot 2R with two link lengths $L_1 = 5\ m$, $L_2 = 3\ m$ and joint angles $\theta_1 = 45°, \theta_2 = 30°$. Use homogeneous transformation matrix to compute the end effector position. Plot and study the work space for the joint limits $\theta_{1min} = 0°, \theta_{1max} = 180°, \theta_{2min} = -90°, \theta_{2max} = 90°$ and 120 sampling points.



Problem on Robot 3R:

Planar Robot 3R with two link lengths $L_1 = 5\ m$, $L_2 = 3\ m$, $L_3 = 2\ m$ and joint angles $\theta_1 = 45°, \theta_2 = 30°, \theta_3 = -30°$. Use homogeneous transformation matrix to compute the end effector position. Plot and study the work space for the joint limits $\theta_{1min} = 0°, \theta_{1max} = 180°, \theta_{2min} = -90°, \theta_{2max} = 90°, \theta_{3min} = -90°, \theta_{3max} = 90°$ and 30 sampling points.



Solution	i) Identify the parameters and mathematical concept **System** • Planar robotic manipulator • Joint type: Revolute joints • **2R → 2 DOF**, **3R → 3 DOF** **Parameters** **2R Manipulator** • l_1, l_2: link lengths • θ_1, θ_2: joint angles • x, y: end-effector position **3R Manipulator** • l_1, l_2, l_3: link lengths • $\theta_1, \theta_2, \theta_3$: joint angles • x, y, ϕ: position and orientation **Mathematical Concept** • Trigonometry • Forward and inverse kinematics
	ii) Solve analytically  2] 2R Manipulator L_1, L_2 → Link lengths. θ_1, θ_2 → Joint Angles. (x, y) → End Effector position. Forword kinematics. $x = L_1\cos\theta_1 + L_2\cos(\theta_1+\theta_2)$ $y = L_1\sin\theta_1 + L_2\sin(\theta_1+\theta_2)$ Inverse Kinematics. Joint 2 $\cos\theta_2 = \frac{x^2+y^2-L_1^2-L_2^2}{2L_2L_1}$ $\theta_2 = \cos^{-1}\left(\frac{x^2+y^2-L_1^2-L_2^2}{2L_1L_2}\right)$ Joint 1 $\tan\theta_1 = \frac{y}{x} - \left(\frac{L_2\sin\theta_2}{L_1+L_2\cos\theta_2}\right)$ $\theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{L_2\sin\theta_2}{L_1+L_2\cos\theta_2}\right)$ $\theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{L_2\sin\theta_2}{L_1+L_2\cos\theta_2}\right)$

3R Manipulator.

Parameters.

L_1, L_2, L_3 → Link Joints.

$\theta_1, \theta_2, \theta_3$ → Joint angles.

(x, y, ϕ) → End-effector pose.

Forward kinematics.

$$x = L_1\cos\theta_1 + L_2\cos(\theta_1+\theta_2) + L_3\cos(\theta_1+\theta_2+\theta_3)$$

$$y = L_1\sin\theta_1 + L_2\sin(\theta_1+\theta_2) + L_3\sin(\theta_1+\theta_2+\theta_3).$$

$$\phi = \theta_1+\theta_2+\theta_3$$

Inverse kinematics.

wrist center.

$$x' = x - L_3\cos\phi.$$

$$y' = y - L_3\sin\phi.$$

Joint 2

$$\cos\theta_2 = \frac{x'^2 + y'^2 - L_1^2 - L_2^2}{2L_1L_2}$$

$$\theta_2 = \cos^{-1}\left(\frac{x'^2 + y'^2 - L_1^2 - L_2^2}{2L_1L_2}\right)$$

Joint 1

$$\theta_1 = \tan^{-1}\left(\frac{y'}{x'}\right) - \tan^{-1}\left(\frac{L_2\sin\theta_2}{L_1 + L_2\cos\theta_2}\right)$$

$$\theta_3 = \phi - \theta_1 - \theta_2$$

iii) GeoGebra Screenshot / Program Execution

3R:

```python
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import scipy.linalg as la

# Link lengths
L1, L2, L3 = 50, 50, 50

# Joint angle limits (radians)
theta1_min, theta1_max = 0, np.pi
theta2_min, theta2_max = -np.pi/2, np.pi/2
theta3_min, theta3_max = -np.pi/2, np.pi/2

# Sampling resolution
num_points = 30

# Generate joint angle vectors for workspace sampling
theta1 = np.linspace(theta1_min, theta1_max, num_points)
theta2 = np.linspace(theta2_min, theta2_max, num_points)
theta3 = np.linspace(theta3_min, theta3_max, num_points)

# Compute workspace points by forward kinematics
workspace_x, workspace_y = [], []
for t1 in theta1:
    for t2 in theta2:
        for t3 in theta3:
            x = L1*np.cos(t1) + L2*np.cos(t1 + t2) + L3*np.cos(t1 + t2 + t3)
            y = L1*np.sin(t1) + L2*np.sin(t1 + t2) + L3*np.sin(t1 + t2 + t3)
            workspace_x.append(x)
            workspace_y.append(y)

# Plot workspace
plt.figure()
plt.plot(workspace_x, workspace_y, '.', markersize=5)
plt.axis('equal')
plt.grid(True)
plt.title('Workspace of 3R Robot Arm')
plt.xlabel('X Position')
plt.ylabel('Y Position')

# Plot sample robot arm configuration
sample_theta = [np.pi/2, np.pi/2, -np.pi/2]
joint1 = np.array([0, 0])
joint2 = joint1 + np.array([L1*np.cos(sample_theta[0]), L1*np.sin(sample_theta[0])])
joint3 = joint2 + np.array([L2*np.cos(sample_theta[0]+sample_theta[1]), L2*np.sin(sample_theta[0]+sample_theta[1])])
end_effector = joint3 + np.array([L3*np.cos(sum(sample_theta)), L3*np.sin(sum(sample_theta))])
```

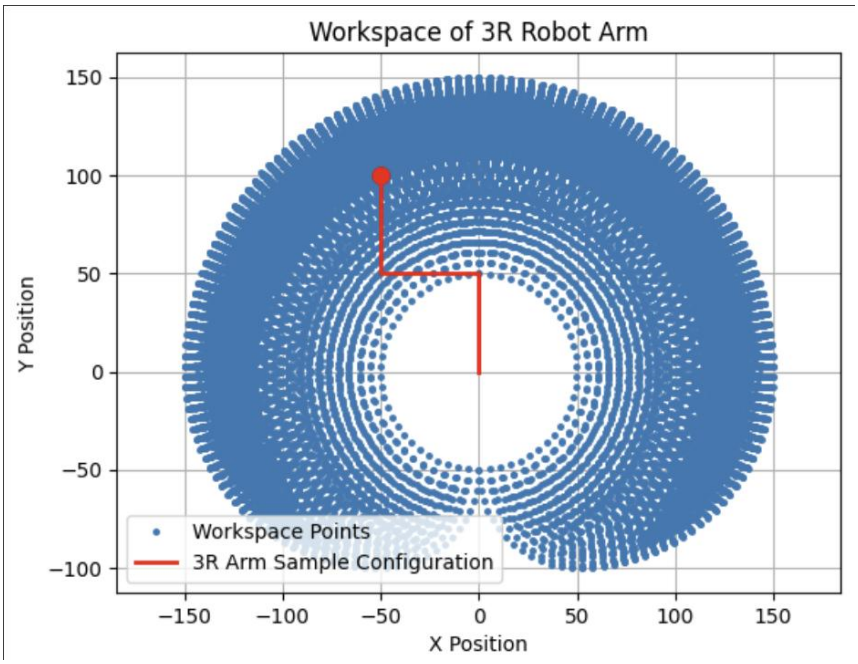
```
plt.plot([joint1[0], joint2[0], joint3[0], end_effector[0]],
         [joint1[1], joint2[1], joint3[1], end_effector[1]], 'r-', linewidth=2)
plt.plot(end_effector[0], end_effector[1], 'ro', markersize=8, markerfacecolor='r')
plt.legend(['Workspace Points', '3R Arm Sample Configuration'])

# Plot Configuration Space (C-Space)
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
Theta1, Theta2, Theta3 = np.meshgrid(theta1, theta2, theta3)
ax.scatter(Theta1.flatten(), Theta2.flatten(), Theta3.flatten(), c='g', marker='.')
ax.grid(True)
ax.set_title('Configuration Space (C-Space) of 3R Robot Arm')
ax.set_xlabel(r'$\theta_1$ (rad)')
ax.set_ylabel(r'$\theta_2$ (rad)')
ax.set_zlabel(r'$\theta_3$ (rad)')
ax.set_xlim([theta1_min, theta1_max])
ax.set_ylim([theta2_min, theta2_max])
ax.set_zlim([theta3_min, theta3_max])

# Jacobian matrix at sample configuration
theta1, theta2, theta3 = sample_theta
J = np.zeros((2, 3))
J[:,0] = [-L1*np.sin(theta1) - L2*np.sin(theta1 + theta2) - L3*np.sin(theta1 + theta2 + theta3),
          L1*np.cos(theta1) + L2*np.cos(theta1 + theta2) + L3*np.cos(theta1 + theta2 + theta3)]
J[:,1] = [-L2*np.sin(theta1 + theta2) - L3*np.sin(theta1 + theta2 + theta3),
          L2*np.cos(theta1 + theta2) + L3*np.cos(theta1 + theta2 + theta3)]
J[:,2] = [-L3*np.sin(theta1 + theta2 + theta3),
          L3*np.cos(theta1 + theta2 + theta3)]

# Compute null space basis vectors of Jacobian
null_vecs = la.null_space(J)

# Plot null space vectors in joint velocity space
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.quiver(0, 0, 0, null_vecs[0,0], null_vecs[1,0], null_vecs[2,0], color='r', linewidth=2)
ax.quiver(0, 0, 0, -null_vecs[0,0], -null_vecs[1,0], -null_vecs[2,0], color='r', linewidth=2)
ax.grid(True)
ax.set_xlabel('Joint 1 velocity')
ax.set_ylabel('Joint 2 velocity')
ax.set_zlabel('Joint 3 velocity')
ax.set_title('Null Space of 3R Robot Arm Jacobian at Sample Configuration')
ax.set_box_aspect([1,1,1])

plt.show()

# Display null space vectors
print('Null space vectors (joint velocities causing no end-effector motion):')
print(null_vecs)
```

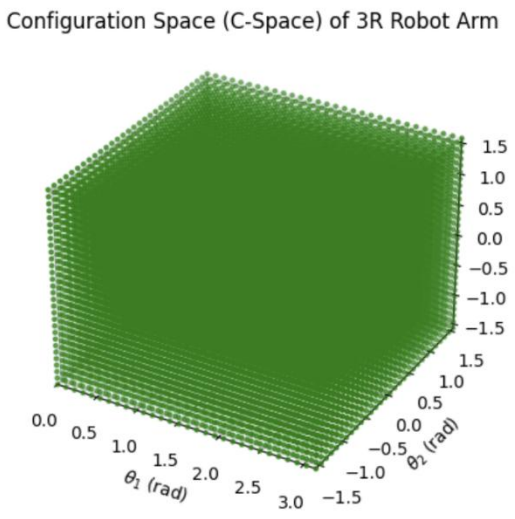
2R:

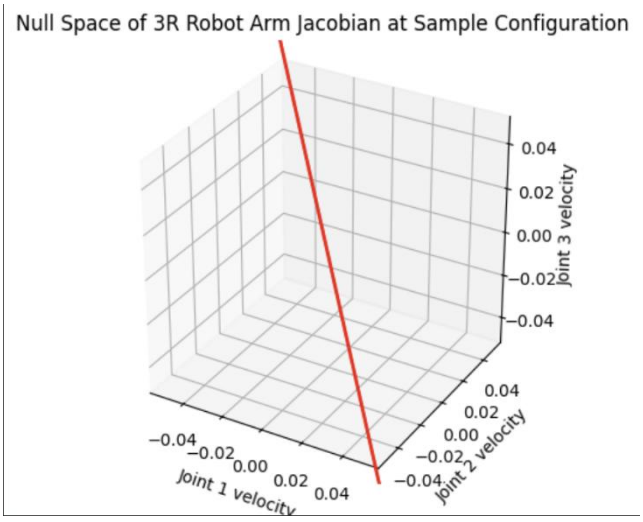
```python
import numpy as np
import matplotlib.pyplot as plt

# Link lengths
L1 = 20
L2 = 25

# Joint angle limits in degrees
theta1_min, theta1_max = 0, 360
theta2_min, theta2_max = -180, 180

# Convert to radians
theta1_vals = np.radians(np.linspace(theta1_min, theta1_max, 120))
theta2_vals = np.radians(np.linspace(theta2_min, theta2_max, 120))

# Lists for storing workspace points
workspace_x = []
workspace_y = []

for t1 in theta1_vals:
    for t2 in theta2_vals:
        # Forward Kinematics using transformation matrices
        x = L1*np.cos(t1) + L2*np.cos(t1 + t2)
        y = L1*np.sin(t1) + L2*np.sin(t1 + t2)

        workspace_x.append(x)
        workspace_y.append(y)

# Example Configuration: θ1 = 45°, θ2 = 30°
t1_sample = np.radians(45)
t2_sample = np.radians(30)

x1 = L1*np.cos(t1_sample)
y1 = L1*np.sin(t1_sample)

x2 = x1 + L2*np.cos(t1_sample + t2_sample)
y2 = y1 + L2*np.sin(t1_sample + t2_sample)

# Plotting
plt.figure(figsize=(7, 7))
plt.scatter(workspace_x, workspace_y, s=3, color='blue', label='Workspace Points')

# Plot robot arm configuration
plt.plot([0, x1, x2], [0, y1, y2], '-o', color='red', linewidth=2, markersize=8, label='2R Arm Configuration')

plt.title("Workspace of 2R Robot Arm")
plt.xlabel("X Position")
plt.ylabel("Y Position")
plt.grid(True)
plt.axis('equal')
plt.legend()
plt.show()
```

iv) Results and analysis from the graph

3R:

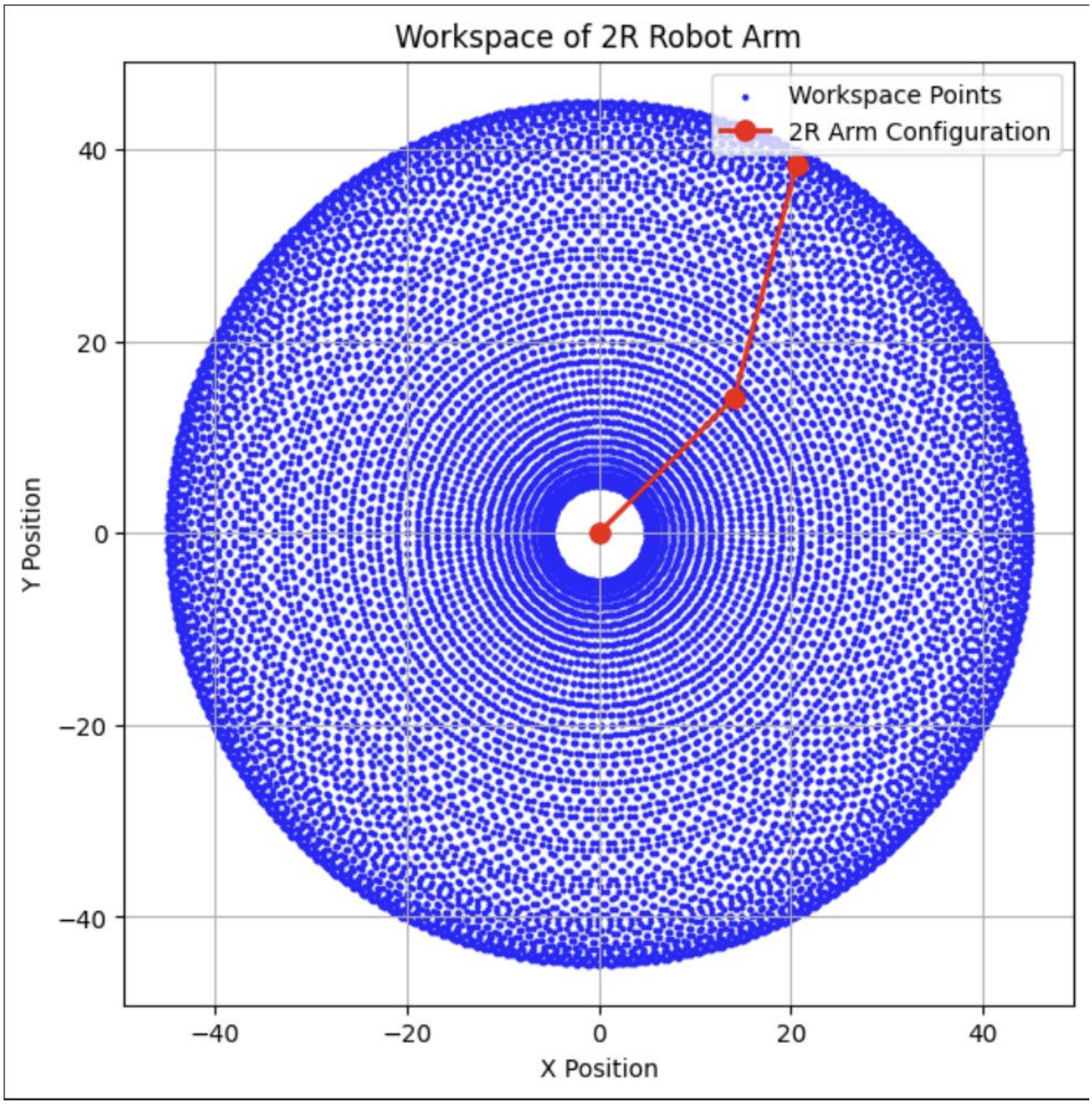







```
Null space vectors (joint velocities causing no end-effector motion):
[[-0.57735027]
 [ 0.57735027]
 [ 0.57735027]]
```

2R:


Workspace of 2R Robot Arm
Workspace Points
2R Arm Configuration
Y Position
X Position
40
20
0
−20
−40
−40
−20
0
20
40