Assignment 2

Linear Joint-Space Trajectory

q_{1s}, q_{2s} = start angles

q_{1e}, q_{2e} = end angles

T = total motion time

$$q_1(t) = q_{1s} + \frac{t}{T}(q_{1e} - q_{1s})$$
$$q_2(t) = q_{2s} + \frac{t}{T}(q_{2e} - q_{2s})$$

Joint Velocities

$$\dot{q}_1(t) = \frac{q_{1e} - q_{1s}}{T}$$
$$\dot{q}_2(t) = \frac{q_{2e} - q_{2s}}{T}$$

This velocity is constant, meaning:

Velocity jumps instantly from 0 to a finite value at t=0

Velocity drops instantly to 0 at t=T

This implies infinite acceleration at start and end, it is not smooth

Smooth Polynomial Joint-Space Trajectory(Cubic)

$$q_i(t) = a_{0i} + a_{1i}t + a_{2i}t^2 + a_{3i}t^3$$

Boundary Conditions:

$$q_i(0) = q_{i,s}$$
$$q_i(T) = q_{i,e}$$
$$\dot{q}_i(0) = 0$$
$$\dot{q}_i(T) = 0$$

Solving for Coefficients:

$$a_{0i} = q_{i,s}$$
$$a_{1i} = 0$$
$$a_{2i} = \frac{3(q_{i,e} - q_{i,s})}{T^2}$$
$$a_{3i} = -\frac{2(q_{i,e} - q_{i,s})}{T^3}$$

Thus,

$$q_i(t) = q_{i,s} + 3\frac{(q_{i,e} - q_{i,s})}{T^2}t^2 - 2\frac{(q_{i,e} - q_{i,s})}{T^3}t^3$$

Velocity:

$$\dot{q}_i(t) = 6\frac{(q_{i,e} - q_{i,s})}{T^2}t - 6\frac{(q_{i,e} - q_{i,s})}{T^3}t^2$$

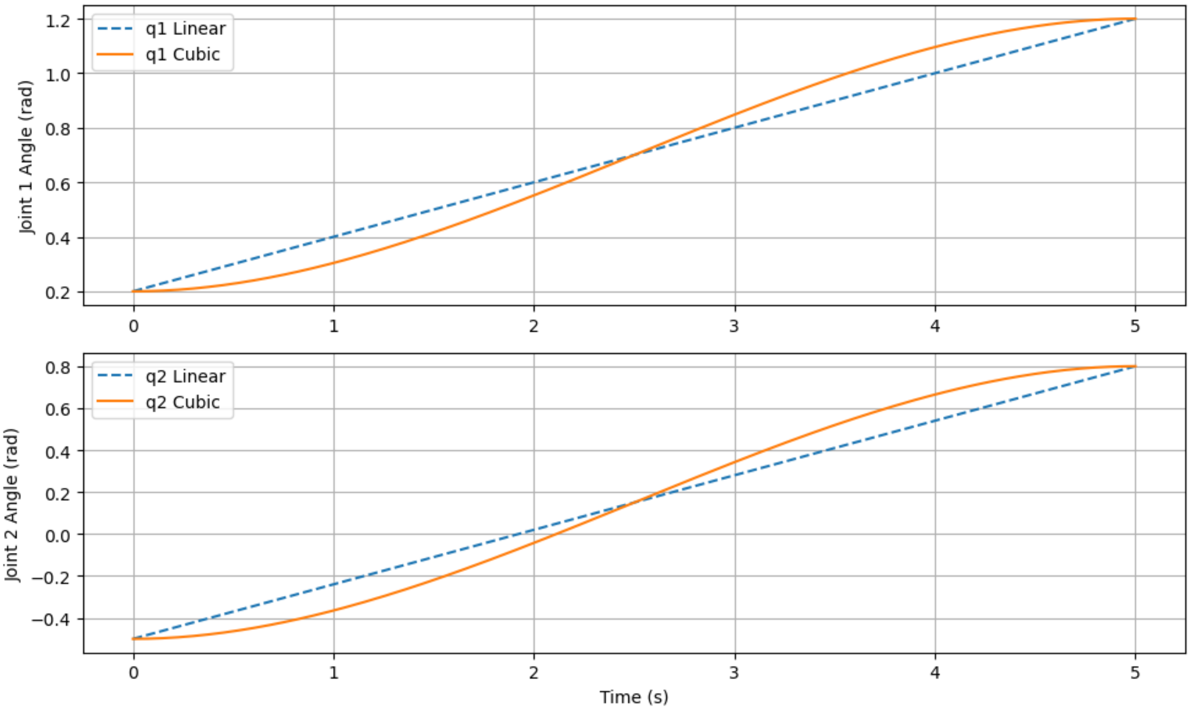
Velocity starts and ends at 0, that is smooth acceleration and deceleration

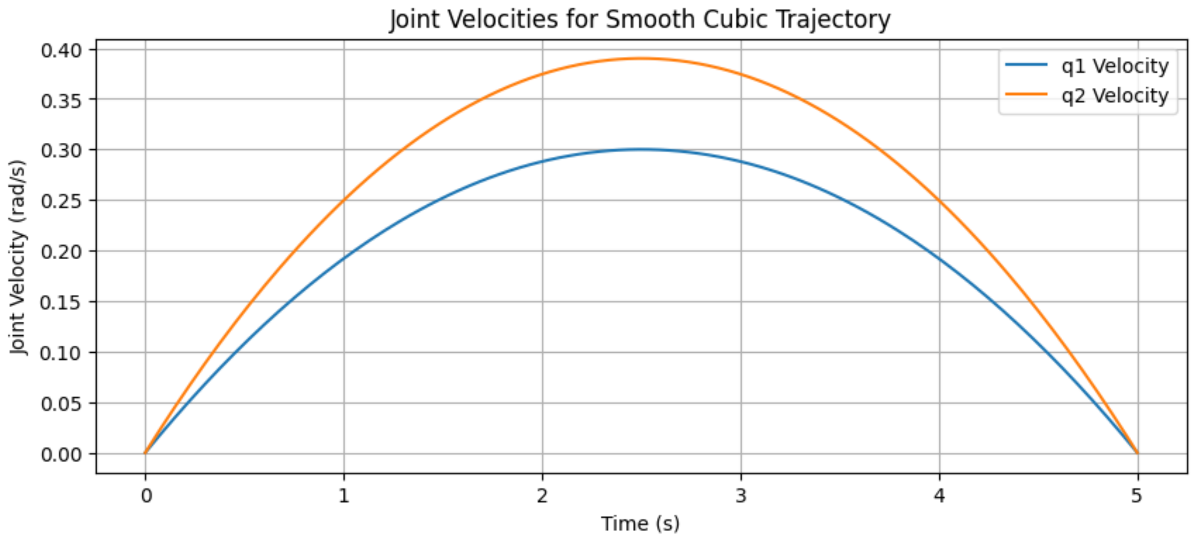
Python Script

```
import numpy as np

import matplotlib.pyplot as plt


T = 5.0

t = np.linspace(0, T, 200)

q1_start, q1_end = 0.2, 1.2

q2_start, q2_end = -0.5, 0.8


q1_linear = q1_start + (q1_end - q1_start) * (t / T)
```

```
q2_linear = q2_start + (q2_end - q2_start) * (t / T)


def cubic_trajectory(q_start, q_end, T, t):
    a0 = q_start
    a1 = 0
    a2 = 3 * (q_end - q_start) / T**2
    a3 = -2 * (q_end - q_start) / T**3
    q = a0 + a1*t + a2*t**2 + a3*t**3
    q_dot = a1 + 2*a2*t + 3*a3*t**2
    return q, q_dot


q1_cubic, q1_dot = cubic_trajectory(q1_start, q1_end, T, t)
q2_cubic, q2_dot = cubic_trajectory(q2_start, q2_end, T, t)


plt.figure(figsize=(10, 6))


plt.subplot(2, 1, 1)
plt.plot(t, q1_linear, '--', label='q1 Linear')
plt.plot(t, q1_cubic, label='q1 Cubic')
plt.ylabel('Joint 1 Angle (rad)')
plt.legend()
plt.grid()


plt.subplot(2, 1, 2)
```

```
plt.plot(t, q2_linear, '--', label='q2 Linear')
plt.plot(t, q2_cubic, label='q2 Cubic')
plt.xlabel('Time (s)')
plt.ylabel('Joint 2 Angle (rad)')
plt.legend()
plt.grid()

plt.tight_layout()
plt.show()

plt.figure(figsize=(10, 4))
plt.plot(t, q1_dot, label='q1 Velocity')
plt.plot(t, q2_dot, label='q2 Velocity')
plt.xlabel('Time (s)')
plt.ylabel('Joint Velocity (rad/s)')
plt.title('Joint Velocities for Smooth Cubic Trajectory')
plt.legend()
plt.grid()
plt.show()
```





Short Discussion

In a linear joint-space trajectory, joint angles vary uniformly with time, resulting in sudden changes in velocity at the start and end of motion. These abrupt velocity changes can introduce high accelerations and mechanical stress on robot joints. In contrast, the smooth cubic polynomial trajectory ensures zero velocity at both the beginning and end, leading to a gradual acceleration and deceleration. This produces smoother joint motion and reduces wear on actuators. Smooth trajectories are more suitable for real robotic systems because they improve motion stability, reduce vibrations, and enhance tracking performance. Therefore, polynomial trajectories are preferred in practical robotic applications where safety and precision are critical.