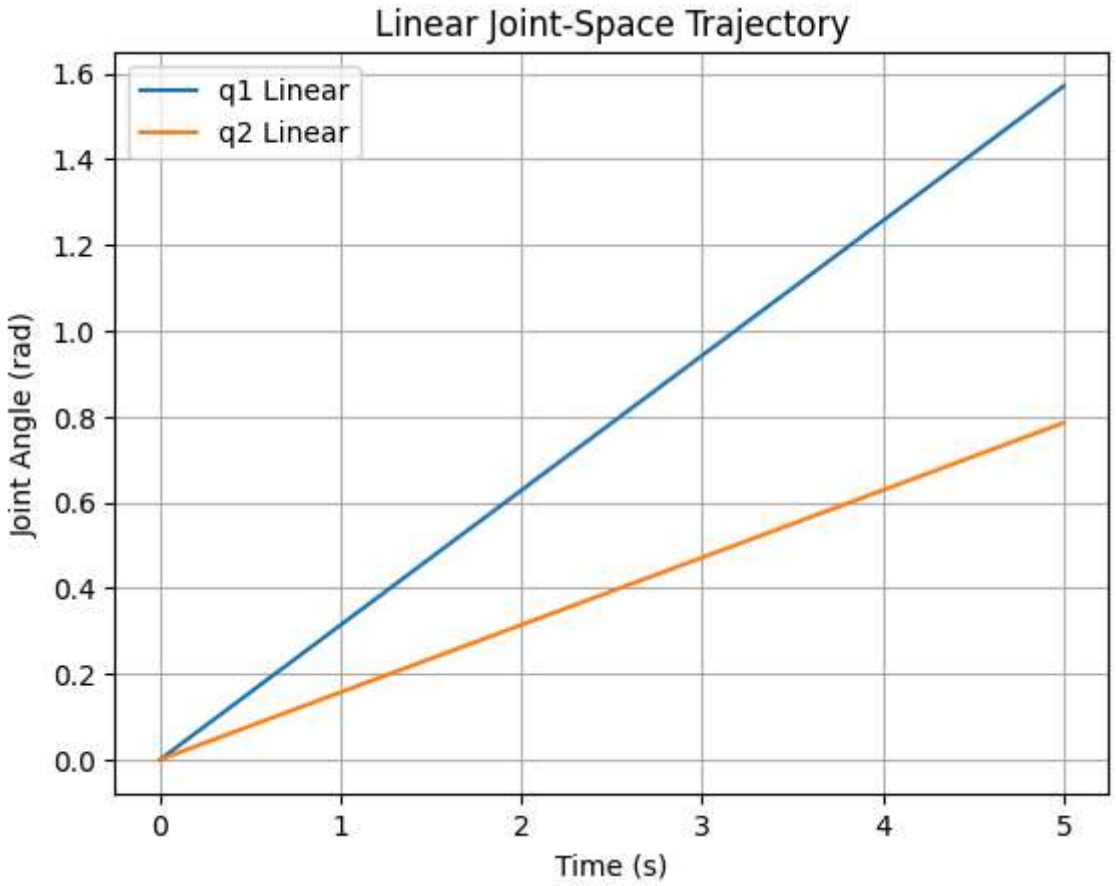
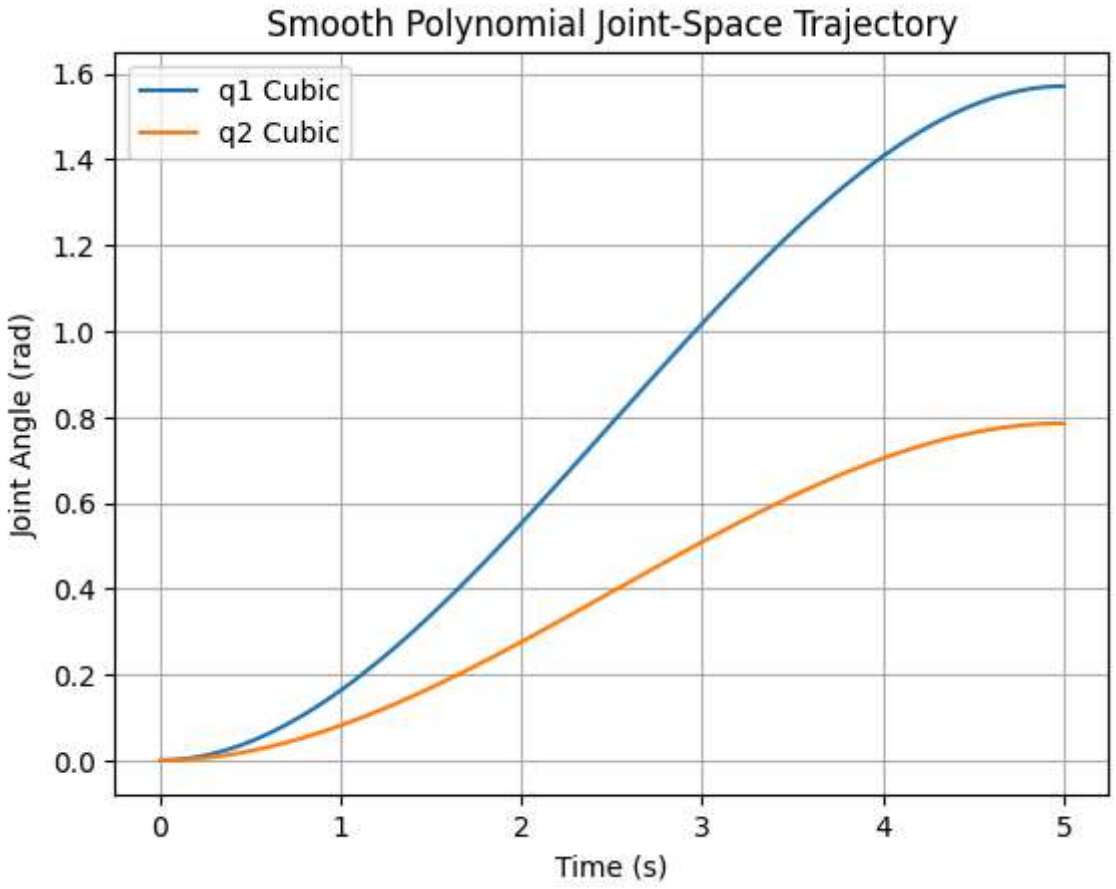
In [ ]:
```python
import numpy as np
import matplotlib.pyplot as plt
q1_start = 0
q2_start = 0

q1_end = np.pi/2
q2_end = np.pi/4

T = 5   # total time (seconds)
t = np.linspace(0, T, 100)
```

In [ ]:
```python
def linear_trajectory(q_start,q_end,T,t):
    return q_start + (q_end - q_start) * (t / T)
q1_linear= linear_trajectory(q1_start,q1_end,T,t)
q2_linear= linear_trajectory(q2_start,q2_end,T,t)
# Plot
plt.figure()
plt.plot(t, q1_linear, label='q1 Linear')
plt.plot(t, q2_linear, label='q2 Linear')
plt.xlabel('Time (s)')
plt.ylabel('Joint Angle (rad)')
plt.title('Linear Joint-Space Trajectory')
plt.legend()
plt.grid()
plt.show()
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)

    return a0 + a1*t + a2*t**2 + a3*t**3
q1_cubic = cubic_trajectory(q1_start, q1_end, T, t)
q2_cubic = cubic_trajectory(q2_start, q2_end, T, t)

# Plot
plt.figure()
plt.plot(t, q1_cubic, label='q1 Cubic')
plt.plot(t, q2_cubic, label='q2 Cubic')
plt.xlabel('Time (s)')
```

```
plt.ylabel('Joint Angle (rad)')
plt.title('Smooth Polynomial Joint-Space Trajectory')
plt.legend()
plt.grid()
plt.show()
```

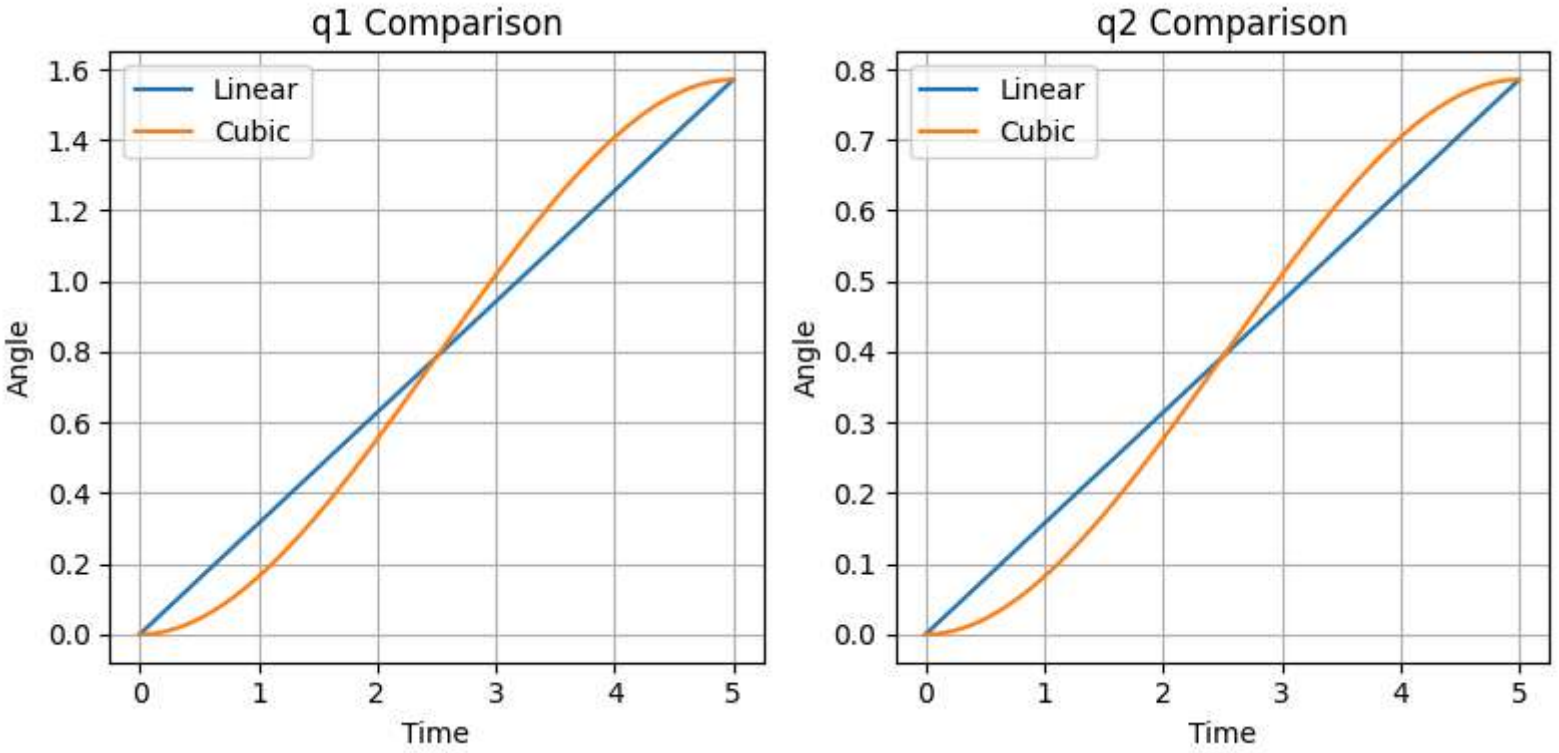




In [ ]:

In [8]:
```
plt.figure(figsize=(8,4))

plt.subplot(1,2,1)
plt.plot(t, q1_linear, label='Linear')
plt.plot(t, q1_cubic, label='Cubic')
plt.title('q1 Comparison')
plt.xlabel('Time')
plt.ylabel('Angle')
plt.legend()
plt.grid()

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

```
plt.plot(t, q2_linear, label='Linear')
plt.plot(t, q2_cubic, label='Cubic')
plt.title('q2 Comparison')
plt.xlabel('Time')
plt.ylabel('Angle')
plt.legend()
plt.grid()

plt.tight_layout()
plt.show()
```



Short Discussion

In the linear joint-space trajectory, the joint angles change at a constant rate from the start to the end position. Although this method is simple, the motion starts and stops suddenly, which can be seen near the beginning and end of the plot. In the smooth polynomial trajectory, the joint angles change more gradually, and the motion starts and ends smoothly. This happens because the velocity is zero at the start and end of the motion. Smooth trajectories are better for real robots as they reduce sudden jerks and mechanical stress. Linear trajectories are easier to implement but are not ideal for physical systems. Therefore, smooth polynomial trajectories are more suitable for practical robotic applications.