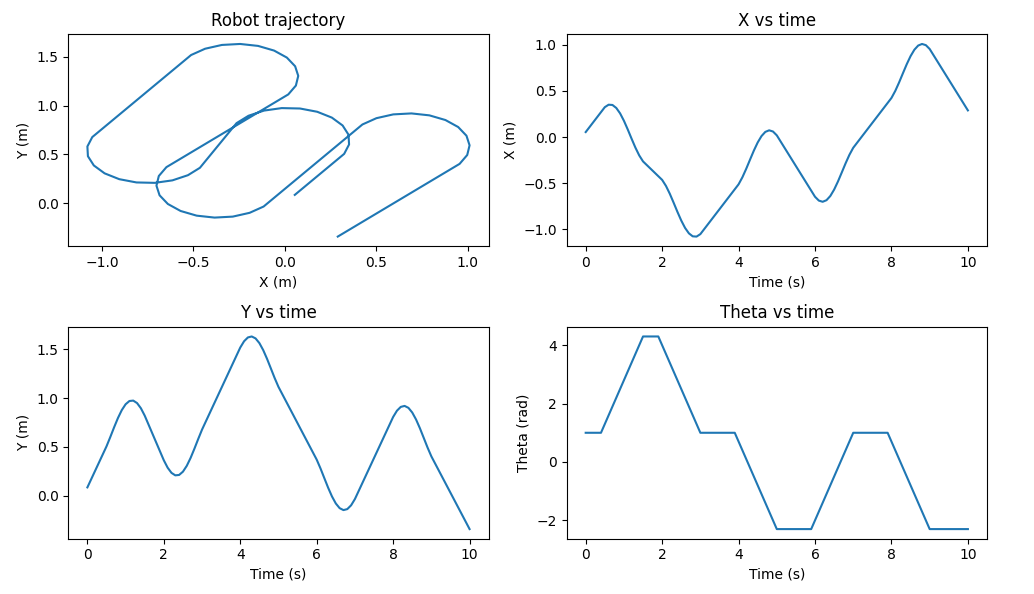
**Answer to the Question no. – 2(a, b)**

**Google Colab Code:** Source code file has also attached(Name: *HomeWork1\_Question\_2(a,\_b).ipynb*)

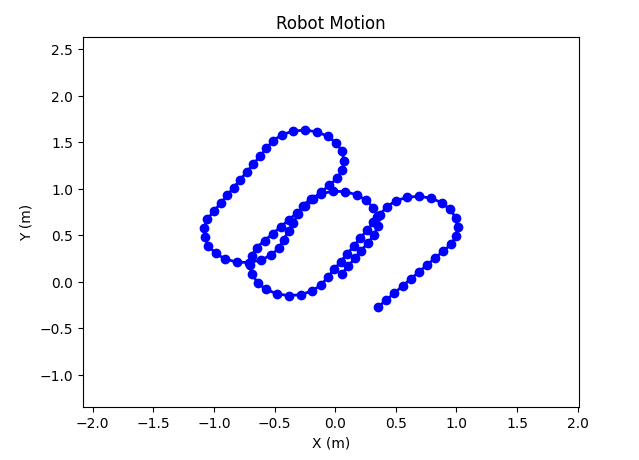
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101 | from google.colab import drive  # Mount Google Drive  drive.mount('/content/drive')  import numpy as np  import matplotlib.pyplot as plt  from matplotlib.animation import FuncAnimation  # Define control inputs  def control\_input(t):  v = np.where((t >= 0) & (t <= 10), 1, 0)  w = np.where(((t >= 0.5) & (t <= 1.5)) | ((t >= 6) & (t <= 7)), 3, 0)  w = np.where(((t >= 2) & (t <= 3)) | ((t >= 4) & (t <= 5)) | ((t >= 8) & (t <= 9)), -3, w)  return v, w  # Define differential equations  def f(x, y, theta, v, w):  x\_dot = v \* np.cos(theta)  y\_dot = v \* np.sin(theta)  theta\_dot = w  return x\_dot, y\_dot, theta\_dot  # Simulation parameters  dt = 0.1  T = 10  # Initial conditions  x = 0  y = 0  theta = 1  # Time array  t\_data = np.arange(0, T + dt, dt)  # Control inputs  v, w = control\_input(t\_data)  # Initialize data storage  x\_data = np.zeros(len(t\_data))  y\_data = np.zeros(len(t\_data))  theta\_data = np.zeros(len(t\_data))  # Euler's method loop  for i in range(len(t\_data)):  x\_dot, y\_dot, theta\_dot = f(x, y, theta, v[i], w[i])  x += x\_dot \* dt  y += y\_dot \* dt  theta += theta\_dot \* dt  x\_data[i] = x  y\_data[i] = y  theta\_data[i] = theta  # Plot the results  plt.figure(figsize=(10, 6))  plt.subplot(2, 2, 1)  plt.plot(x\_data, y\_data)  plt.xlabel('X (m)')  plt.ylabel('Y (m)')  plt.title('Robot trajectory')  plt.subplot(2, 2, 2)  plt.plot(t\_data, x\_data)  plt.xlabel('Time (s)')  plt.ylabel('X (m)')  plt.title('X vs time')  plt.subplot(2, 2, 3)  plt.plot(t\_data, y\_data)  plt.xlabel('Time (s)')  plt.ylabel('Y (m)')  plt.title('Y vs time')  plt.subplot(2, 2, 4)  plt.plot(t\_data, theta\_data)  plt.xlabel('Time (s)')  plt.ylabel('Theta (rad)')  plt.title('Theta vs time')  plt.tight\_layout()  plt.show()  # Animation  fig, ax = plt.subplots()  line, = ax.plot([], [], 'bo-', lw=2)  def init():  ax.set\_xlim(min(x\_data) - 1, max(x\_data) + 1)  ax.set\_ylim(min(y\_data) - 1, max(y\_data) + 1)  ax.set\_xlabel('X (m)')  ax.set\_ylabel('Y (m)')  ax.set\_title('Robot Motion')  return line,  def update(frame):  line.set\_data(x\_data[:frame], y\_data[:frame])  return line,  ani = FuncAnimation(fig, update, frames=len(t\_data), init\_func=init, blit=True)  ani.save('/content/drive/My Drive/robot\_motion.mp4', fps=10)  plt.show() |

**Output:**

**2(a):**

****

**2(b):** A mp4 file has also attached *(*Name*: HomeWork1 Question 2(b).mp4)*

****

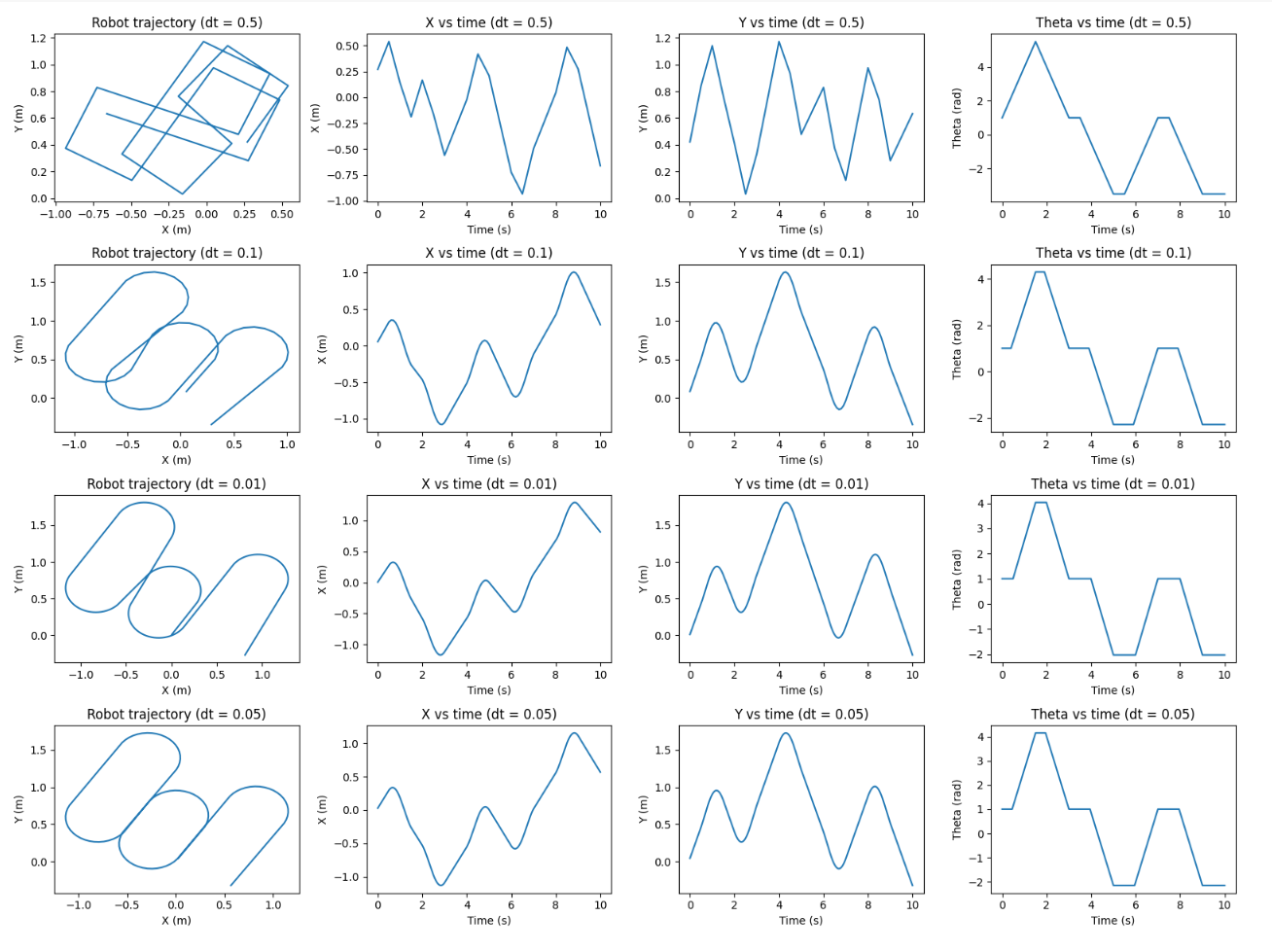
**Updated version:**

∆t = 0.5, 0.1, 0.01, 0.05.

**Google Colab Code:** Source code file has also attached(Name: *HomeWork1\_Question\_2(delta\_t\_updated).ipynb*)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85 | # If time step ∆t = 0.5, 0.1, 0.01, 0.05.  import numpy as np  import matplotlib.pyplot as plt  # Define control inputs  def control\_input(t):  v = np.where((t >= 0) & (t <= 10), 1, 0)  w = np.where(((t >= 0.5) & (t <= 1.5)) | ((t >= 6) & (t <= 7)), 3, 0)  w = np.where(((t >= 2) & (t <= 3)) | ((t >= 4) & (t <= 5)) | ((t >= 8) & (t <= 9)), -3, w)  return v, w  # Define differential equations  def f(x, y, theta, v, w):  x\_dot = v \* np.cos(theta)  y\_dot = v \* np.sin(theta)  theta\_dot = w  return x\_dot, y\_dot, theta\_dot  # Simulation parameters  dt\_values = [0.5, 0.1, 0.01, 0.05]  T = 10  # Initial conditions  x0 = 0  y0 = 0  theta0 = 1  # Plotting setup  plt.figure(figsize=(16, 12))  plot\_idx = 1  for dt in dt\_values:  # Time array  t\_data = np.arange(0, T + dt, dt)  # Control inputs  v, w = control\_input(t\_data)  # Initialize data storage  x\_data = np.zeros(len(t\_data))  y\_data = np.zeros(len(t\_data))  theta\_data = np.zeros(len(t\_data))  # Euler's method loop  x = x0  y = y0  theta = theta0  for i in range(len(t\_data)):  x\_dot, y\_dot, theta\_dot = f(x, y, theta, v[i], w[i])  x += x\_dot \* dt  y += y\_dot \* dt  theta += theta\_dot \* dt  x\_data[i] = x  y\_data[i] = y  theta\_data[i] = theta  # Plotting  plt.subplot(4, 4, plot\_idx)  plt.plot(x\_data, y\_data)  plt.xlabel('X (m)')  plt.ylabel('Y (m)')  plt.title(f'Robot trajectory (dt = {dt})')  plt.subplot(4, 4, plot\_idx + 1)  plt.plot(t\_data, x\_data)  plt.xlabel('Time (s)')  plt.ylabel('X (m)')  plt.title(f'X vs time (dt = {dt})')  plt.subplot(4, 4, plot\_idx + 2)  plt.plot(t\_data, y\_data)  plt.xlabel('Time (s)')  plt.ylabel('Y (m)')  plt.title(f'Y vs time (dt = {dt})')  plt.subplot(4, 4, plot\_idx + 3)  plt.plot(t\_data, theta\_data)  plt.xlabel('Time (s)')  plt.ylabel('Theta (rad)')  plt.title(f'Theta vs time (dt = {dt})')  plot\_idx += 4  plt.tight\_layout()  plt.show() |

**Output:**

****

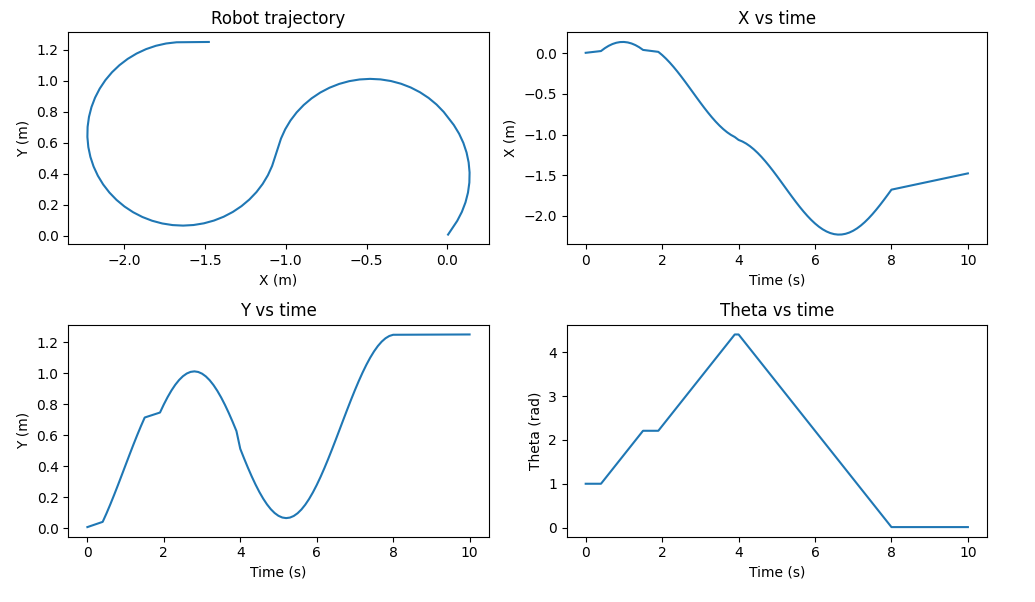
**Answer to the Question no. – 3(a, b)**

**Google Colab Code:** Source code file has also attached(Name: *HomeWork1\_Question\_3(a,\_b).ipynb*)

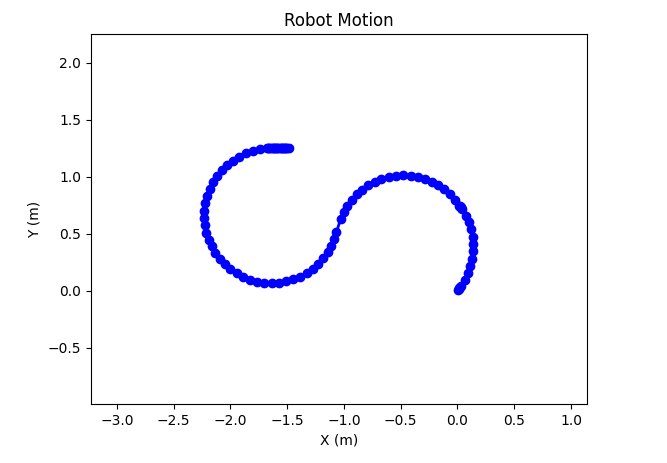
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106 | from google.colab import drive  # Mount Google Drive  drive.mount('/content/drive')  import numpy as np  import matplotlib.pyplot as plt  from matplotlib.animation import FuncAnimation  # Define control inputs  def control\_input(t):  wl = np.where((t >= 4) & (t <= 6), 12, np.where((t >= 6) & (t <= 8), 12, 1))  wr = np.where(((t >= 0.5) & (t <= 1.5)) | ((t >= 2) & (t <= 4)), 12, 1)  return wl, wr  # Define differential equations  def f(x, y, theta, wl, wr):  r = 0.1  L = 1  v = (wl + wr) \* r / 2  w = (wr - wl) \* r / L  x\_dot = v \* np.cos(theta)  y\_dot = v \* np.sin(theta)  theta\_dot = w  return x\_dot, y\_dot, theta\_dot  # Simulation parameters  dt = 0.1  T = 10  # Initial conditions  x = 0  y = 0  theta = 1  # Time array  t\_data = np.arange(0, T + dt, dt)  # Control inputs  wl, wr = control\_input(t\_data)  # Initialize data storage  x\_data = np.zeros(len(t\_data))  y\_data = np.zeros(len(t\_data))  theta\_data = np.zeros(len(t\_data))  # Euler's method loop  for i in range(len(t\_data)):  x\_dot, y\_dot, theta\_dot = f(x, y, theta, wl[i], wr[i])  x += x\_dot \* dt  y += y\_dot \* dt  theta += theta\_dot \* dt  x\_data[i] = x  y\_data[i] = y  theta\_data[i] = theta  # Plot the results  plt.figure(figsize=(10, 6))  plt.subplot(2, 2, 1)  plt.plot(x\_data, y\_data)  plt.xlabel('X (m)')  plt.ylabel('Y (m)')  plt.title('Robot trajectory')  plt.subplot(2, 2, 2)  plt.plot(t\_data, x\_data)  plt.xlabel('Time (s)')  plt.ylabel('X (m)')  plt.title('X vs time')  plt.subplot(2, 2, 3)  plt.plot(t\_data, y\_data)  plt.xlabel('Time (s)')  plt.ylabel('Y (m)')  plt.title('Y vs time')  plt.subplot(2, 2, 4)  plt.plot(t\_data, theta\_data)  plt.xlabel('Time (s)')  plt.ylabel('Theta (rad)')  plt.title('Theta vs time')  plt.tight\_layout()  plt.show()  # Animation  fig, ax = plt.subplots()  line, = ax.plot([], [], 'bo-', lw=2)  def init():  ax.set\_xlim(min(x\_data) - 1, max(x\_data) + 1)  ax.set\_ylim(min(y\_data) - 1, max(y\_data) + 1)  ax.set\_xlabel('X (m)')  ax.set\_ylabel('Y (m)')  ax.set\_title('Robot Motion')  return line,  def update(frame):  line.set\_data(x\_data[:frame], y\_data[:frame])  return line,  ani = FuncAnimation(fig, update, frames=len(t\_data), init\_func=init, blit=True)  ani.save('/content/drive/My Drive/robot\_trajectory.mp4', fps=10)  plt.show() |

**Output:**

**3(a):**

****

**3(b):** A mp4 file has also attached *(*Name*: HomeWork1 Question 3(b).mp4)*

****

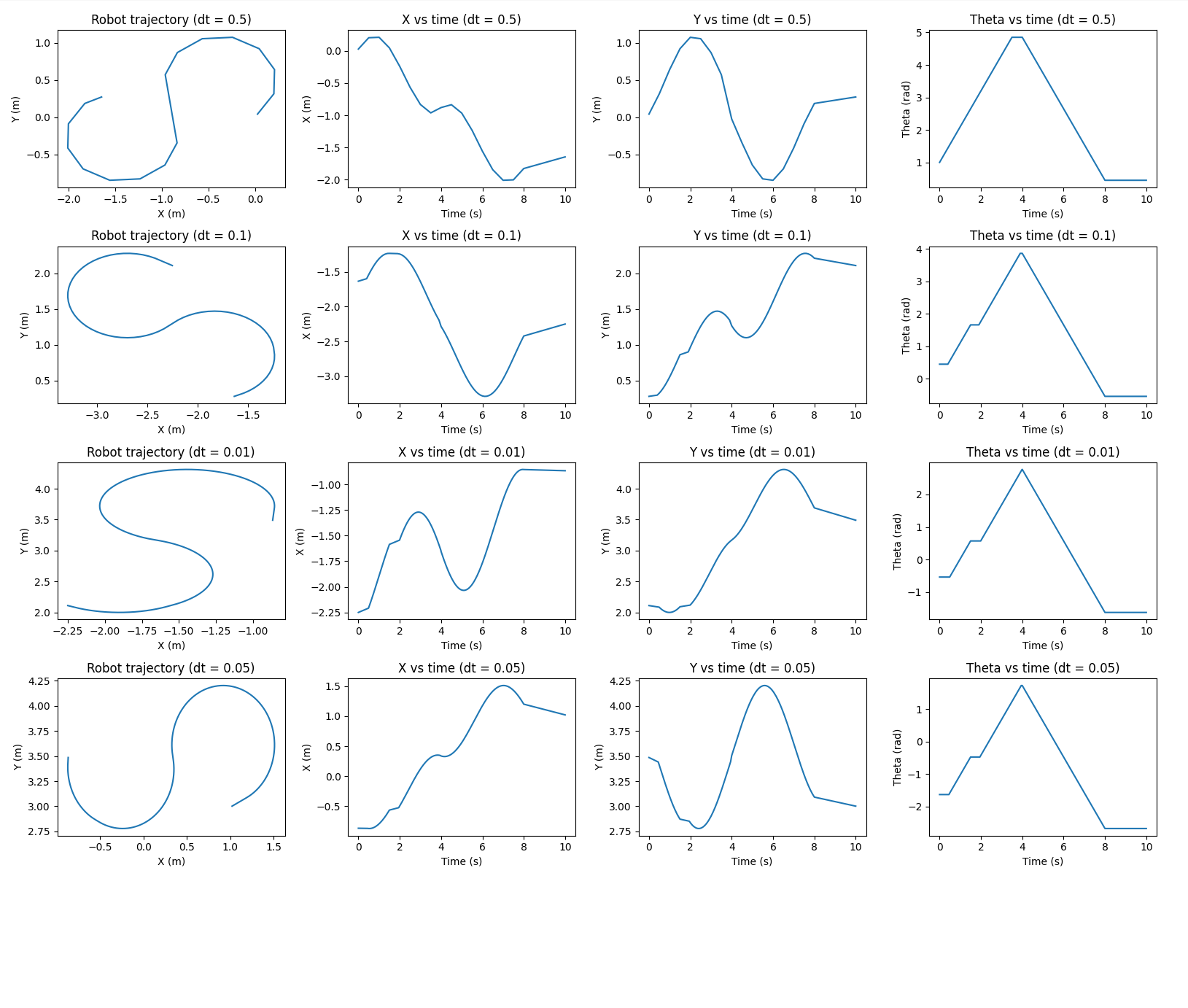
**Updated version:**

∆t = 0.5, 0.1, 0.01, 0.05.

**Google Colab Code:** Source code file has also attached(Name: *HomeWork1\_Question\_3(delta\_t\_updated).ipynb*)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88 | # If time step ∆t = 0.5, 0.1, 0.01, 0.05.  import numpy as np  import matplotlib.pyplot as plt  # Define control inputs  def control\_input(t):  wl = np.where((t >= 4) & (t <= 6), 12, np.where((t >= 6) & (t <= 8), 12, 1))  wr = np.where(((t >= 0.5) & (t <= 1.5)) | ((t >= 2) & (t <= 4)), 12, 1)  return wl, wr  # Define differential equations  def f(x, y, theta, wl, wr):  r = 0.1  L = 1  v = (wl + wr) \* r / 2  w = (wr - wl) \* r / L  x\_dot = v \* np.cos(theta)  y\_dot = v \* np.sin(theta)  theta\_dot = w  return x\_dot, y\_dot, theta\_dot  # Simulation parameters  dt\_values = [0.5, 0.1, 0.01, 0.05]  T = 10  # Initial conditions  x = 0  y = 0  theta = 1  # Plotting setup  plt.figure(figsize=(16, 12))  plot\_idx = 1  for dt in dt\_values:  # Time array  t\_data = np.arange(0, T + dt, dt)  # Control inputs  wl, wr = control\_input(t\_data)  # Initialize data storage  x\_data = np.zeros(len(t\_data))  y\_data = np.zeros(len(t\_data))  theta\_data = np.zeros(len(t\_data))  # Euler's method loop  for i in range(len(t\_data)):  x\_dot, y\_dot, theta\_dot = f(x, y, theta, wl[i], wr[i])  x += x\_dot \* dt  y += y\_dot \* dt  theta += theta\_dot \* dt  x\_data[i] = x  y\_data[i] = y  theta\_data[i] = theta  # Plotting  plt.subplot(4, 4, plot\_idx)  plt.plot(x\_data, y\_data)  plt.xlabel('X (m)')  plt.ylabel('Y (m)')  plt.title(f'Robot trajectory (dt = {dt})')  plt.subplot(4, 4, plot\_idx + 1)  plt.plot(t\_data, x\_data)  plt.xlabel('Time (s)')  plt.ylabel('X (m)')  plt.title(f'X vs time (dt = {dt})')  plt.subplot(4, 4, plot\_idx + 2)  plt.plot(t\_data, y\_data)  plt.xlabel('Time (s)')  plt.ylabel('Y (m)')  plt.title(f'Y vs time (dt = {dt})')  plt.subplot(4, 4, plot\_idx + 3)  plt.plot(t\_data, theta\_data)  plt.xlabel('Time (s)')  plt.ylabel('Theta (rad)')  plt.title(f'Theta vs time (dt = {dt})')  plot\_idx += 4  plt.tight\_layout()  plt.show() |

**Output:**

****