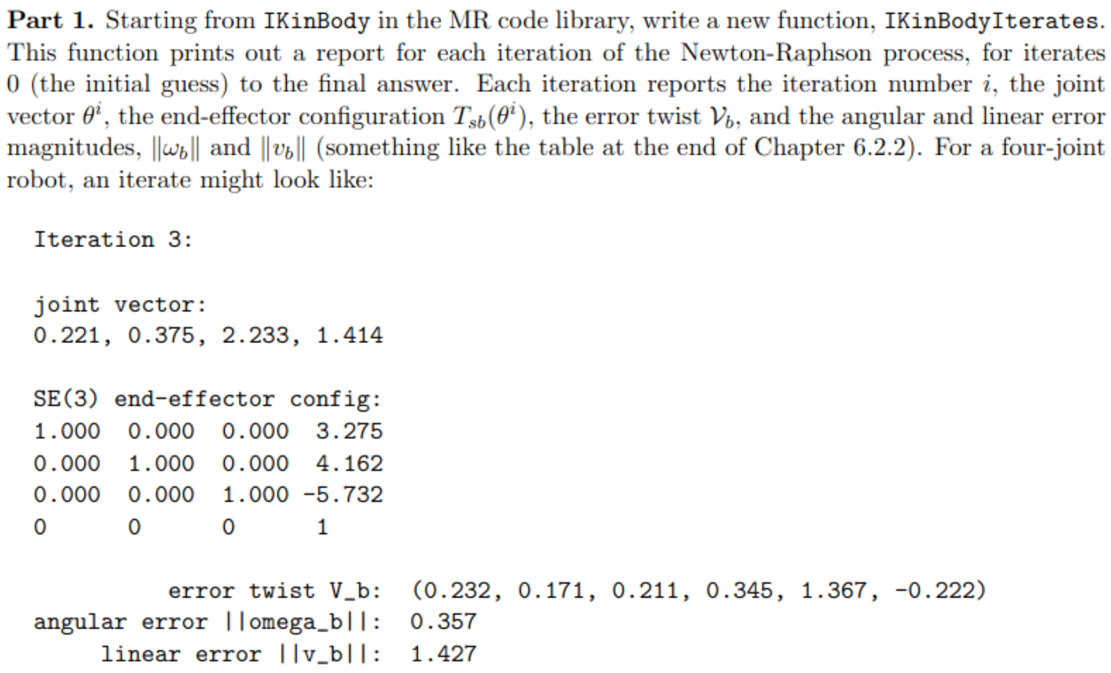
ME449 Homework 2 - Zhengyang Kris Weng submission 10/31/2024

Part 1:



Part 1. Starting from `IKinBody` in the MR code library, write a new function, `IKinBodyIterates`. This function prints out a report for each iteration of the Newton-Raphson process, for iterates 0 (the initial guess) to the final answer. Each iteration reports the iteration number i, the joint vector θ^i, the end-effector configuration $T_{sb}(\theta^i)$, the error twist $\mathcal{V}_b$, and the angular and linear error magnitudes, $\|\omega_b\|$ and $\|v_b\|$ (something like the table at the end of Chapter 6.2.2). For a four-joint robot, an iterate might look like:

```
Iteration 3:

joint vector:
0.221, 0.375, 2.233, 1.414

SE(3) end-effector config:
1.000  0.000  0.000  3.275
0.000  1.000  0.000  4.162
0.000  0.000  1.000 -5.732
0      0      0      1

          error twist V_b:  (0.232, 0.171, 0.211, 0.345, 1.367, -0.222)
angular error ||omega_b||:  0.357
     linear error ||v_b||:  1.427
```

In [42]:

```python
import modern_robotics as mr
import numpy as np
import math

# Set NumPy print options to limit float precision to 2 decimal places
np.set_printoptions(precision=2, suppress=True)
# mr.IKinBody()
```

In [64]:

```
# The IKinBodyIterates function:
def IKinBodyIterates(Blist, M, T, thetalist0, eomg, ev):
    """Computes inverse kinematics in the body frame for an open chain robot
       iteratively using newton-raphson method

    :param Blist: The joint screw axes in the end-effector frame when the
                  manipulator is at the home position, in the format of a
                  matrix with axes as the columns
    :param M: The home configuration of the end-effector
    :param T: The desired end-effector configuration Tsd
    :param thetalist0: An initial guess of joint angles that are close to
                       satisfying Tsd
    :param eomg: A small positive tolerance on the end-effector orientation
                 error. The returned joint angles must give an end-effector
                 orientation error less than eomg
    :param ev: A small positive tolerance on the end-effector linear position
               error. The returned joint angles must give an end-effector
               position error less than ev
    :return thetalist: Joint angles that achieve T within the specified
                       tolerances,
    :return success: A logical value where TRUE means that the function found
                     a solution and FALSE means that it ran through the set
                     number of maximum iterations without finding a solution
                     within the tolerances eomg and ev.
    Uses an iterative Newton-Raphson root-finding method.
    The maximum number of iterations before the algorithm is terminated has
    been hardcoded in as a variable called maxiterations. It is set to 20 at
    the start of the function, but can be changed if needed.

    Example Input:
        Blist = np.array([[0, 0, -1, 2, 0,   0],
                          [0, 0,  0, 0, 1,   0],
                          [0, 0,  1, 0, 0, 0.1]]).T
        M = np.array([[-1, 0,  0, 0],
                      [ 0, 1,  0, 6],
                      [ 0, 0, -1, 2],
                      [ 0, 0,  0, 1]])
        T = np.array([[0, 1,  0,     -5],
                      [1, 0,  0,      4],
                      [0, 0, -1, 1.6858],
                      [0, 0,  0,      1]])
        thetalist0 = np.array([1.5, 2.5, 3])
        eomg = 0.01
        ev = 0.001
    Output:
        (np.array([1.57073819, 2.999667, 3.14153913]), True)
    """
    thetalist = np.array(thetalist0).copy()
    i = 0
    maxiterations = 20
    Vb = mr.se3ToVec(mr.MatrixLog6(np.dot(mr.TransInv(mr.FKinBody(M, Blist,
                                                      thetalist)), T)))
    err = np.linalg.norm([Vb[0], Vb[1], Vb[2]]) > eomg \
        or np.linalg.norm([Vb[3], Vb[4], Vb[5]]) > ev

    traj = np.array([thetalist])
    ee_pos = np.array(mr.FKinBody(M, Blist, thetalist))[:3,3].T
    err_angle = np.linalg.norm([Vb[0], Vb[1], Vb[2]])
    err_pos = np.linalg.norm([Vb[3], Vb[4], Vb[5]])
    err_list = np.array([err_angle, err_pos])


    while err and i < maxiterations:
        thetalist = thetalist \
            + np.dot(np.linalg.pinv(mr.JacobianBody(Blist,
                                                    thetalist)), Vb)

        thetalist = [math.atan2(np.sin(theta), np.cos(theta)) for theta in thetalist]
        thetalist = np.array(thetalist)

        i = i + 1
        Vb \
            = mr.se3ToVec(mr.MatrixLog6(np.dot(mr.TransInv(mr.FKinBody(M, Blist,
                                                                       thetalist)), T)))
        err = np.linalg.norm([Vb[0], Vb[1], Vb[2]]) > eomg \
            or np.linalg.norm([Vb[3], Vb[4], Vb[5]]) > ev

        print(f"Iteration {i}:\n")
        print("joint vector:")
        print(f"{thetalist}\n")
        print("SE(3) end-effector config:\n")
        print(f"{mr.FKinBody(M, Blist, thetalist)}\n")

        print(f"          error twist V_b: {Vb}")
        print(f"angular error ||omega_b||: {np.linalg.norm([Vb[0], Vb[1], Vb[2]])}")
        print(f"     linear error ||v_b||: {np.linalg.norm([Vb[3], Vb[4], Vb[5]])}\n")

        traj = np.vstack([traj, thetalist])
        ee_pos = np.vstack([ee_pos, np.array(mr.FKinBody(M, Blist, thetalist))[:3,3].T])
        err_list = np.vstack([err_list, np.array([np.linalg.norm([Vb[0], Vb[1], Vb[2]]), np.lin
        print(f"Trajectory:\n{traj}\n")

    # Save to csv file
    np.savetxt("IKinBodyIterates.csv", traj, delimiter=",")

    return (thetalist, not err, i, traj, ee_pos, err_list)


# Example:
J1_B = np.array([0,0,1,0,3,0])
J2_B = np.array([0,0,1,0,2,0])
J3_B = np.array([0,0,1,0,1,0])

Blist = np.column_stack([J1_B, J2_B, J3_B])

T_sb = np.array([[1, 0, 0, 3],[0,1,0,0],[0,0,1,0],[0,0,0,1]])
T_sd = np.array([[-0.585, -0.811, 0, 0.076],[0.811, -0.585, 0, 2.608],[0,0,1,0],[0,0,0,1]])
result = IKinBodyIterates(Blist, T_sb, T_sd, np.array([np.pi/4, np.pi/4, np.pi/4]), 0.01, 0.001
```

```
Iteration 1:

joint vector:
[0.91 0.63 0.66]

SE(3) end-effector config:

[[-0.59 -0.81  0.    0.06]
 [ 0.81 -0.59  0.    2.6 ]
 [ 0.    0.    1.    0.  ]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 0.    0.   -0.    0.   -0.02  0.  ]
angular error ||omega_b||: 2.898944863771553e-06
     linear error ||v_b||: 0.015631596143492823

Trajectory:
[[0.79 0.79 0.79]
 [0.91 0.63 0.66]]

Iteration 2:

joint vector:
[0.92 0.59 0.68]

SE(3) end-effector config:

[[-0.59 -0.81  0.    0.08]
 [ 0.81 -0.59  0.    2.61]
 [ 0.    0.    1.    0.  ]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 0.  0. -0.  0. -0.  0.]
angular error ||omega_b||: 5.218183997954751e-11
     linear error ||v_b||: 0.00046494374043423795

Trajectory:
[[0.79 0.79 0.79]
 [0.91 0.63 0.66]
 [0.92 0.59 0.68]]
```

In [90]:

```
# Construct UR5 Model

# Config:
W1 = 0.109
W2 = 0.082
L1 = 0.425
L2 = 0.392
H1 = 0.089
H2 = 0.095

# Blist:
J1_B = np.array([0, 1, 0, W1+W2, 0, L1+L2])
J2_B = np.array([0, 0, 1, H2, -(L1+L2), 0])
J3_B = np.array([0, 0, 1, H2, -L2, 0])
J4_B = np.array([0, 0, 1, H2, 0, 0])
J5_B = np.array([0, -1, 0, -W2, 0, 0])
J6_B = np.array([0, 0, 1, 0, 0, 0])

# Home config:
M = np.array([[-1, 0, 0, L1+L2], [0, 0, 1, W1+W2],
             [0, 1, 0, H1-H2], [0, 0, 0, 1]])

Blist = np.column_stack([J1_B, J2_B, J3_B, J4_B, J5_B, J6_B])

T_sd = np.array([[1, 0, 0, 0.3], [0, 1, 0, 0.3], [0, 0, 1, 0.4], [0, 0, 0, 1]])


# wrap answers from -2pi to 2pi
def wrap_joint_range(thetalist):
    for i in range(len(thetalist)):
        thetalist[i] = thetalist[i] % (2*np.pi)
    return thetalist
```

Now, testing out the new function:

In [91]:

```python
init_guess_bad = np.array([-0.4, -5, -2, -1, -1, 0])
init_guess_good = np.array([0.5, 0.3, 4.0, 3.0, 4.5, 1.0])

result_bad = IKinBodyIterates(Blist, M, T_sd, init_guess_bad, 0.0001, 0.001)

print(f"Joint angles for bad guess: {result_bad[0]}. It took {result_bad[2]} iterations to conv

# Plot a 3d trajectory of the joint angles
def plot_trajectory(bad_ee, good_ee):
    import matplotlib.pyplot as plt
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    ax.plot(bad_ee[:,0], bad_ee[:,1], bad_ee[:,2], label='bad guess')
    ax.plot(good_ee[:,0], good_ee[:,1], good_ee[:,2], label='good guess')
    ax.plot(bad_ee[-1,0], bad_ee[-1,1], bad_ee[-1,2], 'x', label='end')
    ax.plot(bad_ee[0,0], bad_ee[0,1], bad_ee[0,2], 'o', label='start of bad guess')
    ax.plot(good_ee[0,0], good_ee[0,1], good_ee[0,2], 'o', label='start of good guess')
    ax.set_xlabel('x (m)')
    ax.set_ylabel('y (m)')
    ax.set_zlabel('z (m)')
    ax.legend()
    ax.set_title('End-effector trajectory')
    plt.show()


def plot_pos_error(err_list_bad, err_list_good):
    import matplotlib.pyplot as plt
    fig, ax = plt.subplots()
    ax.plot(err_list_bad[:,1], label='bad guess')
    ax.plot(err_list_good[:,1], label='good guess')
    ax.set_xlabel('iteration')
    ax.set_ylabel('linear error')
    ax.legend()
    ax.set_title('Linear error vs iteration')
    plt.show()

def plot_angular_error(err_list_bad, err_list_good):
    import matplotlib.pyplot as plt
    fig, ax = plt.subplots()
    ax.plot(err_list_bad[:,0], label='bad guess')
    ax.plot(err_list_good[:,0], label='good guess')
    ax.set_xlabel('iteration')
    ax.set_ylabel('angular error')
    ax.legend()
    ax.set_title('Angular error vs iteration')
    plt.show()
```

```
Iteration 1:

joint vector:
[ 0.81  1.53 -1.87  2.39  0.94  2.11]

SE(3) end-effector config:

[[ 0.73  0.03 -0.68  0.07]
 [ 0.17 -0.98  0.14  0.31]
 [-0.66 -0.21 -0.72 -0.22]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 2.74  0.16 -1.08  0.07 -0.97  0.07]
angular error ||omega_b||: 2.9527846944857012
     linear error ||v_b||: 0.9767389130075668

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]]

Iteration 2:

joint vector:
[ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]

SE(3) end-effector config:

[[-0.7  -0.25  0.67 -0.48]
 [-0.13  0.97  0.23  0.22]
 [-0.7   0.07 -0.71  0.62]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 0.27 -2.35 -0.21  0.64 -0.03  0.82]
angular error ||omega_b||: 2.374954220531417
     linear error ||v_b||: 1.0395124861318912

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]]

Iteration 3:

joint vector:
[ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]

SE(3) end-effector config:

[[ 0.62 -0.55  0.56  0.43]
 [-0.55  0.2   0.81 -0.35]
 [-0.56 -0.81 -0.18 -0.  ]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 1.44 -1.    0.   -0.4   0.26  0.7 ]
angular error ||omega_b||: 1.7530671267526747
     linear error ||v_b||: 0.8470288360301853

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]]

Iteration 4:

joint vector:
[-2.61  3.    2.6   0.64 -1.43 -2.45]

SE(3) end-effector config:

[[ 0.27 -0.27  0.93  0.23]
 [-0.72  0.58  0.38 -0.01]
 [-0.64 -0.77 -0.04  0.18]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 0.96 -1.31  0.38 -0.17  0.17  0.35]
angular error ||omega_b||: 1.665884627479942
     linear error ||v_b||: 0.4270183360364482
```

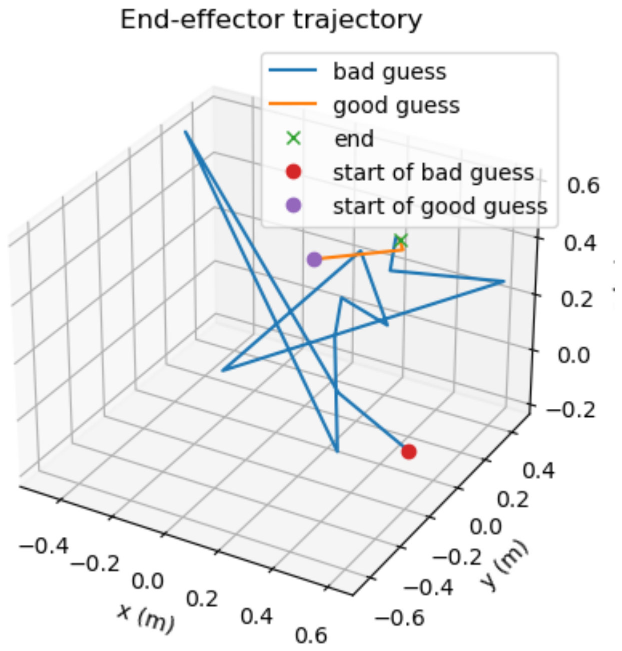
```
Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]]

Iteration 5:

joint vector:
[-1.24  2.28  2.85  2.7  -0.45 -2.23]

SE(3) end-effector config:

[[ 0.   -0.53  0.85  0.1 ]
 [ 0.82  0.49  0.3   0.26]
 [-0.57  0.69  0.43  0.16]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [-0.32 -1.15 -1.08  0.04 -0.    0.33]
angular error ||omega_b||: 1.608337037443677
     linear error ||v_b||: 0.33068142024396086

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]]

Iteration 6:

joint vector:
[-2.84  2.89  2.03  1.22 -1.44 -1.11]

SE(3) end-effector config:

[[-0.2  -0.1   0.98  0.42]
 [ 0.4   0.9   0.17  0.  ]
 [-0.89  0.43 -0.14  0.26]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [-0.24 -1.72 -0.46 -0.12  0.34 -0.01]
angular error ||omega_b||: 1.79223227627422
     linear error ||v_b||: 0.35803569759991566

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]
 [-2.84  2.89  2.03  1.22 -1.44 -1.11]]

Iteration 7:

joint vector:
[-1.92 -2.91  2.23  2.17  0.02 -1.65]

SE(3) end-effector config:

[[ 0.34  0.07  0.94  0.25]
 [ 0.93  0.13 -0.34  0.12]
 [-0.15  0.99 -0.02  0.42]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [-1.28 -1.05 -0.82  0.15  0.11 -0.09]
angular error ||omega_b||: 1.8511924126400205
     linear error ||v_b||: 0.2003822547952564

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]
 [-2.84  2.89  2.03  1.22 -1.44 -1.11]
 [-1.92 -2.91  2.23  2.17  0.02 -1.65]]

Iteration 8:

joint vector:
[-1.5  -0.45  0.15  0.79 -1.3  -1.47]

SE(3) end-effector config:

[[-0.13 -0.97  0.2   0.18]
 [ 0.48  0.12  0.87 -0.63]
 [-0.87  0.21  0.46  0.34]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 0.63 -1.03 -1.4   0.66  0.63  0.52]
angular error ||omega_b||: 1.8533399462683384
     linear error ||v_b||: 1.0485740888705484

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]
 [-2.84  2.89  2.03  1.22 -1.44 -1.11]
 [-1.92 -2.91  2.23  2.17  0.02 -1.65]
 [-1.5  -0.45  0.15  0.79 -1.3  -1.47]]

Iteration 9:

joint vector:
[-2.33  2.7   1.29 -2.61 -1.3  -2.74]

SE(3) end-effector config:

[[ 0.88  0.36  0.32  0.61]
 [-0.36  0.93 -0.05  0.46]
 [-0.31 -0.07  0.95  0.26]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [ 0.01 -0.33  0.37 -0.3  -0.22  0.09]
angular error ||omega_b||: 0.49650931669485643
     linear error ||v_b||: 0.37755015501298167

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]
 [-2.84  2.89  2.03  1.22 -1.44 -1.11]
 [-1.92 -2.91  2.23  2.17  0.02 -1.65]
 [-1.5  -0.45  0.15  0.79 -1.3  -1.47]
 [-2.33  2.7   1.29 -2.61 -1.3  -2.74]]

Iteration 10:

joint vector:
[-2.14  2.52  2.37  2.97 -1.57 -2.59]

SE(3) end-effector config:

[[ 1.    0.02  0.    0.29]
 [-0.02  1.   -0.01  0.25]
 [-0.    0.01  1.    0.31]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [-0.01 -0.    0.02  0.01  0.05  0.09]
angular error ||omega_b||: 0.020585385802748348
     linear error ||v_b||: 0.10328920337947964

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
```

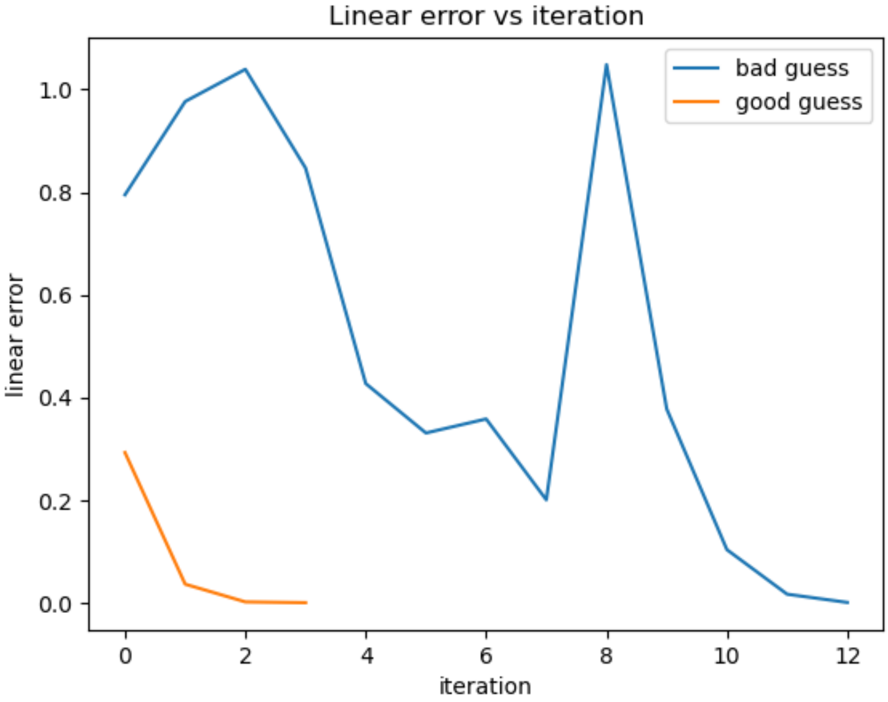
```
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]
 [-2.84  2.89  2.03  1.22 -1.44 -1.11]
 [-1.92 -2.91  2.23  2.17  0.02 -1.65]
 [-1.5  -0.45  0.15  0.79 -1.3  -1.47]
 [-2.33  2.7   1.29 -2.61 -1.3  -2.74]
 [-2.14  2.52  2.37  2.97 -1.57 -2.59]]

Iteration 11:

joint vector:
[-2.09  2.8   2.17  2.88 -1.57 -2.62]

SE(3) end-effector config:

[[ 1.   -0.   -0.    0.29]
 [ 0.    1.   -0.    0.29]
 [ 0.    0.    1.    0.41]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [-0.    0.   -0.    0.01  0.01 -0.01]
angular error ||omega_b||: 0.0002827051768145772
     linear error ||v_b||: 0.016539266212810502

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]
 [-2.84  2.89  2.03  1.22 -1.44 -1.11]
 [-1.92 -2.91  2.23  2.17  0.02 -1.65]
 [-1.5  -0.45  0.15  0.79 -1.3  -1.47]
 [-2.33  2.7   1.29 -2.61 -1.3  -2.74]
 [-2.14  2.52  2.37  2.97 -1.57 -2.59]
 [-2.09  2.8   2.17  2.88 -1.57 -2.62]]

Iteration 12:

joint vector:
[-2.1   2.77  2.15  2.93 -1.57 -2.62]

SE(3) end-effector config:

[[ 1.   0.   0.   0.3]
 [-0.   1.  -0.   0.3]
 [-0.   0.   1.   0.4]
 [ 0.   0.   0.   1. ]]

          error twist V_b: [-0. -0.  0. -0.  0.  0.]
angular error ||omega_b||: 1.0556676702073443e-06
     linear error ||v_b||: 0.00044402764020668253

Trajectory:
[[-0.4  -5.   -2.   -1.   -1.    0.  ]
 [ 0.81  1.53 -1.87  2.39  0.94  2.11]
 [ 2.58 -0.72 -0.42  0.14 -2.15 -2.18]
 [ 2.51 -2.37 -1.92 -0.56  2.96 -1.11]
 [-2.61  3.    2.6   0.64 -1.43 -2.45]
 [-1.24  2.28  2.85  2.7  -0.45 -2.23]
 [-2.84  2.89  2.03  1.22 -1.44 -1.11]
 [-1.92 -2.91  2.23  2.17  0.02 -1.65]
 [-1.5  -0.45  0.15  0.79 -1.3  -1.47]
 [-2.33  2.7   1.29 -2.61 -1.3  -2.74]
 [-2.14  2.52  2.37  2.97 -1.57 -2.59]
 [-2.09  2.8   2.17  2.88 -1.57 -2.62]
 [-2.1   2.77  2.15  2.93 -1.57 -2.62]]

Joint angles for bad guess: [-2.1   2.77  2.15  2.93 -1.57 -2.62]. It took 12 iterations to converge.
```

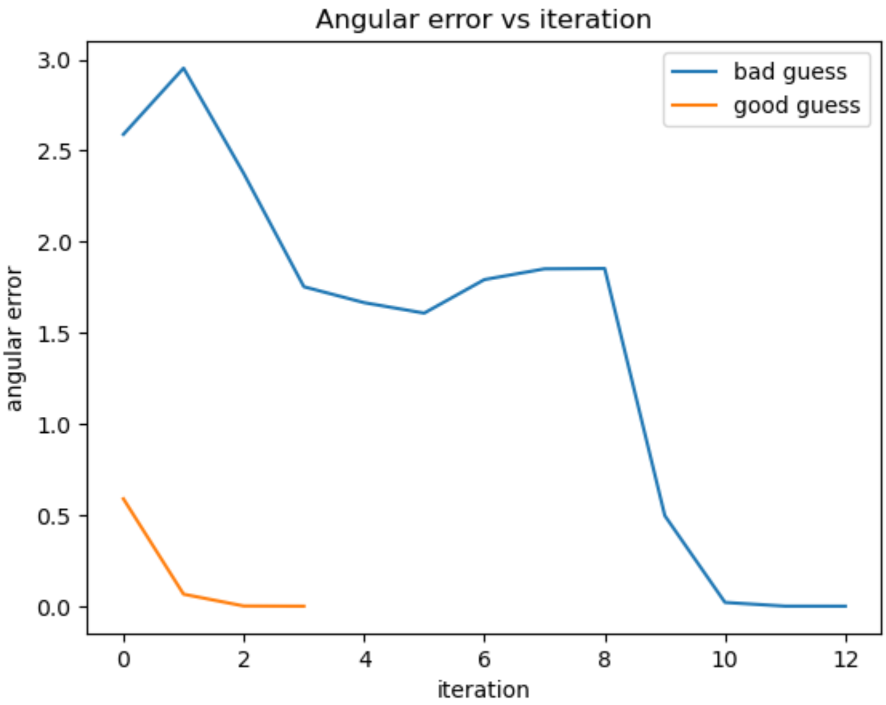
In [92]:

```
result_good = IKinBodyIterates(Blist, M, T_sd, init_guess_good, 0.0001, 0.001)
# joint_list_good = wrap_joint_range(result_good[0])
print(f"Joint angles for good guess: {result_good[0]}. It took {result_good[2]} iterations to c
plot_trajectory(result_bad[4], result_good[4])
plot_pos_error(result_bad[5], result_good[5])
plot_angular_error(result_bad[5], result_good[5])
```

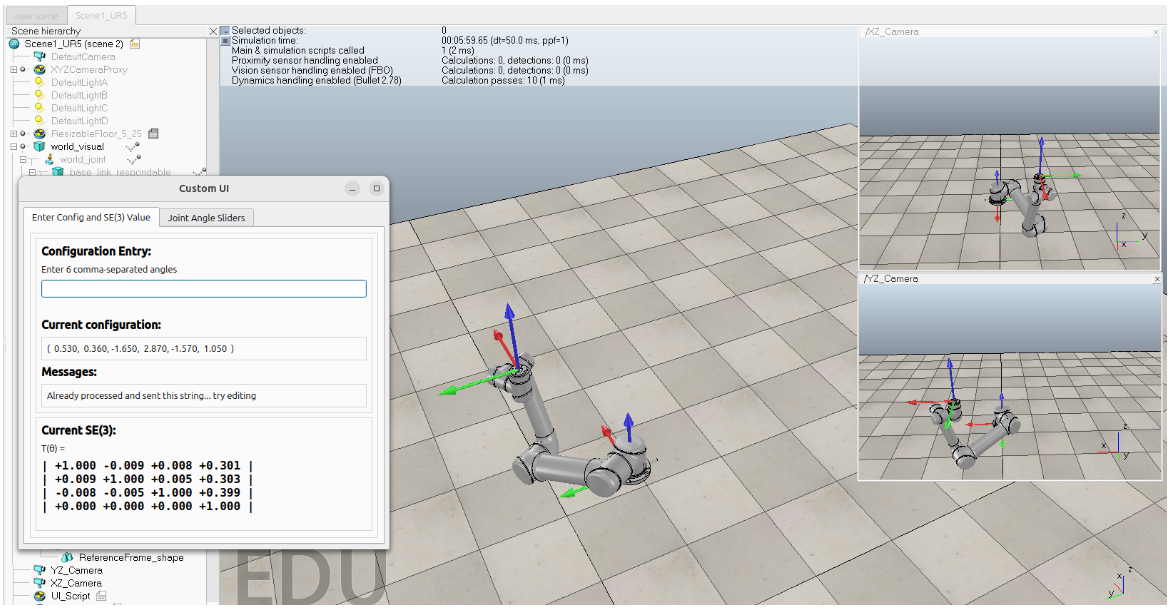
```
Iteration 1:

joint vector:
[ 0.51  0.43 -1.67  2.83 -1.59  1.12]

SE(3) end-effector config:

[[ 1.   -0.06  0.02  0.31]
 [ 0.06  1.   -0.01  0.3 ]
 [-0.02  0.01  1.    0.37]
 [ 0.    0.    0.    1.  ]]

          error twist V_b: [-0.01 -0.02 -0.06 -0.01  0.    0.03]
angular error ||omega_b||: 0.06586899104348297
     linear error ||v_b||: 0.03607246766369

Trajectory:
[[ 0.5   0.3   4.    3.    4.5   1.  ]
 [ 0.51  0.43 -1.67  2.83 -1.59  1.12]]

Iteration 2:

joint vector:
[ 0.52  0.36 -1.66  2.87 -1.57  1.05]

SE(3) end-effector config:

[[ 1.   0.  -0.   0.3]
 [-0.   1.  -0.   0.3]
 [ 0.   0.   1.   0.4]
 [ 0.   0.   0.   1. ]]

          error twist V_b: [-0.  0.  0.  0.  0.  0.]
angular error ||omega_b||: 0.00115005379318028
     linear error ||v_b||: 0.001500755483736736

Trajectory:
[[ 0.5   0.3   4.    3.    4.5   1.  ]
 [ 0.51  0.43 -1.67  2.83 -1.59  1.12]
 [ 0.52  0.36 -1.66  2.87 -1.57  1.05]]

Iteration 3:

joint vector:
[ 0.53  0.36 -1.65  2.87 -1.57  1.05]

SE(3) end-effector config:

[[ 1.  -0.  -0.   0.3]
 [ 0.   1.   0.   0.3]
 [ 0.  -0.   1.   0.4]
 [ 0.   0.   0.   1. ]]

          error twist V_b: [ 0.  0. -0.  0. -0.  0.]
angular error ||omega_b||: 2.084407717138313e-06
     linear error ||v_b||: 3.1283584271726847e-06

Trajectory:
[[ 0.5   0.3   4.    3.    4.5   1.  ]
 [ 0.51  0.43 -1.67  2.83 -1.59  1.12]
 [ 0.52  0.36 -1.66  2.87 -1.57  1.05]
 [ 0.53  0.36 -1.65  2.87 -1.57  1.05]]

Joint angles for good guess: [ 0.53  0.36 -1.65  2.87 -1.57  1.05]. It took 3 iterations to con
verge.
```


End-effector trajectory
bad guess
good guess
end
start of bad guess
start of good guess
−0.4
−0.2
0.0
0.2
0.4
0.6
x (m)
−0.6
−0.4
−0.2
0.0
0.2
0.4
y (m)
−0.2
0.0
0.2
0.4
0.6


Linear error vs iteration
bad guess
good guess
linear error
0.0
0.2
0.4
0.6
0.8
1.0
iteration
0
2
4
6
8
10
12



Simulation result from good guess:



```
Selected objects:                         0
Simulation time:                          00:05:59.65 (dt=50.0 ms, ppf=1)
Main & simulation scripts called          1 (2 ms)
Proximity sensor handling enabled         Calculations: 0, detections: 0 (0 ms)
Vision sensor handling enabled (FBO)      Calculations: 0, detections: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)   Calculation passes: 10 (1 ms)
```

Custom UI

Enter Config and SE(3) Value | Joint Angle Sliders

Configuration Entry:

Enter 6 comma-separated angles

Current configuration:

(0.530, 0.360, -1.650, 2.870, -1.570, 1.050)

Messages:

Already processed and sent this string... try editing

Current SE(3):

T(θ) =

```
| +1.000 -0.009 +0.008 +0.301 |
| +0.009 +1.000 +0.005 +0.303 |
| -0.008 -0.005 +1.000 +0.399 |
| +0.000 +0.000 +0.000 +1.000 |
```

It's harder for the bad initial guess to converge to the result since it starts from joint configurations having many opposite signs in the initial guess. It needs to update more compared to a good initial guess to reach to a satisfactory joint configuration that yields minimal error.

Part 2:

2 (a):

Given an initial joint config θ^0 and a desired joint config θ^*, assuming we want to reach to a desired end-effector config in t seconds while moving in constant joint angle, we can command a joint speed vector $\dot{\theta}$:

$$\dot{\theta} = \frac{\theta^0 - \theta^*}{t_f}$$

This yields constant joint speeds for all joints in the robot.

2 (b):

In order to achieve a constant end-effector twist V_{ee}, we first find the twist required to move from T^0 to T_{sb}:

$$V_{ee} = vec(log(T^0) - log(T_{sb}))/t_f$$

Now, this end-effector V_{ee} can be achieved through:

$$V_{ee} = J_b(\theta)\dot{\theta}$$

Therefore, to get this end-effector V_{ee}, we need:

$$\dot{\theta} = J_b(\theta)^{-1}V_{ee}$$

Specifically, at time t=0:

$$\dot{\theta}_{t=0} = J_b(\theta_0)^{-1}vec(log(T^0) - log(T_{sb}))/t_f$$

And at time t = $t_f/2$:

$$\dot{\theta}_{t=t_f/2} = J_b(\theta_{t_f/2})^{-1}vec(log(T^0) - log(T_{sb}))/t_f$$

2 (c):

Since approach in (a) has more direct control over joint velocities, it's less likely for it to violate any joint velocity limit - where as the approach in (b) derives joint velocity as a function of the end-effector twist, it's possible to yield something that the robot joint actuator cannot achieve.

However, it's also possible for appoach (a) to yield an end-effector twist that's dangerous since it's agnostic of the resulting end-effector twist, which could be dangerous in a real robot. Approach (b) avoids this issue since it directly controls the end-effector twist.