6ELEN018W - Tutorial 4 2026 Solutions

Exercise 3

[1]:
```python
from sympy import *
from roboticstoolbox import *
from spatialmath.base import *
import numpy as np
```

[2]:
```python
a1=1; a2 =1; a3=1;
link1 = Link2(ET2.R(), name="link1")
link2 = Link2(ET2.tx(a1)*ET2.R(), name="link2",parent=link1)
link3 = Link2(ET2.tx(a2)*ET2.R(), name="link3", parent=link2)
link4 = Link2(ET2.tx(a3), name="link4", parent=link3)
robot = ERobot2([link1, link2, link3, link4], name="my_robot")
te = robot.fkine(np.deg2rad([30, 40, 50]))

# Find the Jacobian of the robot for a specific configuration
# The jacobian() function works only for the Matrix class in SymPy which needs␣
↪symbolic q variables
J = robot.jacob0(np.deg2rad([30, 40, 50]))
print(f'Jacobian: {J}\n')

Q = np.linalg.inv(J)
print(f'Inverse of Jacobian: {Q}\n')

#qdot = np.array([1, 1.2, 3])
#print(J@qdot)  # v for end-effector

v = np.array([-7.07, -0.98, 5.2])  # desired speed for end-effector
qdot = Q@v
print(f'Required velocities for the joints: {qdot}')
```

```
Jacobian: [[-2.30571802 -1.80571802 -0.8660254 ]
 [ 0.70804555 -0.15797986 -0.5       ]
 [ 1.          1.          1.        ]]

Inverse of Jacobian: [[ 0.53208889  1.4619022   1.19175359]
 [-1.87938524 -2.23976411 -2.74747742]
```

```
 [ 1.34729636  0.77786191  2.55572383]]

Required velocities for the joints: [1.0025861  1.19533991 3.00207399]
```