In [5]:
```
import numpy as np
from scipy.linalg import expm, logm
```

Problem 1

In [6]:
```
import numpy as np

A = np.array([[0.81790325, 0.11007091, -0.56472884, 0.24637657], [0.27105327, 0.79206968, 0.54695133, 0.13113728], [0.50750802, -0.60042487, 0.61800128, -0.42045502], [0.00000000, 0.00000000, 0.00000000, 1.00000000]])
B = np.array([[-0.75687540, 0.23259733, 0.61076847, -0.95712585], [0.29343167, -0.71410204, 0.63557544, -0.01909166], [0.58398415, 0.66027022, 0.47223484, 0.77973625], [0.00000000, 0.00000000, 0.00000000, 1.00000000]])
p_sensor = np.array([[0.14800855], [0.77770102], [-0.97742119]])
```

In [7]:
```
# Let's first see what A and B look like
print("A:\n", A)
print(A.shape)
print("\nB:\n", B)
print(B.shape)
print("\np_sensor:\n", p_sensor)
print(p_sensor.shape)
```

```
A:
 [[ 0.81790325  0.11007091 -0.56472884  0.24637657]
 [ 0.27105327  0.79206968  0.54695133  0.13113728]
 [ 0.50750802 -0.60042487  0.61800128 -0.42045502]
 [ 0.          0.          0.          1.        ]]
(4, 4)

B:
 [[-0.7568754   0.23259733  0.61076847 -0.95712585]
 [ 0.29343167 -0.71410204  0.63557544 -0.01909166]
 [ 0.58398415  0.66027022  0.47223484  0.77973625]
 [ 0.          0.          0.          1.        ]]
(4, 4)

p_sensor:
 [[ 0.14800855]
 [ 0.77770102]
 [-0.97742119]]
(3, 1)
```

In [8]:
```
p_robot = B @ p_sensor
print("\np_robot:\n", p_robot)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[8], line 1
----> 1 p_robot = B @ p_sensor
      2 print("\np_robot:\n", p_robot)

ValueError: matmul: Input operand 1 has a mismatch in its core dimension 0, with gufunc signature (n?,k),(k,m?)->(n?,m?) (size 3 is different from 4)
```

In [9]:
```
# Why did that fail? Because p_sensor is 3x1 and B is 4x4. We need to make p_sensor homogeneous by adding a 1 at the end.
p_sensor_homogeneous = np.block([[p_sensor], [1]])
print("\np_sensor_homogeneous:\n", p_sensor_homogeneous)
print(p_sensor_homogeneous.shape)
```

```
p_sensor_homogeneous:
 [[ 0.14800855]
 [ 0.77770102]
 [-0.97742119]
 [ 1.        ]]
(4, 1)
```

In [10]:
```
p_robot_homo = B @ p_sensor_homogeneous
p_global_homo = A @ p_robot_homo
print("\np_global_homo:\n", p_global_homo)

p_global_direct_homo = A @ B @ p_sensor_homogeneous
print("\np_global_direct_homo:\n", p_global_direct_homo)
```

```
p_global_homo:
 [[-1.61370462]
 [-0.68194727]
 [ 0.08499305]
 [ 1.        ]]

p_global_direct_homo:
 [[-1.61370462]
 [-0.68194727]
 [ 0.08499305]
 [ 1.        ]]
```

In [11]:
```
p_global = p_global_homo[:3]
print("\np_global:\n", p_global.tolist())
print(p_global.shape)
```

```
p_global:
 [[-1.613704622556333], [-0.6819472714861399], [0.08499305455675121]]
(3, 1)
```

Problem 2

In [12]:
```
import numpy as np

A = np.array([[1.0000, 0.0000, 0.0000], [0.0000, 0.6031, -0.7977], [0.0000, 0.7977, 0.6031]])
B = np.array([[-0.0105, 0.0000, 0.9999], [0.0000, 1.0000, 0.0000], [-0.9999, 0.0000, -0.0105]])
```

In [13]:
```
print("\nA:\n", A)
print(A.shape)
print("\nB:\n", B)
print(B.shape)
```

```
A:
 [[ 1.      0.      0.    ]
 [ 0.      0.6031 -0.7977]
 [ 0.      0.7977  0.6031]]
(3, 3)

B:
 [[-0.0105  0.      0.9999]
 [ 0.      1.      0.    ]
 [-0.9999  0.     -0.0105]]
(3, 3)
```

In [14]:
```
AB = A @ B
BA = B @ A
print("\nAB:\n", AB.tolist())
print("\nBA:\n", BA.tolist())
if np.all(AB == BA):
    print('They commute!')
else:
    print('They do not commute.')
```

```
AB:
 [[-0.0105, 0.0, 0.9999], [0.7976202299999999, 0.6031, 0.00837585], [-0.60303969, 0.7977, -0.00633255]]

BA:
 [[-0.0105, 0.7976202299999999, 0.60303969], [0.0, 0.6031, -0.7977], [-0.9999, -0.00837585, -0.00633255]]
They do not commute.
```

Problem 3

In [15]:
```
import numpy as np

w_sbinb = np.array([[-1], [-1], [-6]])
v_sbinb = np.array([[ 1], [ 2], [-8]])
```

In [16]:
```
twist_sbinb = np.block([[w_sbinb], [v_sbinb]])
print(twist_sbinb.tolist())
print(twist_sbinb.shape)
```

```
[[-1], [-1], [-6], [1], [2], [-8]]
(6, 1)
```

Problem 4

In [17]:
```
import numpy as np

V_mat = np.array([[  0,  10,   8,  -8], [-10,   0,   9,  -8], [ -8,  -9,   0,   5], [  0,   0,   0,   0]])
```

In [18]:
```
twist = np.array([[V_mat[2,1]],[V_mat[0,2]],[V_mat[1,0]],[V_mat[0,3]],[V_mat[1,3]],[V_mat[2,3]]])
print(twist.tolist())
print(twist.shape)
```

```
[[-9], [8], [-10], [-8], [-8], [5]]
(6, 1)
```

Problem 5

In [19]:
```
import numpy as np

M = np.array([[0.14646782, -0.54878165, 0.82303456, 0.11733897], [0.38572645, 0.79783280, 0.46333350, -2.85007510], [-0.91091288, 0.24960275, 0.32853643, -0.73371172], [0.00000000, 0.00000000, 0.00000000, 1.00000000]])
S = np.array([[0.84568872], [-0.43027893], [-0.31570656], [0.26600728], [0.48278670], [2.53120659]])
theta = 0.37859059
```

In [20]:
```
print("\nM:\n", M)
print(M.shape)
print("\nS:\n", S)
print(S.shape)
```

```
M:
 [[ 0.14646782 -0.54878165  0.82303456  0.11733897]
 [ 0.38572645  0.7978328   0.4633335  -2.8500751 ]
 [-0.91091288  0.24960275  0.32853643 -0.73371172]
 [ 0.          0.          0.          1.        ]]
(4, 4)

S:
 [[ 0.84568872]
 [-0.43027893]
 [-0.31570656]
 [ 0.26600728]
 [ 0.4827867 ]
 [ 2.53120659]]
(6, 1)
```

In [21]:
```
bracket_omega = np.array([[0, -S[2,0], S[1,0]], [S[2,0], 0, -S[0,0]], [-S[1,0], S[0,0], 0]])
print("\nbracket_omega:\n", bracket_omega)
print(bracket_omega.shape)
```

```
bracket_omega:
 [[ 0.          0.31570656 -0.43027893]
 [-0.31570656  0.         -0.84568872]
 [ 0.43027893  0.84568872  0.        ]]
(3, 3)
```

In [22]:
```
bracket_v_sb = np.block([[bracket_omega, S[3:]], [0, 0, 0, 0]])
print("\nbracket_v_sb:\n", bracket_v_sb)
print(bracket_v_sb.shape)
```

```
bracket_v_sb:
 [[ 0.          0.31570656 -0.43027893  0.26600728]
 [-0.31570656  0.         -0.84568872  0.4827867 ]
 [ 0.43027893  0.84568872  0.          2.53120659]
 [ 0.          0.          0.          0.        ]]
(4, 4)
```

In [23]:
```
transform = expm(bracket_v_sb * theta)
print("\ntransform:\n", transform)
print(transform.shape)

T_sb = transform @ M
print("\nT_sb:\n", T_sb)
print(T_sb.shape)
print(T_sb.tolist())
```

```
transform:
 [[ 0.97983156  0.09092093 -0.17794239  0.02605036]
 [-0.14245645  0.9422969  -0.30295661  0.02393309]
 [ 0.14012946  0.32219549  0.93624452  0.97479631]
 [ 0.          0.          0.          1.        ]]
(4, 4)

T_sb:
 [[ 0.34067441 -0.50958879  0.79010139  0.01244973]
 [ 0.61857063  0.75435406  0.21981885 -2.45611673]
 [-0.70803341  0.41384686  0.57220579 -0.61397597]
 [ 0.          0.          0.          1.        ]]
(4, 4)
[[0.34067441160577683, -0.5095887931645415, 0.790101394195915, 0.012449729473892751], [0.6185706304169211, 0.7543540607138678, 0.2198188534488579, -2.4561167289457986], [-0.7080334115970235, 0.4138468580353478, 0.5722057877825854, -0.6139759668607697], [0.0, 0.0, 0.0, 1.0]]
```

Problem 6

In [24]:
```
import numpy as np

M = np.array([[0.92312088, -0.22311794, 0.31315529, 2.87892900], [-0.31001199, -0.91366561, 0.26288347, -0.17707504], [0.22746520, -0.33975512, -0.91259303, 2.30944924], [0.00000000, 0.00000000, 0.00000000, 1.00000000]])
B = np.array([[0.00000000], [0.00000000], [0.00000000], [0.41835746], [0.87281427], [0.25134100]])
theta = 1.04488973
```

In [25]:
```
print("\nM:\n", M)
print(M.shape)
print("\nB:\n", B)
print(B.shape)
```

```
M:
 [[ 0.92312088 -0.22311794  0.31315529  2.878929  ]
 [-0.31001199 -0.91366561  0.26288347 -0.17707504]
 [ 0.2274652  -0.33975512 -0.91259303  2.30944924]
 [ 0.          0.          0.          1.        ]]
(4, 4)

B:
 [[0.        ]
 [0.        ]
 [0.        ]
 [0.41835746]
 [0.87281427]
 [0.251341  ]]
(6, 1)
```

In [26]:
```
bracket_omega = np.array([[0, -B[2,0], B[1,0]], [B[2,0], 0, -B[0,0]], [-B[1,0], B[0,0], 0]])
print("\nbracket_omega:\n", bracket_omega)
print(bracket_omega.shape)
```

```
bracket_omega:
 [[ 0. -0.  0.]
 [ 0.  0. -0.]
 [-0.  0.  0.]]
(3, 3)
```

In [27]:
```
bracket_v_bs = np.block([[bracket_omega, B[3:]], [0, 0, 0, 0]])
print("\nbracket_v_bs:\n", bracket_v_bs)
print(bracket_v_bs.shape)
```

```
bracket_v_bs:
 [[ 0.         -0.          0.          0.41835746]
 [ 0.          0.         -0.          0.87281427]
 [-0.          0.          0.          0.251341  ]
 [ 0.          0.          0.          0.        ]]
(4, 4)
```

In [28]:
```
transform = expm(bracket_v_bs * theta)
print("\ntransform:\n", transform)
print(transform.shape)

T_bs = M @ transform
print("\nT_bs:\n", T_bs)
print(T_bs.shape)
print(T_bs.tolist())
```

```
transform:
 [[1.         0.         0.         0.43713741]
 [0.         1.         0.         0.91199467]
 [0.         0.         1.         0.26262363]
 [0.         0.         0.         1.        ]]
(4, 4)

T_bs:
 [[ 0.92312088 -0.22311794  0.31315529  3.16121928]
 [-0.31001199 -0.91366561  0.26288347 -1.07681163]
 [ 0.2274652  -0.33975512 -0.91259303  1.85935944]
 [ 0.          0.          0.          1.        ]]
(4, 4)
[[0.92312088, -0.22311794, 0.31315529, 3.161219281282569], [-0.31001199, -0.91366561, 0.26288347, -1.0768116320467134], [0.2274652, -0.33975512, -0.91259303, 1.8593594377610527], [0.0, 0.0, 0.0, 1.0]]
```

Problem 7

In [29]:
```
import numpy as np

T_1in0 = np.array([[0.58260087, -0.78693103, -0.20326285, -0.16331626], [0.61107616, 0.58900330, -0.52882988, 0.21503648], [0.53587512, 0.18388767, 0.82402863, 0.61116227], [0.00000000, 0.00000000, 0.00000000, 1.00000000]])
T_2in0 = np.array([[0.48872007, 0.01537159, -0.87230523, -0.05045123], [-0.48921424, 0.83269040, -0.25941498, 0.25574963], [0.72237256, 0.55352544, 0.41447251, 0.60413934], [0.00000000, 0.00000000, 0.00000000, 1.00000000]])
```

In [30]:
```
print("\nT_1in0:\n", T_1in0)
print("\nT_2in0:\n", T_2in0)
```

```
T_1in0:
 [[ 0.58260087 -0.78693103 -0.20326285 -0.16331626]
 [ 0.61107616  0.5890033  -0.52882988  0.21503648]
 [ 0.53587512  0.18388767  0.82402863  0.61116227]
 [ 0.          0.          0.          1.        ]]

T_2in0:
 [[ 0.48872007  0.01537159 -0.87230523 -0.05045123]
 [-0.48921424  0.8326904  -0.25941498  0.25574963]
 [ 0.72237256  0.55352544  0.41447251  0.60413934]
 [ 0.          0.          0.          1.        ]]
```

In [31]:
```
relative_transform = T_2in0 @ np.linalg.inv(T_1in0)
print("\nrelative_transform:\n", relative_transform)
```

```
relative_transform:
 [[ 0.4499396   0.76900017 -0.4540849   0.13518769]
 [-0.88755713  0.32869662 -0.32280162  0.23739953]
 [-0.09897832  0.54826752  0.8304252  -0.0374475 ]
 [ 0.          0.          0.          1.        ]]
```

In [32]:
```
bracket_v_sb = logm(relative_transform)
print("\nbracket_v_sb:\n", bracket_v_sb)
```

```
bracket_v_sb:
 [[-1.10365029e-09  1.09684727e+00 -2.35124786e-01 -1.47053672e-02]
 [-1.09684727e+00 -6.11639648e-10 -5.76756274e-01  2.67124755e-01]
 [ 2.35124794e-01  5.76756264e-01 -2.16372982e-09 -1.22637934e-01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]
```

In [33]:
```
omega_s = np.array([[bracket_v_sb[2,1]], [bracket_v_sb[0,2]], [bracket_v_sb[1,0]]])
print("\nomega_s:\n", omega_s)
v_s = np.array([[bracket_v_sb[0,3]], [bracket_v_sb[1,3]], [bracket_v_sb[2,3]]])
print("\nv_s:\n", v_s)
V_s = np.block([[omega_s], [v_s]])
```

```
print("\nV_s:\n", V_s)
print(V_s.tolist())
```

```
omega_s:
 [[ 0.57675626]
 [-0.23512479]
 [-1.09684727]]

v_s:
 [[-0.01470537]
 [ 0.26712476]
 [-0.12263793]]

V_s:
 [[ 0.57675626]
 [-0.23512479]
 [-1.09684727]
 [-0.01470537]
 [ 0.26712476]
 [-0.12263793]]
[[0.5767562640272914], [-0.23512478584302626], [-1.0968472725057055], [-0.014705367187613273], [0.26712475509121847], [-0.12263793438729467]]
```