Part 1)

A white board with math equations and formulas

AI-generated content may be incorrect.

A math equations and formulas on a white board

AI-generated content may be incorrect.

A whiteboard with blue writing

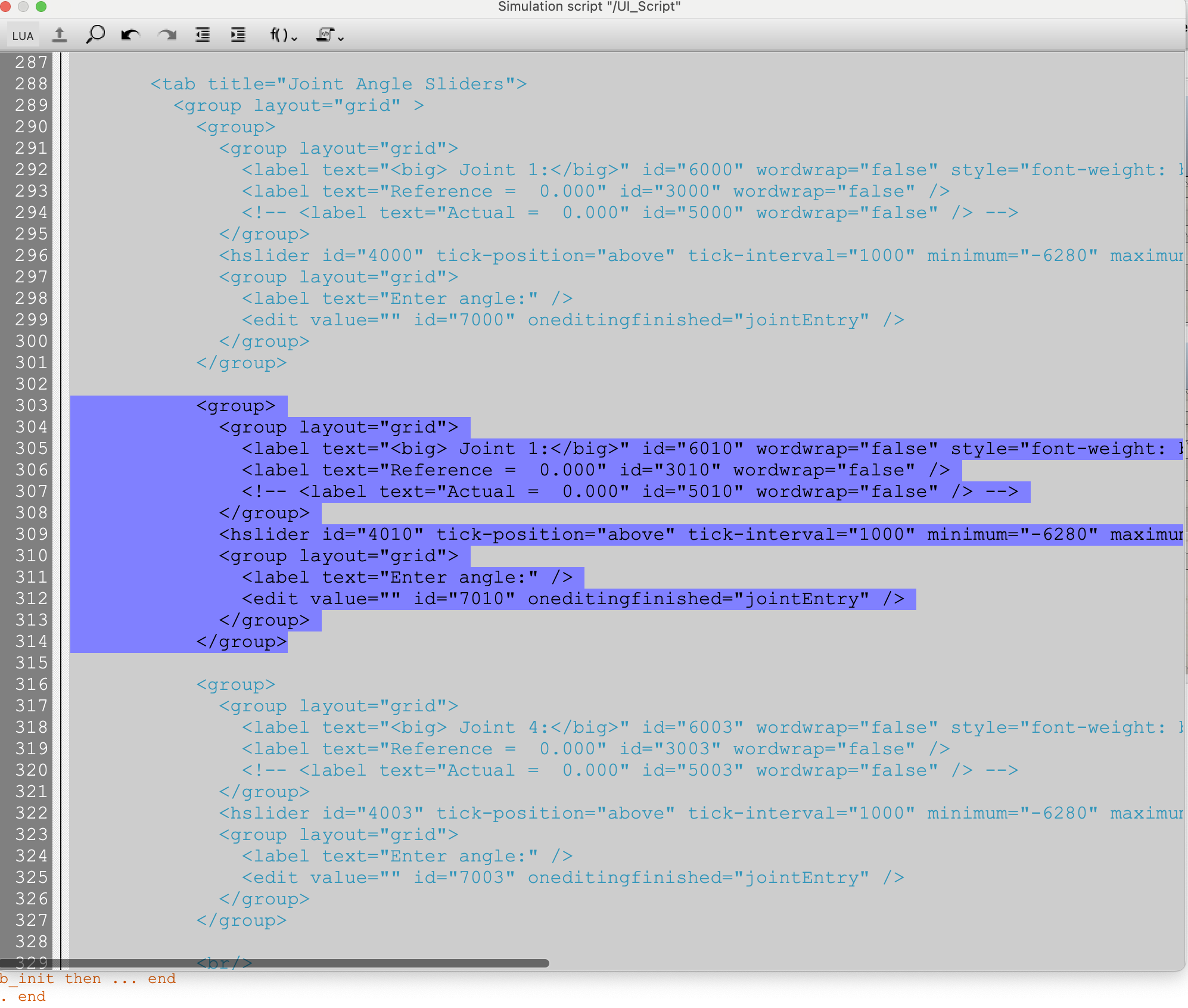
AI-generated content may be incorrect.

A whiteboard with a blue pen and a piece of paper

AI-generated content may be incorrect.

Part 2b)

I duplicated the section for Joint 1, because as the largest joint it must surely be the most important, and therefore deserves duplicate, albeit non-functioning, entries in the UI. In order to avoid runtime errors, the id of each new element was modified to be unique.



And here’s the new UI:

A screenshot of a computer

AI-generated content may be incorrect.

Part 3) Derivation

I first isolated single joint rotation matrices by using the subscript cancellation rule. I then used the matrix logarithm to convert each rotation matrix into so(3) skew-symmetric matrices. I extracted the ω vector from each so(3) matrix and finally multiplied by the transpose of the unit rotation vector to get the rotation angle.

A white board with blue writing on it

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.

Code:

w1 =np.array( [0,0,1])

w2 =np.array( [0, 1, 0])

w3 =np.array( w2)

w4 =np.array( w2)

w5 =np.array( [0, 0, -1])

w6 =np.array( w2)

R13 = np.array([[-0.7071, 0, -0.7071], [0, 1, 0], [0.7071, 0, -0.7071]])

Rs2 = np.array([[-0.6964, 0.1736, 0.6964], [-0.1228, -0.9848, 0.1228], [0.7071, 0, 0.7071]])

R25 = np.array([[-0.7566, -0.1198, -0.6428], [-0.1564, 0.9877, 0], [0.6348, 0.1005, -0.7661]])

R12 = np.array([[0.7071, 0, -0.7071], [0, 1, 0], [0.7071, 0, 0.7071]])

R34 = np.array([[0.6428, 0, -0.7660], [0, 1, 0], [0.7660, 0, 0.6428]])

Rs6 = np.array([[0.9418, 0.3249, -0.0859], [0.3249, -0.9456, -0.0151], [-0.0861, -0.0136, -0.9962]])

R6b = np.array([[-1, 0, 0], [0, 0, 1], [0, 1, 0]])

so3\_1 = mr.VecToso3(w1)

so3\_2 = mr.VecToso3(w2)

so3\_3 = mr.VecToso3(w3)

so3\_4 = mr.VecToso3(w4)

so3\_5 = mr.VecToso3(w5)

so3\_6 = mr.VecToso3(w6)

log12 = mr.MatrixLog3(R12)

v12 = mr.so3ToVec(log12)

a2\_test = v12 @ w2.T

Rs1 = Rs2 @ R12.T

R23 = R12.T @ R13

R45 = R34.T @ R23.T @ R25

R56 = (Rs1 @ R12 @ R23 @ R34 @ R45).T @ Rs6

a1 = mr.so3ToVec(mr.MatrixLog3(Rs1)) @ w1.T

a2 = mr.so3ToVec(mr.MatrixLog3(R12)) @ w2.T

a3 = mr.so3ToVec(mr.MatrixLog3(R23)) @ w3.T

a4 = mr.so3ToVec(mr.MatrixLog3(R34)) @ w4.T

a5 = mr.so3ToVec(mr.MatrixLog3(R45)) @ w5.T

a6 = mr.so3ToVec(mr.MatrixLog3(R56)) @ w6.T

angles = [a1, a2, a3, a4, a5, a6]

for i in range(len(angles)):

print("angle {}: {}".format(i+1, angles[i]))

Rsb = Rs1 @ R12 @ R23 @ R34 @ R45 @ R56 @ R6b

print("Rsb", Rsb)

pass

Output:

angle 1: -2.969482157066879

angle 2: -0.7853926894212007

angle 3: -1.5707661989213484

angle 4: -0.8726096667837093

angle 5: 0.15702980014757423

angle 6: 9.050365593053613e-07

Rsb [[-0.94155819 -0.08587249 0.32478673]

[-0.32492508 -0.01511086 -0.94558895]

[ 0.08609033 -0.9960651 -0.01360784]]