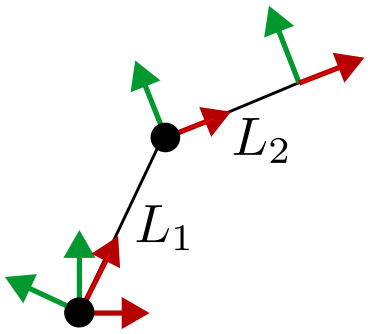
ECE 4560
Assignment 7

Due: October 24th, 11:59pm

This assignment will explore forward kinematics and the geometric approach to inverse kinematics.

1. Consider the following planar manipulator. Assume the following manipulator parameters: $L_1 = 2, L_2 = 4$. Solve the following inverse kinematics problem using the geometric approach. In all cases, be sure to specify whether you are using the "Elbow-Up" or "Elbow-Down" solution. **Even if you use code to solve this problem, show you work!**



   (a) (2 points) What is one set of joint angles that would move the end-effector to the position $(7, 0)$?

   (b) (2 points) What is one set of joint angles that would move the end-effector to the position $(3, 3)$?

   (c) (2 points) What is one set of joint angles that would move the end-effector to the position $(-2, 2)$?

2. (4 points) Using the manipulator from Assignment 6 (the UR5 manipulator) we will be repeating the forward kinematics problem using the Product of Exponentials method. The steps to complete this assignment are as follows:

   (a) If you haven't done so, clone the github repository for the UR5 manipulator. If you need help with git, either Prof. Tucker or one of the TAs can help.

   (b) Update the Product of Exponential parameters in lines 69-89 of 'forward_kinematics.py' (within the 'forward_kinematics_product_exp' function). Note that you should obtain the same final result as you obtained last week with the 'forward_kinematics_product_lie' function. Once you've defined the parameters, you can run the forward kinematics by running

```
python test-assignment7a.py
```

To get credit for this question, please write down your parameters for the product of exponentials (g_0 as well as ω_i and q_i for each joint) and upload a screenshot of the MuJoCo simulator. Note that as before, the blue cylinder should be located on top of the end-effector.

3. Finally, we will be exploring how to implement an optimization-based inverse kinematics solver for the UR5 manipulator. The code that I've prepared demonstrates how to implement inverse kinematics for MuJoCo, but the process would be identical if you'd like to replicate it for the visualizer.

   (a) (1 point) Run the 'test-assignment7b.py' mujoco script. Take a screenshot of the manipulator for 5 different instances of running the code. Each time you run the script, it will randomly generate a desired position and run an inverse kinematics solver to move the end-effector to that position. You should notice that sometimes it fails to exactly reach the desired position (shown in the simulation by a red sphere). Report how many times the solver failed to reach the desired position, and how many times it succeeded.

   (b) (1 point) Look at the code in inverse_kinematics.py (within the Example-MuJoCo repository) and write down (as a mathematical expression, not in words or code) the optimization problem that is being solved.

4. **LAB COMPONENT:** This week we will be completing the inverse kinematics module for lab.

   For submission, please show your final simulation to one of the TAs during lab office hours. If this is not possible, you may also email a video to Prof. Tucker with the TAs cc-ed. **For full credit, each member of each group must submit their own simulation video.**