Lab 2. Two-link arm ball toss competition.

Deliverables: **Each group must submit a *short*, Lab Report that includes required plots and calculations for both Lab 1 and Lab 2. Please be sure you SAVE all data throughout the course of the lab, and also measure the distance from the base of the arm to the location (e.g., the cup) where the ball typically lands**.

Overview

The goal of this lab is to control a two-link arm to toss a ping pong ball into a cup. This requires:

1. Designing a controller, with feedback and feedforward components.
2. Recording a step response, to compare the closed-loop system response with the predicted closed-loop dynamics.
3. Planning (or revising) trajectories for each of the motors, such that a ball would be tossed with a desired ballistic trajectory.
4. Observing the repeatability of your controlled ball-tossing system.

As always, you are encouraged to explore various solutions, depending on your personal knowledge, interests, curiosity, and persistence.

Tasks

1. Design a controller, Cs = Kp (just a gain) that will give about 50% overshoot for the plant: `P = tf([11.3],[.25 1 0])`

   For example, you can use:
   ```
   Pc = feedback(Cs*P,1)
   ```
   to create a closed-loop transfer function, using unity feedback.

   To test a step response, use in MATLAB:
   ```
   step(Pc)
   ```

2. Convert both plant, P(s), and controller, C(s), to discrete-time transfer functions:
   ```
   Pz = c2d(P,.001,'zoh')
   Cz = c2d(Cs,.001,'tustin')
   ```
   and test that the discrete-time control still works as expected, e.g.,
   ```
   Pzc = feedback(Cz*Pz,1)
   ```

3. Record a step response for each motor, and compare with theory. SAVE YOUR DATA. If there is steady state error, add in an integral term to the controller, to take the form:
   ```
   Cs = Kp + Ki*(1/s)
   ```
   Tune both Kp and Ki as desired for "good" performance. SAVE DATA for your final controller, for a step in motorA (with motorB=0) and for a step in motorB (with motorA=0). Note that Kp and Ki might be different for motorA vs for motorB…

   To create a step trajectory of 30 degrees in just a1, you may do the following:
   ```
   a1 = zeros(1,500);  a1(101:end) = 30*pi/180;  a2 = zeros(1,500);
   ```
   For a step in a2, just swap values in a1 and a2. Note that a1 and a2 will be auto-scaled to be completed in 0.5 seconds, regardless of how long each vector is. (Also, ask TA's about the feedforward that is included in the controller, if you have time and are curious.)

4. Create trajectories a1 and a2 to toss the ball. (Hopefully, you can use problem 1 from HW2, at least as a good starting point.) Note the angle offsets shown in the attached diagram! Your encoders will ALWAYS start at "zero", so you may need to add appropriate offsets to a1 and a2, such that the geometry shown is assumed.

5. Finally, run 5 trials in a row with your cup set as desired. You MUST tell the TA when your official "trials" are to begin in order to have them count. Your score for our contest will be the number of successful times the ball lands in the cup times the distance from the base of the Lego arm to the cup, measured in millimeters. The team with the highest score will receive extra credit in the class. SAVE DATA for at least one trial, to compare planned and actual trajectories over time.

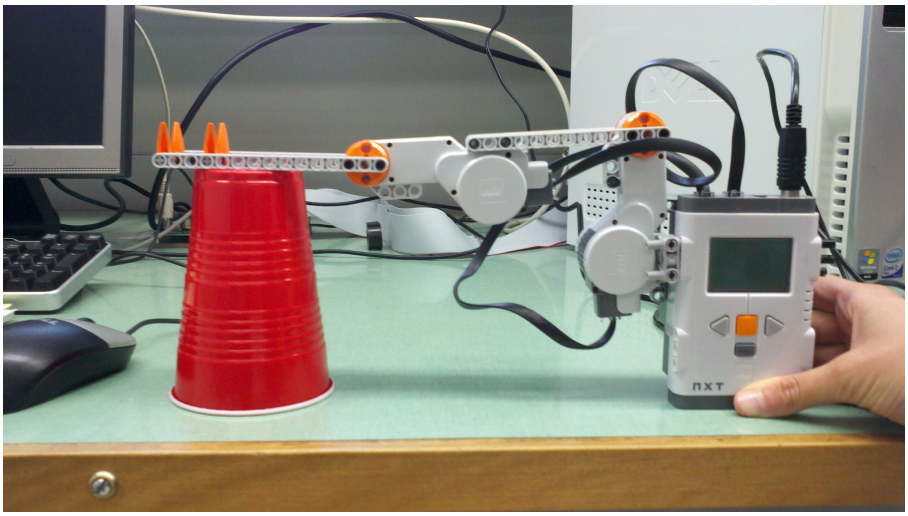

Fig. 1: Suggested starting configuration for ball toss, with link 2 horizontal.

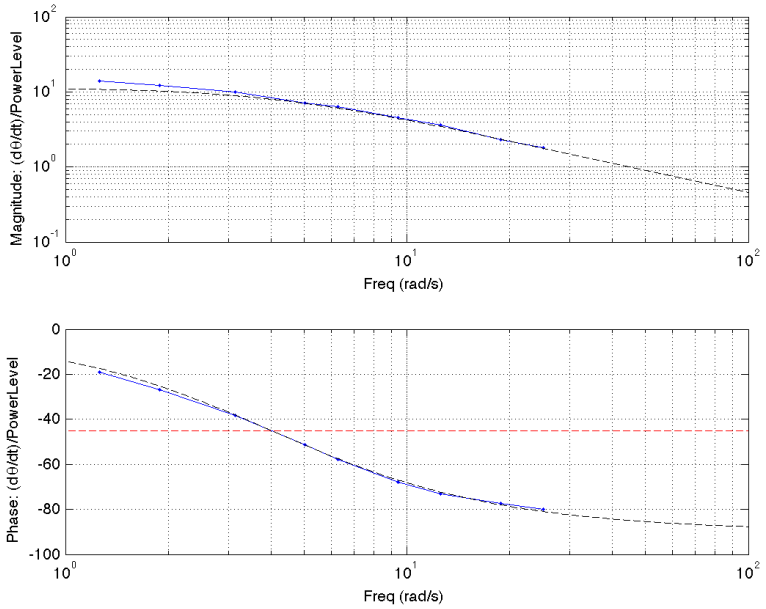

Fig. 2: Experimental Bode plot of open-loop plant, from Power Level to motor Angle (in deg)

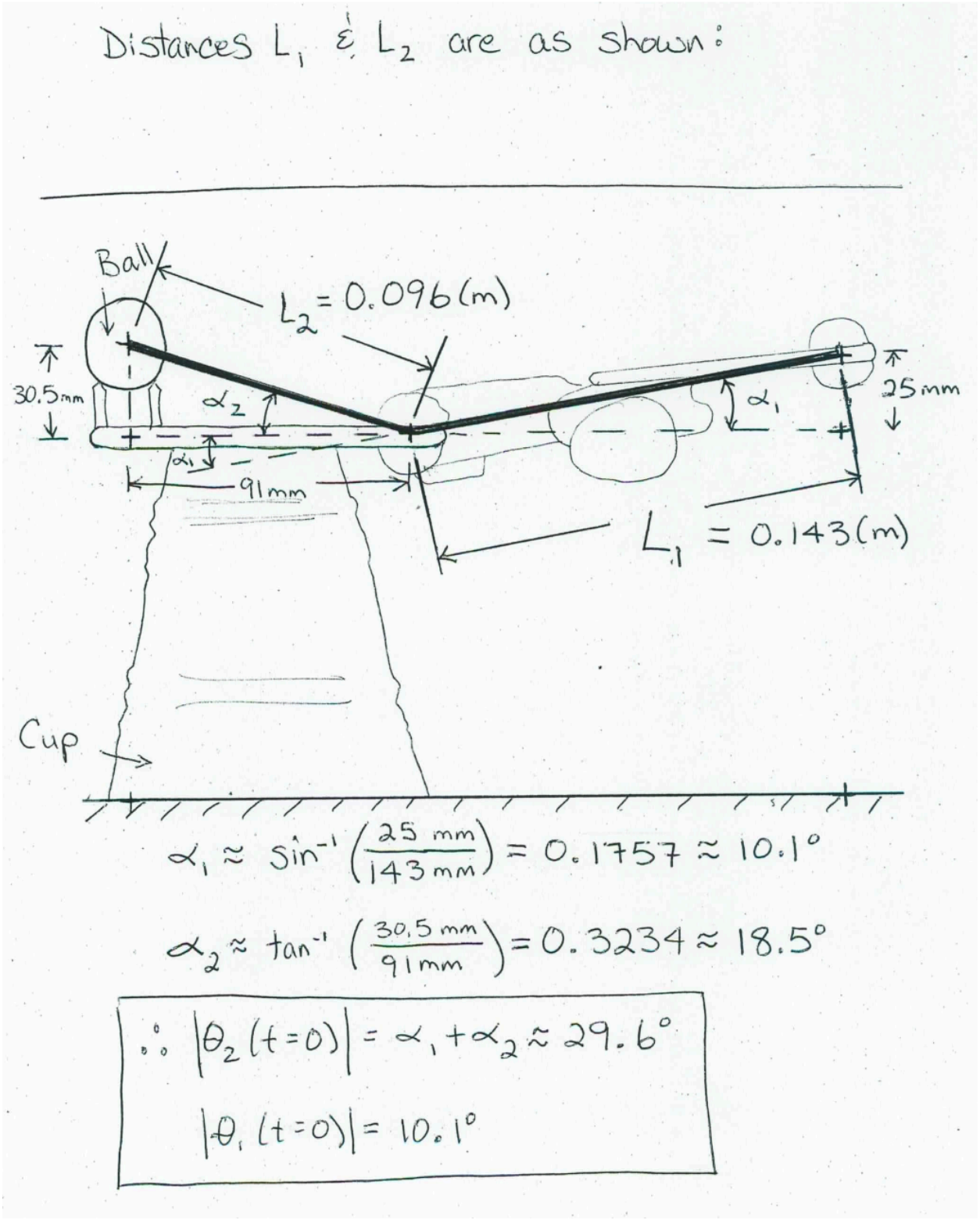

Fig. 3: Diagram of approximate geometry for an initial condition as shown in Fig. 1.