Lab 1A. 2-Link Arm: Draw a triangle with a laser pointer

In this lab, you should modify your Simulink model from Prelab 1A, so that you send signals to and read encoder values from the two motors in the 2-link arm, instead of the "Simulated Lego Motor" blocks. Be sure the rest of model is configured to run on the LEGO EV3 bricks, as described in Lab 0. You and your partner(s) can begin with any of your respective solutions, as you wish. You have the opportunity to test each of your designed K_p and K_d values, causal velocity filters, and/or reference trajectory sets, if you wish, by modifying the simulink model appropriately. To complete the lab, however, you are only required to demonstrate the following for one particular Simulink model:

1. Use MATLAB to plot encoder data for a step response for one motor under PD control; use a step in reference of magnitude 0.4 radians (about 23°). Also use MATLAB to overlay a theoretical step response, which should match (approximately), based on your choice of K_p and K_d.

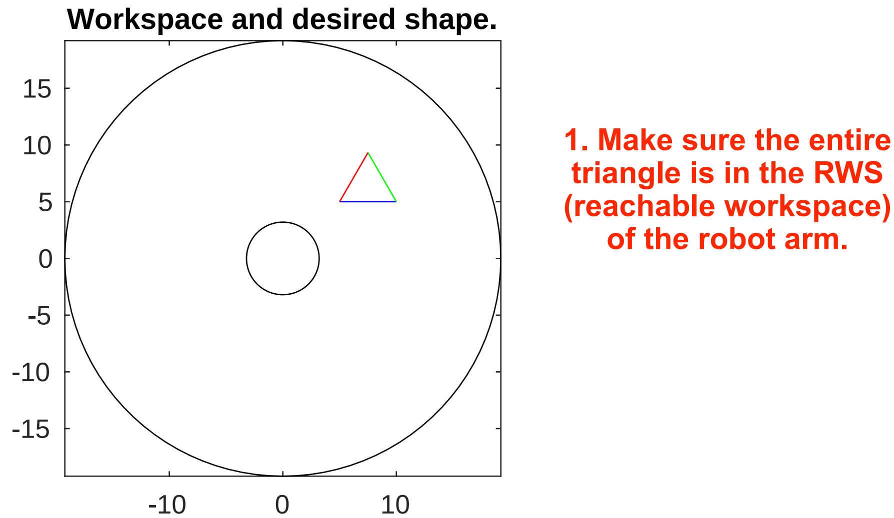
2 Record a video (e.g., with a cell phone) of the arm tracing a triangle on the wall with the laser pointer turned on. You should position a piece of paper with the desired triangle shape on the wall, so that the laser path aligns with the shape.

3. Produce a plot of the (x_{ee}, y_{ee}) trajectory that was followed, based on encoder data, using MATLAB.

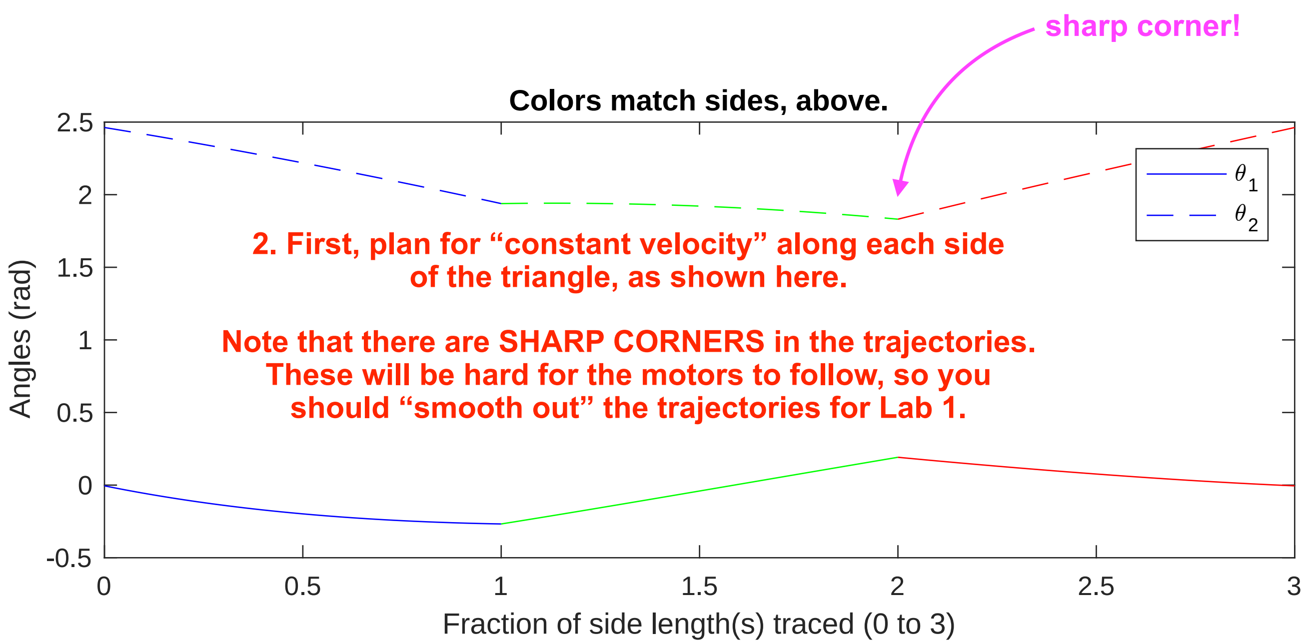
4. Submit pdf-format files with your MATLAB figures (from 1. and 3.), along with your video (from 2.), on GauchoSpace. (e.g., use "print -dpdf Figure1" to create Figure1.pdf from the current figure in MATLAB.) Please plan to show these three items to the TA during lab, to be sure your work looks fine. Only one turn-in is required per group.

Here are some tips and suggestions:

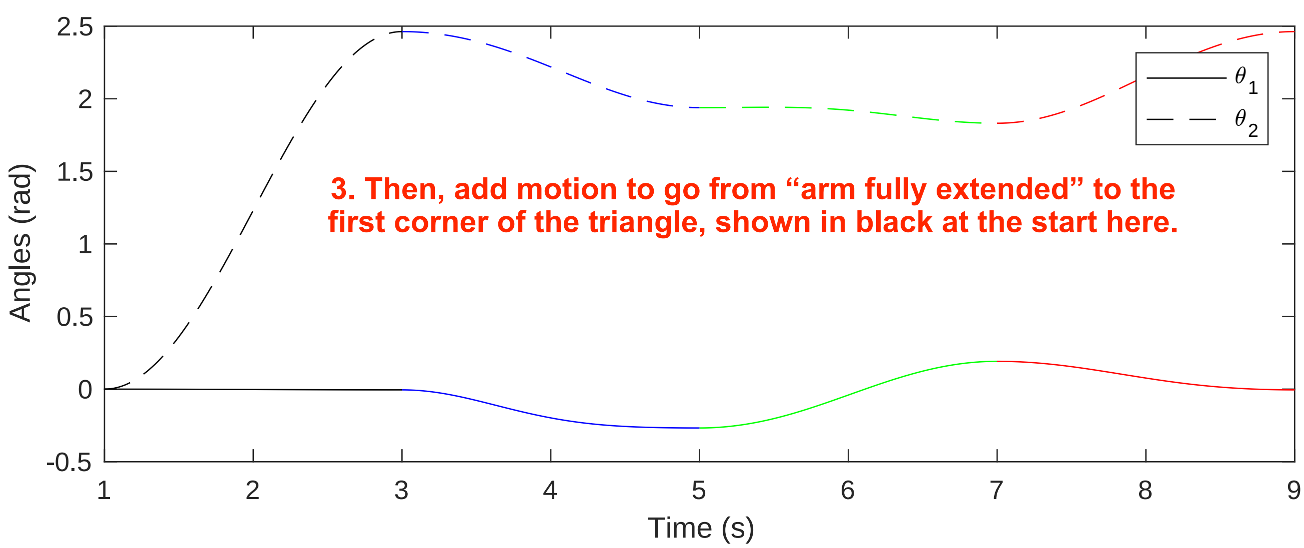
- Make sure your causal velocity filter seems reasonable. Record data and plot both θ and $\dot{\theta}$ for each motor, and compare the $\dot{\theta}$ estimates versus the slope of the $\theta(t)$, at various times. They should be approximately the same (as this is the entire goal of the velocity estimate).
- Test a step response for a single motor, to help set K_p and K_d. You can create an example step response for a 2nd-order system in MATLAB, to see if your expected ζ and ω_n match the true step response. For example, modify the following to match your planned ζ and ω_n:
  `zeta = 1; wn = 10; G = tf([wn*wn],[1 2*zeta*wn wn*wn]); step(G)`
- Modify your angle trajectories at the start as follows. Pick a starting orientation for the arm: for example, with both links exactly "straight out". Each encoder will always start at a reading of zero, when your model runs on LEGO EV3 brick. The arms needs to move from a configuration with $\theta_1 = 0$ and $\theta_2 = 0$ to the starting location on your triangle itself.
- Adjust the speed of the end effector along the triangular path, to avoid "sharp corners" in the joint reference trajectories. E.g., for each side, start at zero velocity, speed up, and arrive at the next triangle corner with zero velocity. The next page shows an example of this.
- Check magnitudes and signs of both your reference and actual motor angles! Which rotational direction is "positive" for each motor? Also, be sure angles are always measured in radians. (K_p and K_d should have been designed assuming radians for angles and (rad/s) for velocity.)



Workspace and desired shape.
1. Make sure the entire triangle is in the RWS (reachable workspace) of the robot arm.



sharp corner!
Colors match sides, above.
2. First, plan for “constant velocity” along each side of the triangle, as shown here.
Note that there are SHARP CORNERS in the trajectories. These will be hard for the motors to follow, so you should “smooth out” the trajectories for Lab 1.
Angles (rad)
Fraction of side length(s) traced (0 to 3)
θ_1
θ_2



3. Then, add motion to go from “arm fully extended” to the first corner of the triangle, shown in black at the start here.
Angles (rad)
Time (s)
θ_1
θ_2