LAB ASSIGNMENT #2: 2-DOF MOTION CONTROL

ME EN 5230/6230, CS 6330

Intro to Robot Control – Spring 2023

Introduction

The purpose of this assignment is to implement coordinated motion control on a real robot manipulator. We will use the Quanser 2-DOF serial robots, shown in Figure 1, whose dynamics we simulated in Problem Set #5. Each robot is formed by connecting two of the SRV-02, which should already be assembled together for you. The robot should be oriented on the desk as shown in Figure 1, with the module controlling joint 1 on the right, and the module controlling joint 2 on the left. Since joint 2 is driven by a belt transmission, we will control the robot using absolute joint angles rather than relative joint angles, just as we simulated in Problem Set #5. For convenience, the "Home" position of the robot will be defined as shown with Link 1 oriented straight up (ϕ_1=90°), and Link 2 oriented straight out towards the front of the desk (ϕ_2=0°). The cables for controlling the robot should be connected as shown in Figure 2. **Please double-check all your cable connections and read the entire lab protocol carefully before beginning your experiments.**

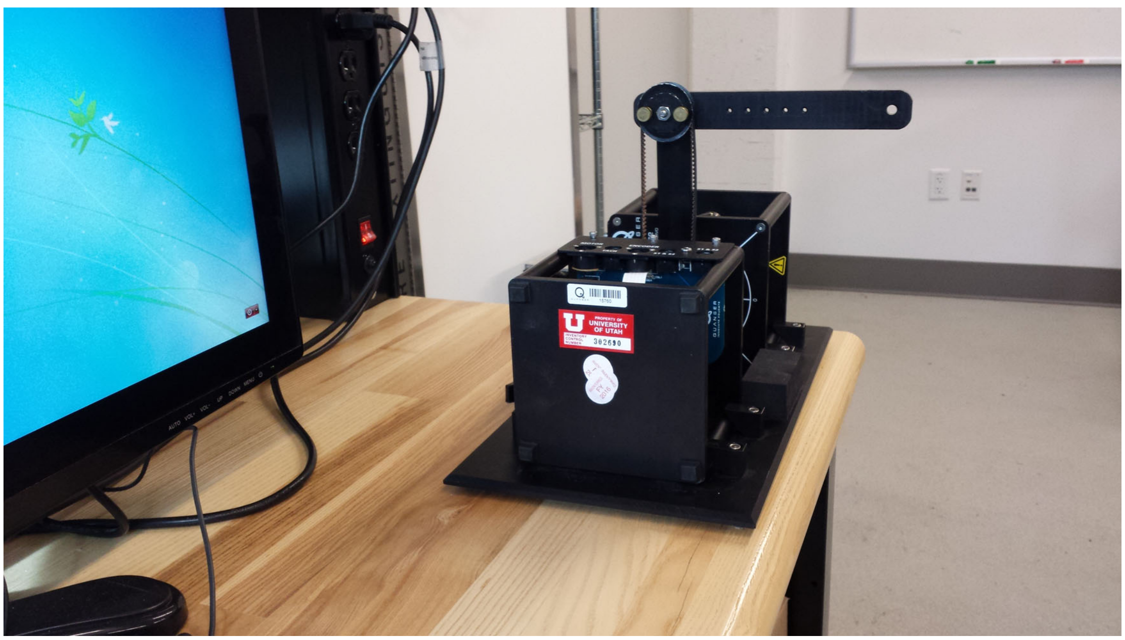

Figure 1. Quanser 2-DOF serial robot in the "Home" position

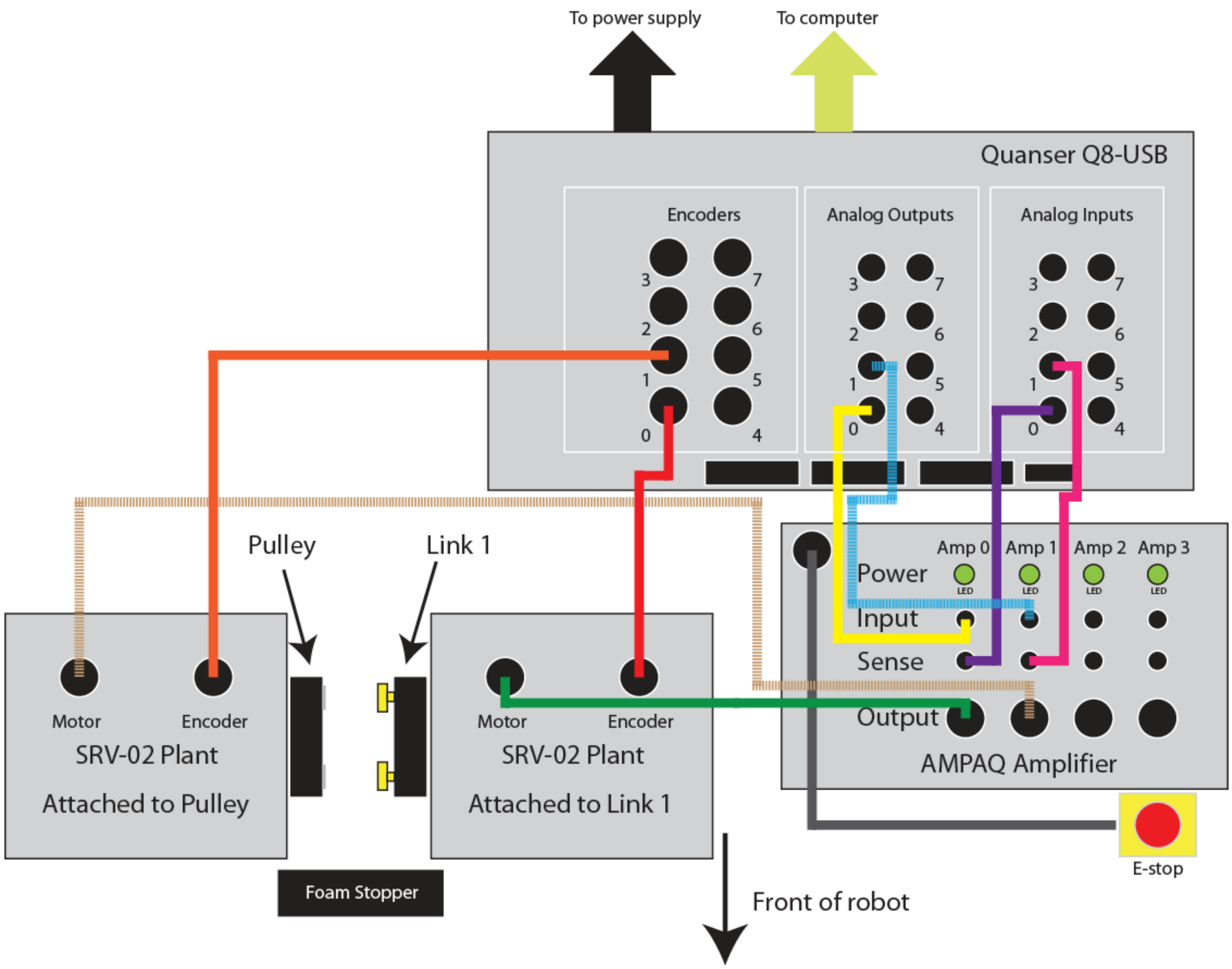

Figure 2. Experimental Setup for 2 DOF system

Dynamics

The dynamic equations are the same as given in PS#5:

$$\begin{bmatrix} N_1 k_{t_1} i_1 \\ N_2 k_{t_2} i_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \ddot{\phi}_1 \\ \ddot{\phi}_2 \end{bmatrix} + \begin{bmatrix} 0 & -h \\ h & 0 \end{bmatrix} \begin{bmatrix} \dot{\phi}_1^2 \\ \dot{\phi}_2^2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} + \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$$

where ϕ_1 and ϕ_2 are the absolute joint angles (measured by the encoders), i_1 and i_2 are the motor currents, and:

$$H_{11} = N_1^2 J_{m1} + I_1 + m_2 a_1^2$$
$$H_{12} = H_{21} = a_1 r_{12} m_2 \cos(\phi_2 - \phi_1)$$
$$H_{22} = N_2^2 J_{m2} + I_2$$

$$h = a_1 r_{12} m_2 \sin(\phi_2 - \phi_1)$$
$$G_1 = (r_{01} m_1 + a_1 m_2) g \cos(\phi_1)$$
$$G_2 = r_{12} m_2 g \cos(\phi_2)$$
$$F_1 = N_1^2 b_1 \dot{\phi}_1 + N_1 c_1 sgn(\dot{\phi}_1)$$
$$F_2 = N_2^2 b_2 \dot{\phi}_2 + N_2 c_2 sgn(\dot{\phi}_2)$$

Assume that the parameters are known to be:

Joint/Link: *i*	**1**	**2**
Link length: a_i	0.150 m	0.150 m
Link mass: m_i	0.092 kg	0.077 kg
Link center of mass: $r_{i,i+1}$	0.062 m	0.036 m
Link moment of inertia: I_i	0.64×10^{-3} kg·m^2	0.30×10^{-3} kg·m^2
Motor inertia: J_{mi}	0.65×10^{-6} kg·m^2	0.65×10^{-6} kg·m^2
Viscous damping constant: b_i	3.1×10^{-6} N·m/(rad/s)	3.1×10^{-6} N·m/(rad/s)
Coulomb friction constant: c_i	0.1×10^{-3} N·m	0.1×10^{-3} N·m
Motor Torque Constant: k_{t_i}	0.0077 N·m/A	0.0077 N·m/A
Gear Ratio: N_i	70	70

Lab Protocol: PLEASE READ CAREFULLY!

Buddy System:

You should conduct this assignment with a partner (you can form a team of 3 if necessary, but teams of 2 are ideal). Each team member is required to turn in a separate write-up. You may share data, but the analysis and plots of data in your write-up should be your own. Do not operate the robot without a "buddy" present.

Safe Robot Operation:

Each amplifier is equipped with a tethered emergency stop button (E-stop). Before you begin, don't forget to turn on the amplifier (power switch on the back) and make sure the E-stop is not depressed (twist button to release). The LED lights above each channel on the amplifier should now be lit, indicating the channels are enabled to receive power. Use caution when operating the robot, especially at high speeds. Keep the E-stop in easy reach, and press it immediately it if something wrong happens during experiments to prevent mechanical damage to the robotic system as well as for your safety. Only operate the robot with a "buddy" present. One person must be responsible for operating the emergency stop while the other person is responsible for operating the controller. Before you conduct your experiments, please make sure there are no obstructions in the robot workspace. Also check the tightness on the thumbscrews and setscrews that hold the robot assembly together. Be aware that an unstable experiment may cause the screws to loosen, so check them periodically. If any of the hardware appears damaged, please notify the instructor and post a notice on the Canvas discussion board.

Simulink Tips:

Don't forget, when using Simulink to control real hardware, you need to run your models in ***External Mode*** as you did in Lab #1 (using the Run on Hardware commands in the Hardware tab in Simulink). Alternatively, you can use the *RobotApp* program that is posted on Canvas to operate your Simulink models in real time. If the GUI is too sluggish, you can increase the sample time in the Phi scope block, which only affects how often the GUI is refreshed. (Note that this is different than the *fundamental sample time* for the model, which is set in the Model Configuration Parameters. The *fundamental sample time* should be kept at 0.001 or 0.002 to ensure smooth control).

Hardware Gains, Sign Conventions & Amp Biases:

Since the SRV-02s are positioned facing each other, they will naturally move in opposite directions when given positive steps. We have already compensated for this in the provided Simulink template by using negative amp and encoder gains on joint 1, which assumes the robot is oriented on the benchtop as shown in Figure 1 and wired as shown in Figure 2. We have also programmed the template with encoder offset values that account for the fact that the home position is not the zero-angle position. The amp gain has been accounted for in your Simulink template, but don't forget to use the motor gain to convert your command signals from torques to motor current when appropriate. The Simulink template also has blocks to compensate for amp bias, but you will need to set the bias values yourself. First, see if there is any bias on the current sense by making sure the sensed current is zero when the amplifier is disabled (E-stop is pressed). If there is a bias, you can remove it by adjust the bias block on the sensed current. Second, determine if there is a bias on the current command. To do this, re-enable the amplifier and send a command of zero to the motors and look at the sensed currents. If there is a bias, you can remove it by adjusting the bias block on the current command. If the bias is severe, you can also adjust the initial/final conditions on the Analog Outputs in your HIL_Initialize block (see the QUARC Tutorial) to remove the bias even when the model is not running, which makes it easier to home the robot (see next section).

Homing the Robot:

Since we are using the encoders for position feedback, the joint angles are always measured relative to where the robot is when the controller is activated. Therefore you will need to manually "home" the robot prior to each trial, by orienting the robot base and links as shown in Figure 1. The robot should be oriented on the benchtop with joint 1 motor on the right and joint 2 motor on the left, such that the designated home position is towards the front of the bench. Link 1 should be oriented straight up (ϕ_1=90°), and Link 2 oriented straight out towards the front of the desk (ϕ_2=0°). The parameters in your Simulink template (see below) have been adjusted according to this designated configuration. If you have adjusted your Simulink model to remove any amp bias, then the friction in the joints should be sufficient for the robot to rest in the home position until you run your controller. Since the home position is not on the circle, the desired trajectory has been designed so that the robot will smoothly move to a point on the circle before beginning to circle. **The robot must be homed prior to each trial, or the robot will likely crash into the benchtop!**

Final Tips:

Whenever you implement a new controller, make sure you first try it out at low speed. Be aware that the AMPAQ amplifiers are hardwired to saturate at 1 Amp of current in order to protect the motors from burning out. As you implement your various controllers, keep an eye on your motor current commands. If your controller saturates the amplifier at low speed, it may be a warning sign that you should decrease your PD gains. If your controller saturates the amp at high speed, it may just be a result of the large tracking error. You should ideally be able to use the same PD gains as in PS#5, but it is it always a good idea to start with low PD gains and work your way up. The real robot may have unmodeled dynamics that lead to instability at higher PD gains.

Lab Exercises

On the course website, you are provided with a Simulink template <Lab2_template2023.slx>, which has the necessary motor outputs and sensor inputs built into the *Robot subsystem*. The desired trajectory and inverse kinematics is also programmed into the template for you. The desired trajectory is a circle in Cartesian coordinates. Using the inverse kinematics, this is converted to a desired trajectory in joint space. The trajectory has been modified slightly from PS#5 to facilitate "homing" of the robot (see above). Your task is to design and implement a series of joint space controllers to track this trajectory:

1. **Decentralized Control**
   - **1.1** PD Control only
   - **1.2** PD Control with gravity, friction, and centripetal/coriolis feedback compensation
   - **1.3** PD Control with Computed Torque Feedforward Control
2. **Centralized Control:** PD Control with Inverse Dynamics Control

For each control scheme, run the robot at low speed (0.2 circles/s) and high speed (1 circles/s) and compare the tracking performance with the other control schemes, just as you did in PS#5. For each part, you should ideally be able to use the same PD gains and compensators that you used in the corresponding parts of PS #5.

Requirements for your analysis/writeup (same as in PS#5):
For each control scheme, provide an image of your Simulink model and the code from your Embedded MATLAB Functions. For each simulation, make a plot of the x-y trajectory along with the desired trajectory and a plot of the joint angle errors. Be sure to properly title your plots and label your axes. (To do this, you'll want to send the x-y coordinates and joint data to the workspace.) Include the code from any MATLAB scripts you use to analyze/plot the data. When comparing the tracking performance of different controllers, use both peak joint errors and the RMS (Root-Mean-Square) of the joint errors as metrics. For a fair comparison, be sure you use the same Model Settings (solver, step time, time duration). It is recommended that you use an application like MS Word to compile your writeup, in which case you should copy your Simulink models and MATLAB plots as Metafiles in order to make high-resolution figures.