AERSP304 Funduino Project
Assignment 2

Due: Date

In the previous assignment, you have identified a transfer function corresponding to your motor. In this assignment, you will analyze this transfer function to design a motor speed controller to run the motor at a desired RPM.

1 Study of the System

This section lists the task related to Matlab Project 3. First, we will study some open-loop and closed-loop response of the system followed by design of a PID (Proportional-Integral-Derivative) controller to meet specified requirements.

Considering that each group may have different motor, the reference speed input will always be consider as $\omega_{ref} = 80\%$ of the nominal motor speed given in RPM. For example, if the nominal speed of the motor is 330 RPM, consider something around 264 RPM as the desired input.

1. **Open-Loop Response**: Compute the poles and zeros of the open loop transfer function and output response of the speed of the motor for a step input of ω_{ref}. Comment on the output response. Some useful Matlab commands here: pole, zero, tf2zp, step, lsim.

2. **Closed-loop Response**: Assuming unity feedback, i.e., $\delta_e = K(\omega - \omega_{des})$; $K = 1$, compute the closed loop transfer function while making use of *feedback* command in Matlab. Compute the response of the closed loop system for input of ω_{ref} while making use of step command in the Matlab. Find rise time, settling time and final value of the response. (Note: you can select system response characteristics (such as rise time, settling time, final value) by selecting them from the right click menu under characteristics on figure generated by step command in Matlab).

3. **Root Locus**: In Part 3), we assumed the feedback gain to be unity. We will examine the effect of varying the value of feedback gain K. Use the Matlab command *rltool* to generate the root locus corresponding to the closed-loop plant. Since our desired signal value is ω_{ref}, you should set the pre-compensator block equal to ω_{ref} in the *rltool* window. Right click on the root-locus plot and select design requirement from the resulting menu to add following requirements:

    - Settling Time less than 5 seconds.
    - Overshoot less than 10%.

Examine the resulting figure to see if a proportional controller meet the aforementioned performance criterion. (Note: For discrete transfer functions, MATLAB, automatically show you unit circle for stability and correct curves for settling time and overshoot.)

4. **PID Controller**: Now change your settling time to be something much less than that one used in part 4). You can choose a new value for your settling time but make sure that there will be no overshoot. Design a PID controller with those requirements. You can use the PID tuning menu on the **rltool** command GUI. Alternatively, you can also use the Control System Design and the PID Control Design GUIs in Matlab too. Some tutorials on how to use these tools can be found in the links below.

   - Control System Design video tutorial
   - PID Control Design video tutorial

   After you designed your controller you can easily simulate the response of the closed loop system with the *feedback* command in Matlab. Supposed that your transfer function is given by the *tf* command and it is stored in the *sys* variable. Then you have designed a PID controller with the PID Control Design tool (or any other tool) and exported it to the workspace with variable named *PIDtf*. So you can easily simulate the step response by doing the following:

   ```
   % sys is a TF
   % c is a PID TF
   sys_closed = feedback(c*sys,1)
   step(sys_closed)
   ```

   Comment on the output response.

5. Use the *tf2ss* command on controller transfer function to get a time domain expression for your controller. Compare your MATLAB simulation results with your actual motor output.

2 Report and Video

Along with a report explaining everything you have done in this second part of the project, we also ask that you produce a video (at max 10 minutes) demonstrating your work and explaining the process as though you were giving a presentation. We ask that in the written report you include:

- Your identified transfer function
- Your open-loop analyses
- Your closed-loop analyses with unity feedback
- Your response in time considering the closed-loop analyses and the unity feedback
- Your root locus analyses
- An explanation on your PID controller design process, what is the final controller transfer function and the expected output (plot of the step response)

- The transfer function for your plant with the controller
- Analyses of the implementation and the tests with experimental set up. Did the motor achieve the desired requirements in practice?

The video should focus more on the implementation and control of your motor, but also have a brief explanation of the above items that are relevant to what you are demonstrating.

3 Useful Commands and Links

Always check the help via **help** $< \ command \ >$ or the html version of the help via **doc** $< command >$ before you use a command.

- **tf** - Generates a transfer function from given numerator and denominator
- **tfdata** - Get the parameters of a transfer function. The inverse of tf
- **zpkdata** - Get zeros, poles and gain of a transfer function
- **c2d** - Convert a transfer function from continuous to discrete time.
- **d2c** - Convert from discrete to continuous time.
- **pole** - Calculate the poles of a transfer function.
- **zero** - Calculate the zeros of a transfer function.
- **damp** - Calculate the natural frequency and damping.
- **pzmap** - Plot pole/zero map.
- **bode** - Plot the bode plot for the system.
- **impulse** - Calculate the impulse response of a system.
- **step** - Calculate the step response of a system.
- **lsim** - Calculate the response of a system to an arbitrary input.
- **rlocus** - Calculate the root locus plot for a system
- **rltool** - GUI for root locus design
- **rlocfind** - Find gain and pole on root locus.
- **sgrid** - Plot the s-plane grid on root locus or pole/zero maps.
- **stepinfo** - Produces many of your system characteristics, such as overshoot, settling time, and more.
- **ss** - https://www.mathworks.com/help/control/ref/ss.html
- **ss2tf** - https://www.mathworks.com/help/matlab/ref/ss2tf.html
- **tf2zp** - https://www.mathworks.com/help/signal/ref/tf2zp.html
- **feedback** - https://www.mathworks.com/help/control/ref/feedback.html
- **residue** - https://www.mathworks.com/help/matlab/ref/residue.html
- Root Locus Design - https://www.mathworks.com/help/control/ug/root-locus-design.html
- PID Control - http://ctms.engin.umich.edu/CTMS/index.php?example=Introduction&section=ControlPID