Hard disk drive control

The positioning of the heads for writing and reading on a magnetic disk is a challenging control problem. The entire head assembly is driven by a single electric motor.

The transfer function of the positioning system including a single resonant mode is given by:

$$G(s) = \frac{(2\zeta\omega s + \omega^2)}{s^2\left(s^2 + 2\zeta\omega s + \omega^2\right)}$$

with the following parameters:

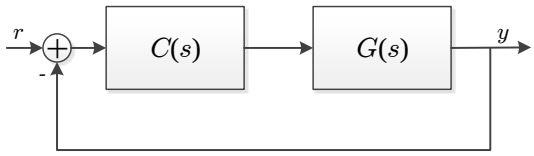
$$\zeta = 0.1$$
$$\omega = 1000\frac{\text{rad}}{\text{s}}$$

This fourth order transfer function will be used for the design and evaluation of a controller.

Questions

Continuous-time control

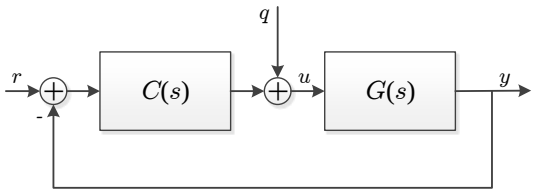
1. First, the system needs to be controlled with a feedback controller in continuous-time.



   We are aiming to control the system in such a way that a reference set-point step has a fast response (i.e., minimal settling time in which the final output value is reached to at least 1% tolerance), and the response has an overshoot of maximum 5% with no steady-state error. Find the most suitable controller structure (from among P, PI, PD, PDD, PID, etc.) and tune it to achieve the following control objectives for a set-point step:

   (a) minimal settling-time
   (b) overshoot $< 5\%$
   (c) steady-state error $= 0$

2. Instead of the reference set-point change, a step is added to the input of the system as a load disturbance q (thus $r = 0$).



   The control objective is to reduce and eliminate the effect of this disturbance as well as possible. This means:

   (a) minimal amplitude of the system output y caused by disturbance q
   (b) minimal duration of the disturbance (i.e., time it takes to reduce to within 1% of its maximal value)
   (c) no offset caused by the disturbance

   What is in this situation the best choice and tuning for a controller (P, PI, PD, PDD, PID, etc.)? Design such a controller and compare your result with that of the previous point.

Please make sure that you design a physically realizable controller (i.e., not a text-book version with ideal D term). Furthermore, completely automated, brute force, or random search for the controller parameters will not be appreciated.

Discrete-time control

3. First, describe the system in the state-space notation:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

   Transform the continuous-time state-space model into the following discrete-time description by selecting an appropriate sampling time h:

$$x_{k+1} = \Phi x_k + \Gamma u_k$$
$$y_k = Cx_k + Du_k$$

   Motivate your choice of discretization method and sampling time.

The rest of the questions will have to be solved with *discrete-time controllers* for the *discrete-time system*. Adjust the sampling period wherever necessary.

4. Put the earlier found continuous-time controllers into a corresponding discrete-time algorithm using a properly chosen sampling time. Motivate your choice of discretization method and explain its advantages and disadvantages. Evaluate a set-point step and disturbance step for the controlled discrete-time system. Compare the results with the continuous-time results.

5. Apply a discrete-time pole placement method to control the system with state feedback. Take different sets of pole locations for the controlled system and construct a servo (tracking) controller using your state feedback design. Evaluate the set-point step response of your tracking scheme. For your final design, place the poles of the system in such a way that the control objectives are achieved as well as possible.

6. Add a discrete-time dynamic observer to estimate the states of the system. Use the same pole locations for the control as in the previous question and place the poles of the observer separately. Where are the poles of the observer placed? Compare the responses of the system controlled with an observer and without an observer using a set-point step (servo tracking scheme). When comparing the two schemes, use nonzero initial conditions in order to investigate the estimation error. Show that the estimation error converges to zero over time. Assume a step disturbance on the plant input and augment your control design with integral action if needed. Provide a revised set of poles for the augmented system. Evaluate the disturbance responses of your observer-based (output feedback) pole-placement design with integral action.

7. Apply now a discrete-time LQ controller for the optimal control of the system using different weighting matrix selections. What is the influence of the different weighting factors? What can you conclude from the results of a step response (servo scheme) regarding the choice of different weighting matrices?

One of the more practical aspects of control is that in many cases the physical components of the real system (actuators) will limit the performance of the controllers. For this reason, you will investigate the effects of limiting the magnitude of the control signal on the performance of the control loop.

8. Calculate the input signal sent to the system by all previously designed controllers during the execution of the set-point step and during the disturbance step. In the following questions we will limit the controller output value to lie within the range ± 200. *If your original controllers did not saturate the specified limit, they likely do not meet the original specifications.*

9. Redesign and retune your discrete-time PID-type controller in order to investigate the best performance you can still achieve while avoiding control signal saturation for set-point and disturbance responses both.

10. Consider now the tuning of your discrete-time pole placement controller within the above limits for the controller output, and revise to avoid saturation. Again, study the set point step (servo scheme) of the controlled system (do it only for the state-feedback case).

11. Finally, retune the discrete-time LQ controller such that the controller output is within the specified limits. Again, evaluate the set point step response (servo scheme) of the controlled system.

Sometimes a steady state error may occur in applying feedback control.

12. Give a method for each of the above control approaches to eliminate a steady state error completely and show with a reference step response and a step disturbance on the input that the steady state error can be reduced completely back to zero (if you have addressed this issue in earlier questions, provide a brief mention here for our reference).

Typically, one extra time step delay is introduced by the computation of the control algorithm.

13. How will one extra time step delay (on the control input) influence the behaviour of the controlled system? Compare the step responses of set-point and disturbance including the extra delay with the previously obtained results. *If your results barely change, this can be an indication that you chose a too small sampling time.* How could this extra time step delay be modeled and incorporated in all your previous controller designs (i.e., both transfer function and state-space methods)? Redesign your controllers and show the responses of the modified control system to a set-point step and to a disturbance step on the input of the system.

What are your overall conclusions for the control of the heads of the hard disk system?

Useful Matlab functions:

`ss2tf`	transforms the state-space form into the transfer function
`tf2ss`	transforms the transfer function into the controllable canonical state-space form
`ltiview`	to analyse system step response, Bode- and Nyquist plots
`rltool`	to analyse quickly the closed-loop behaviour of a system
`c2d`	transforms a continuous-time system for a given sampling time into a discrete-time description
`step`, `dstep`	calculates the response of the system to a unit step on the input (`dstep` can only be used for discrete-time systems)
`place`, `acker`	calculates the control law that places the poles of the system at indicated positions
`dlqr`	linear quadratic control in discrete-time
`stairs`	stairstep plot, useful for drawing time history plots of zero-order-hold digital sampled-data systems

With the MATLAB help function all the details of the mentioned functions can be found directly.