Automatic Control – Basic Course

Laboratory Exercise 3

Control of a Flexible Servo

Department of Automatic Control
Lund University
2012

1. Introduction

In previous labs we have studied a process, which has been relatively simple. We have been able to control it satisfactory using a simple PID-controller. In this lab we will examine a process which is a bit more complicated and which requires a more advanced controller. The controller which we will use is based on state feedback and state estimation.

Preparations

- Read this manual carefully.
- Review the lectures on state feedback, Kalman filtering, and output feedback. At the beginning of the lab you should be able to answer the following questions:
  - What does state feedback mean? Explain in words!
  - Why is an observer often used in connection with state feedback? Explain in words!
  - Draw a block diagram that shows how an observer can be used in connection with state feedback.
- Solve the preparatory assignments 4.2, 5.2, and 6.1.
- Study the MATLAB scripts `define_process.m`, `design1.m`, `design2.m` and `design3.m` which are found in the appendix. Relate their content to the assignments in the manual.

2. The Process

The process is a simulated system consisting of two masses which are interconnected by a spring. A conceptual drawing of the process is shown in figure 1. The mass at one end of the spring can be moved by a motor. We call this end the motor end and the other end the load end.

The purpose is to control the position p_2 of the mass on the load side. In the lab we will assume that only p_2 is measurable. The remaining states will be estimated by a Kalman filter.

A Linear Model of the Process

The two masses are m_1 and m_2. The interconnecting spring has the spring constant k. The damping of the masses are d_1 and d_2, respectively.

The first mass is driven by a motor. Motor and amplifier dynamics are neglected. The force of the motor becomes proportional to the input voltage u of the amplifier according to

$$F = k_m u$$

A force balance gives the following dynamic model:

$$m_1 \frac{d^2 p_1}{dt^2} = -d_1 \frac{dp_1}{dt} - k(p_1 - p_2) + F$$

$$m_2 \frac{d^2 p_2}{dt^2} = -d_2 \frac{dp_2}{dt} + k(p_1 - p_2)$$

Introduce the state vector $x = \begin{pmatrix} p_1 & \dot{p}_1 & p_2 & \dot{p}_2 \end{pmatrix}^T$. The process can then be expressed in state space form as

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{1}$$

where

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{k}{m_1} & -\frac{d_1}{m_1} & \frac{k}{m_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k}{m_2} & 0 & -\frac{k}{m_2} & -\frac{d_2}{m_2} \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ \frac{k_m}{m_1} \\ 0 \\ 0 \end{pmatrix}$$

$$C = \begin{pmatrix} 0 & 0 & k_y & 0 \end{pmatrix}$$

The following constants and coefficients have been measured and estimated for a real lab

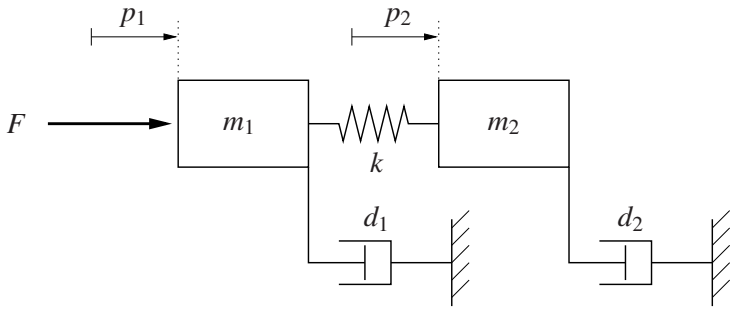


Figure 1 Conceptual drawing of the process.

process:

$$
\begin{aligned}
m_1 &= 2.29 \text{ kg} \\
m_2 &= 2.044 \text{ kg} \\
d_1 &= 3.12 \text{ N/m/s} \\
d_2 &= 3.73 \text{ N/m/s} \\
k &= 400 \text{ N/m} \\
k_m &= 2.96 \text{ N/V} \\
k_y &= 280 \text{ V/m}
\end{aligned}
$$

Analysis of the Process

Assignment 2.1 Log in (as instructed by the lab supervisor) and start MATLAB according to the instructions from your lab assistant. Define the process `Gp` by executing the script `define_process.m`:

```
>> define_process
```

Calculate the poles of the system by typing

```
>> pole(Gp)
```

Where are the poles located? Is the system stable? Asymptotically stable? Simulate the impulse response of the process:

```
>> impulse(Gp,5)
```

Does the behavior agree with the stability analysis?

Assignment 2.2 Draw the Bode plot of the process

```
>> bode(Gp)
>> grid on
```

Note the resonance peak in the amplitude curve. At what frequency is it located? What approximate relation holds between the location of the resonance peak and the locations of the poles? Estimate the natural frequency by again studying the impulse response of the process. Does it coincide with that of the model?

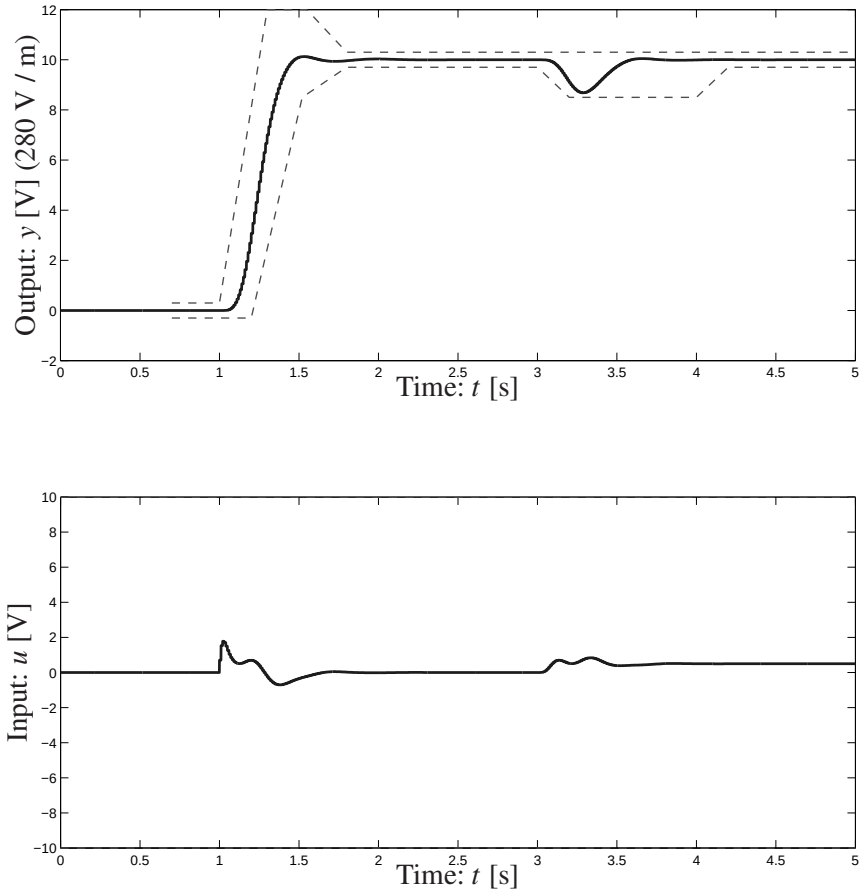


Figure 2 The step response and control signal should stay within the marked regions.

3. Performance Specifications

The performance specifications are the requirements which the controlled system should fulfill. In this case we have chosen to specify the closed loop system in the time domain. The experiment which will be used to evaluate the performance consists of a reference step at $t = 1$ s, followed by a load disturbance step at $t = 3$ s. Simulations will be made both with the nominal model described above, and a more realistic model incorprating friction and measurement noise.

We wish to have a well damped step response with a rise time between 0.2 and 0.4 seconds, see figure 2. At the same time, the magnitude of the control signal should never exceed 10 V, since this could damage the equipment.

During the first part of the lab we will focus on the step response and control signal specifications. Compensation for load disturbances and friction will be dealt with at a later point.

The oscillative properties of the process make it impossible to fulfill the specifications with a PID-controller. An attempt using a PD-controller is shown in figure 3. The step response is fast enough, but the controller cannot damp out the oscillations.

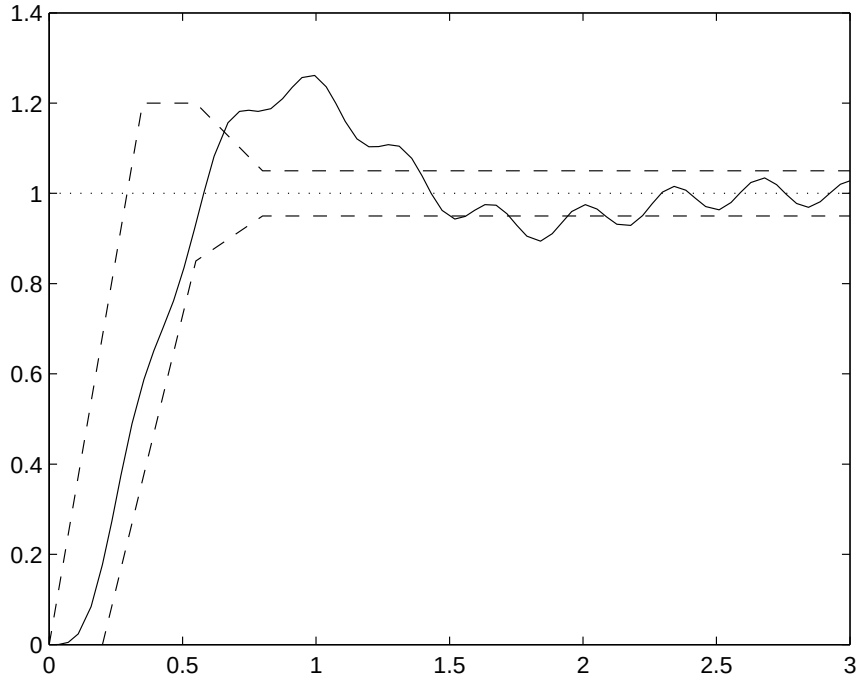


Figure 3 Reference step response using a PD controller with $K = 0.07$ and $T_d = 0.1$.

4. State Feedback

To be able to change the process dynamics arbitrarily we make use of state feedback. First we verify that the system is controllable.

Assignment 4.1 Calculate the rank (i.e. the number of linearly independent columns) of the system's controllability matrix:

```
>> Wc = [ ... ]
>> rank(Wc)
```

(Note that the matrices `A` and `B` are already defined in the workspace.)

What is the rank? Is the system controllable?

If we start out under the assumption that the entire state vector x can be measured, the following control law can be used

$$u = -Lx + l_r r \tag{2}$$

Here L is a row vector, r is the reference value and l_r is a scalar, see figure 4.

Assignment 4.2 (Preparation) Show that the closed loop system can be written in the form

$$\dot{x} = A_{cl}\,x + B_{cl}\,r$$
$$y = C_{cl}\,x$$

when the control law (2) is used on the process, (1). What are A_{cl}, B_{cl} and C_{cl}? How should l_r be chosen to obtain unit static gain from r to y? ◇

By means of state feedback we can place the poles of the closed loop system arbitrarily. However, there are practical limits on the control signal. Somewhat simplified one can say that, the further a pole is moved from its original location, the more control action will be required. In our case, we want to move the poles further into the left half plane, to make the closed-loop system fast and well-damped. For strongly oscillatory process poles, it is usually a good idea to focus on increasing their relative damping rather than changing their magnitude.

The desired pole placement can be expressed using a fourth order characteristic polynomial (see figure 5):

$$\left(s^2 + 2\zeta_a\omega_a s + \omega_a^2\right)\left(s^2 + 2\zeta_b\omega_b s + \omega_b^2\right)$$

Given a pole placement, the feedback vector L is easily computed using the command `place` (see `design1.m`). Type

```
>> help place
```

for further information on what the command does.

Assignment 4.3 Edit the script `design1.m` and insert suitable values of ω_a, ζ_a, ω_b and ζ_b. Then calculate the controller in MATLAB by typing:

```
>> design1
```

Open the Simulink model `model1.mdl` by typing

```
>> model1
```

Simulate the closed loop system and see if it fulfills the specification on the step response. This is done by selecting "Start" from the "Simulation" menu (or pressing "<Ctrl>-T"). Double click on the orange box "Plot against specifications" after a simulation to compare the results to the performance specifications. Change the design parameters and repeat the procedure until the step response and control signal specifications are fulfilled. (We will deal with load disturbance rejection later.) What is a suitable pole placement for the state feedback?

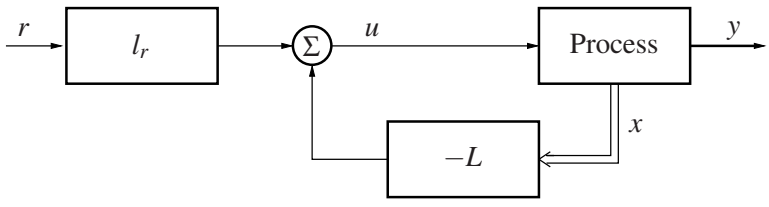


Figure 4 State feedback.

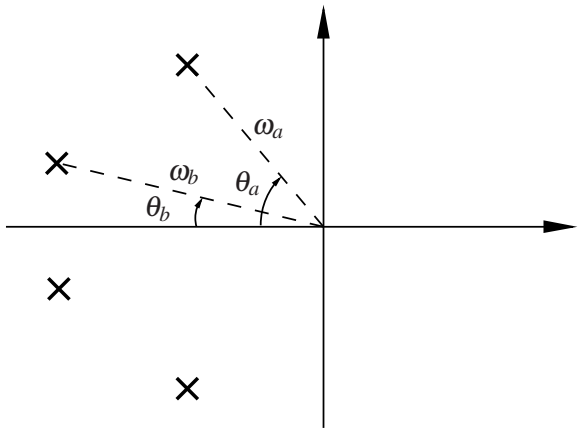


Figure 5 Pole placement according to the characteristic polynomial $\left(s^2+2\zeta_a\omega_a s+\omega_a^2\right)\left(s^2+2\zeta_b\omega_b s+\omega_b^2\right)$, where $\zeta_a=\cos\theta_a$ och $\zeta_b=\cos\theta_b$.

5. Observer

In practice, we cannot measure all the states of the process, but only its output y. Instead we use a model of the process and feed the model with the same input as the real process. The difference between the outputs of the model and real process is used to correct the state of the model so that it converges to the state of the process. Such a device is called an *observer* or a *Kalman filter*. (The Kalman filter is a special type of observer, but here we will not distinguish further between observer and Kalman filter.)

The observer is described by

$$\frac{d\hat{x}}{dt}=A\hat{x}+Bu+K\left(y-C\hat{x}\right) \tag{3}$$

where $\hat{x}$ denotes the estimated states. The column vector K can be chosen such that the states of the observer converge to the states of the process arbitrarily fast, given that the system is observable.

Assignment 5.1 Calculate the rank of the observability matrix for the system:

```
>> Wo = [ ... ]
>> rank(Wo)
```

What is the rank? Is the system observable?

Using the Kalman filter we can establish feedback from the estimated states instead of the real states, see figure 6. The new control law becomes

$$u = -L\hat{x} + l_r r \tag{4}$$

Assignment 5.2 (Preparation) Starting out with (3) and (4), show that the controller based on state feedback from the estimated states can be written in the form

$$\frac{d\hat{x}}{dt} = A_R\hat{x} + B_{R_y}y + B_{R_r}r$$
$$u = C_R\hat{x} + D_{R_y}y + D_{R_r}r$$

What are A_R, B_{R_y}, B_{R_r}, C_R, D_{R_y} and D_{R_r}?

Since the process has four states, the observer will also have four states. We specify the poles of the observer according to the following characteristic polynomial:

$$\left(s^2 + 2\zeta_c\omega_c s + \omega_c^2\right)\left(s^2 + 2\zeta_d\omega_d s + \omega_d^2\right)$$

A suitable choice of poles depend, among other things, on the amount of measurement noise, the size of modeling inaccuracies and whether the initial condition is known. Fast poles mean high amplification of measurement noise, whereas slow poles give slow convergence of the estimate. As starting point, a rule of thumb stating that the observer poles should be 1.5-2 times faster than the state feedback, could be used.

Assignment 5.3 Edit the script `design2.m` and enter the values of ω_a, ζ_a, ω_b and ζ_b from Section 4. Then enter some suitable values for ω_c, ζ_c, ω_d and ζ_d. Calculate the entire controller (state feedback + observer) by executing the script

```
>> design2
```

Then open the Simulink model `model2.mdl` by typing

```
>> model2
```

Simulate the closed loop system and see if it fulfills the reference step response and control signal specifications. Change the design parameters and iterate the procedure until desired behavior is obtained. (If necessary, also change the pole placement for the state feedback.)

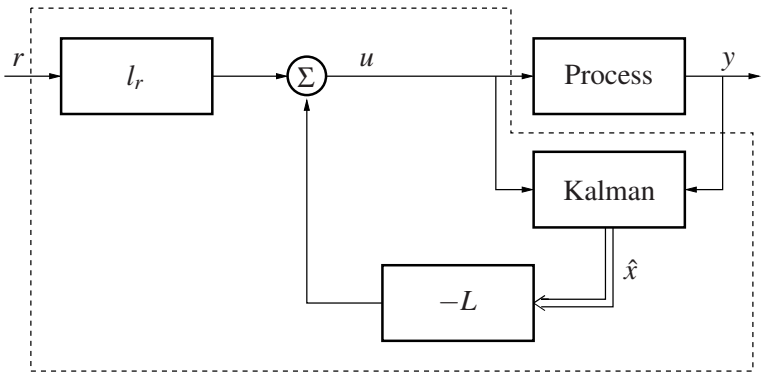


Figure 6 Feedback from estimated states. The controller consists of the blocks within the dashed line.

What is a suitable pole placement for the observer? Why is the load disturbance rejection so poor?

Assignment 5.4 Draw the Bode plot of the controller by typing

```
>> figure()
>> bode(Gr)
>> dcgain(Gr)
```

What gain does the controller have for low frequencies? What does this mean to the controller's ability to suppress constant load disturbances?

Assignment 5.5 Try the controller on the "Real Process" (double click on the process block to toggle between simulated and "real" process). The "real" process has an additional motor time constant, measurement noise, and nonlinear friction added to the dynamics.

How does the "real" step response differ from the simulated one?

6. Integral Action

To eliminate stationary errors due to friction and load disturbances, integral action is introduced in the controller, see figure 6. The integrator is introduced as an extra state x_i according to

$$\begin{aligned} x_i &= \int (r-y)\,dt \\ \dot{x}_i &= r-y = r-Cx \end{aligned} \tag{5}$$

If the extended state vector

$$x_e = \begin{pmatrix} x \\ x_i \end{pmatrix}$$

is introduced, the extended system (i.e. the process and the integrator) can be written

$$\begin{aligned} \dot{x}_e &= \underbrace{\begin{pmatrix} A & 0 \\ -C & 0 \end{pmatrix}}_{A_e} x_e + \underbrace{\begin{pmatrix} B \\ 0 \end{pmatrix}}_{B_e} u + \underbrace{\begin{pmatrix} 0 \\ 1 \end{pmatrix}}_{B_r} r \\ y &= \underbrace{\begin{pmatrix} C & 0 \end{pmatrix}}_{C_e} x_e \end{aligned}$$

If we, for the moment, reassume that the entire state vector is measurable, we can establish feedback from both the states of the process and the integral state according to

$$u = -Lx - l_i x_i + l_r r = -L_e x_e + l_r r$$

where

$$L_e = \begin{pmatrix} L & l_i \end{pmatrix}$$

The closed loop system becomes

$$\begin{aligned} \dot{x}_e &= (A_e - B_e L_e) x_e + (B_e l_r + B_r) r \\ y &= C_e x_e \end{aligned}$$

Because the extended system has five states, the poles of the state feedback are now specified using a fifth order characteristic polynomial:

$$\left(s^2 + 2\zeta_a \omega_a s + \omega_a^2\right)\left(s^2 + 2\zeta_b \omega_b s + \omega_b^2\right)\left(s + \omega_e\right)$$

As before, we cannot measure the states of the process. Consequently, feedback is established from the estimated states and the integrator according to the control law

$$u = -L\hat{x} - l_i x_i + l_r r \tag{6}$$

Assignment 6.1 (Preparation) Starting out with (3), (5) and (6), show that a controller with integral action based on state feedback from estimated states can be written in the form

$$\begin{pmatrix} \frac{d\hat{x}}{dt} \\ \frac{dx_i}{dt} \end{pmatrix} = \underbrace{\begin{pmatrix} * & * \\ * & * \end{pmatrix}}_{A_R} \begin{pmatrix} \hat{x} \\ x_i \end{pmatrix} + \underbrace{\begin{pmatrix} * \\ * \end{pmatrix}}_{B_{Ry}} y + \underbrace{\begin{pmatrix} * \\ * \end{pmatrix}}_{B_{Rr}} r$$

$$u = \underbrace{\begin{pmatrix} * & * \end{pmatrix}}_{C_R} \begin{pmatrix} \hat{x} \\ x_i \end{pmatrix} + \underbrace{\begin{pmatrix} * \end{pmatrix}}_{D_{Ry}} y + \underbrace{\begin{pmatrix} * \end{pmatrix}}_{D_{Rr}} r$$

What are A_R, B_{R_y}, B_{R_r}, C_R, D_{R_y} and D_{R_r}? ◇

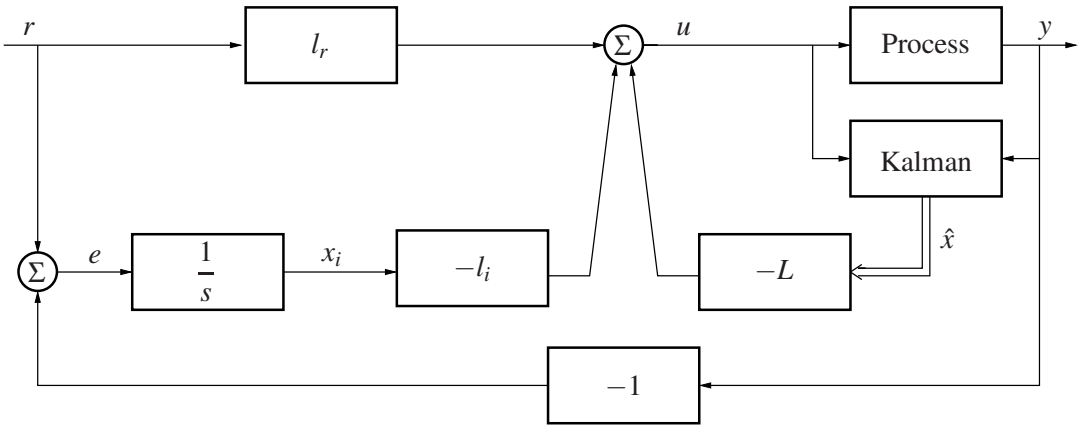


Figure 7 Feedback from estimated states with integral action.

Assignment 6.2 Edit the script `design3.m` and insert your values on ω_a, ζ_a, ω_b, ζ_b, ω_c, ζ_c, ω_d and ζ_d. Also insert suitable values of ω_e. Calculate the entire controller (state feedback + integrator + observer) by executing the script

```
>> design3
```

Open the Simulink model `model3.mdl` by typing

```
>> model3
```

Simulate the closed loop system and see whether it fulfills the specifications. Change the design parameters and iterate the procedure until the specifications are fulfilled. What is a suitable pole placement?

Assignment 6.3 Draw the Bode plot of the controller using

```
>> bode(Gr)
>> grid on
```

How can it seen that the controller has integral action?

Assignment 6.4 Try the controller on the "real" process. How do the results differ from Assignment 5.5.?

When integral action is introduced, the term $l_r r$ is no longer needed in the control law to obtain the correct static gain – this is handled by the integrator. Instead l_r can be chosen to trim the step response at reference value changes. As seen in (6), $l_r \neq 0$ means a direct connection between reference value and control signal. A value $l_r > 0$ can also be used to give the process an extra "push" at a reference value step. (This could be especially useful when controlling the "real" process, which has friction.)

Assignment 6.6 What value does l_r have now? Change the value of `lr` in `design3.m` and conduct new experiments on the "real" process. What is a suitable value for l_r? What happens if l_r is negative?

---

7. Summary

This summary is intended to review relevant questions which you should be able to answer after finishing the experimental part. The lab assistant will go through your summary before you can pass the lab.

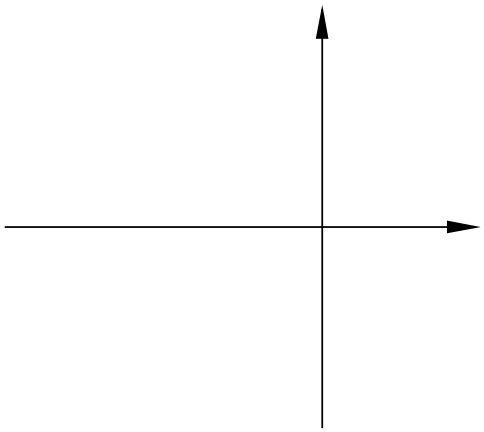
Assignment 8.1 The flexible servo is a strongly resonant process. How can this be seen in its Bode plot and pole-zero diagram, respectively?

Assignment 8.2 How can state feedback be used if all states are not measurable?

Assignment 8.3 When using state feedback from estimated states (Section 5), how many poles does the closed loop system have?

Assignment 8.4 How many states did the controller with integral action (Section 6) contain? Which?

Assignment 8.5 Draw *all* poles of the closed-loop system when using the final controller with integral action (Section 6) in the pole zero plot below:



Assignment 8.6 How can it be seen in the Bode plot of a controller whether it has integral action or not?

A. MATLAB Scripts

define_process.m

```
% Create a linear model of the process

m1 = 2.29; m2 = 2.044;                % masses
d1 = 3.12; d2 = 3.73;                 % damping constants
k = 400;                              % spring constant
km = 2.96;                            % motor constant
ky = 280;                             % measurement constant

A = [0 1 0 0; -k/m1 -d1/m1 k/m1 0; 0 0 0 1; k/m2 0 -k/m2 -d2/m2];
B = [0; km/m1; 0; 0];
C = [0 0 ky 0];
D = 0;

Gp = ss(A,B,C,D);                     % create state space model of the process
```

design1.m — Calculation of controller based on pure state feedback

```
% Design of state feedback

omegaa = ...;                         % speed of one pole pair
zetaa =  ...;                         % damping of one pole pair
omegab = ...;                         % speed of the other pole pair
zetab =  ...;                         % damping of the other pole pair

pc = conv([1 2*omegaa*zetaa omegaa^2],[1 2*omegab*zetab omegab^2]);
poles1 = roots(pc);

L  = place(A,B,poles1);               % compute the state feedback vector L

lr = 1/(C*inv(-A+B*L)*B);             % compute lr such that the static gain
                                      % from r->y becomes 1
```

design2.m — Calculation of controller based on state feedback from observer

```
% Design of state feedback

omegaa = ...;                         % speed of one pole pair
zetaa =  ...;                         % damping of one pole pair
omegab = ...;                         % speed of the other pole pair
zetab =  ...;                         % damping of the other pole pair

pc = conv([1 2*omegaa*zetaa omegaa^2],[1 2*omegab*zetab omegab^2]);
poles1 = roots(pc);

L  = place(A,B,poles1);               % compute the state feedback vector L
lr = 1/(C*inv(-A+B*L)*B);             % compute lr such that the static gain
                                      % from r->y becomes 1
% Design of Observer

omegac = ...;                         % speed of one pole pair
zetac =  ...;                         % damping of one pole pair
omegad = ...;                         % speed of the other pole pair
zetad =  ...;                         % damping of the other pole pair

po = conv([1 2*omegac*zetac omegac^2],[1 2*omegad*zetad omegad^2]);
poles2 = roots(po);
```

```
K = place(A',C',poles2)';              % compute the Kalman gain K

% Computation of controller (observer + state feedback)

AR = A-B*L-K*C;
BRy = K;
BRr = B*lr;
CR = -L;
DRy = 0;
DRr = lr;

Gr = -ss(AR, BRy, CR, DRy);            % transfer function from -y to u
```

design3.m — Calculation of controller based on state feedback from observer with integral action

```
% Design of state feedback with integral action

Ae = [A zeros(4,1); -C 0];             % A-matrix for the extended system
Be = [B; 0];                           % B-matrix for the extended system

omegaa = ...;                          % speed of one pole pair
zetaa =  ...;                          % damping of one pole pair
omegab = ...;                          % speed of the other pole pair
zetab =  ...;                          % damping of the other pole pair
omegae = ...;                          % speed of the fifth pole

pc = conv([1 2*omegaa*zetaa omegaa^2],[1 2*omegab*zetab omegab^2]);
pc = conv(pc, [1 omegae]);
poles1 = roots(pc);

Le  = place(Ae,Be,poles1);             % compute the state feedback vector Le
L = Le(1:4);
li = Le(5);
lr = 0;                                % direct term from reference value

% Design of observer

omegac = ...;                          % speed of one pole pair
zetac =  ...;                          % damping of one pole pair
omegad = ...;                          % speed of the other pole pair
zetad =  ...;                          % damping of the other pole pair

po = conv([1 2*omegac*zetac omegac^2],[1 2*omegad*zetad omegad^2]);
poles2 = roots(po);

K = place(A',C',poles2)';              % compute the Kalman gain K

% Computation of controller (observer + state feedback with integral action)

AR = [A-B*L-K*C -B*li; zeros(1,4) 0];
BRy = [K; -1];
BRr = [B*lr; 1];
CR = [-L -li];
DRy = 0;
DRr = lr;

Gr = -ss(AR,BRy,CR,DRy);               % transfer function from -y to u
```