Automatic Control Laboratory, ETH Zürich
Profs. M. Morari, J. Lygeros

Manual prepared by: P. Brunner, F. Ullmann, S. Richter, C. Fischer
Revision from: February 16, 2013

IfA Fachpraktikum - Experiment 3.7B :

Flexible Shaft B

Comment: This experiment is the continuation of experiment 3.7A and can be done only after that. Here we will design two more feedback controllers, namely

- a loop-shaping controller, and
- an LQR state feedback controller with integral action and observer.

Based on experiment 3.7, the characteristics of so-called flexible structures are introduced. Generally speaking, flexible structures are mechanical systems which can be considered and modeled as many mass points connected by elastic springs, as for instance robots and large satellite dishes. As a simple physical model of such a flexible structure, we are using a shaft consisting of two flywheels connected with a torsional spring. A drive motor accelerates this flexible shaft whereas a load motor is used to simulate a load torque. The goal is to control the speed of the shaft.

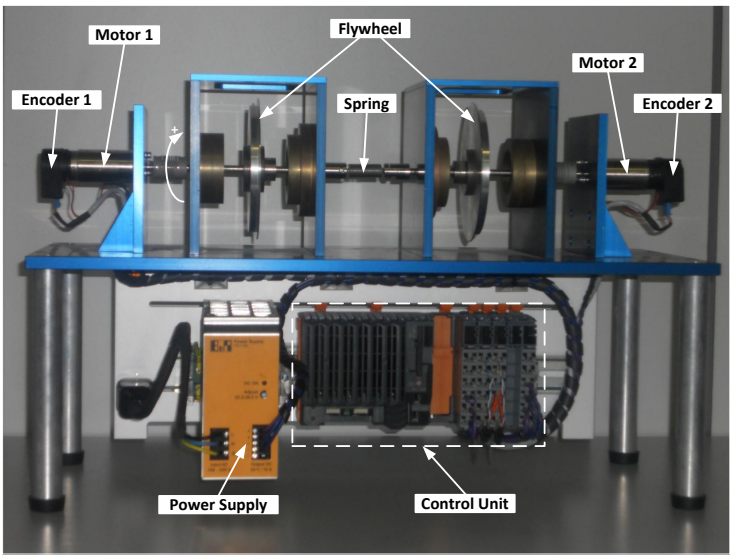

Figure 1: Hardware Setup

Experiment 3.7B is mainly dedicated to the design of feedback controllers. First, a so-called *loop-shaping controller* is developed. Specific tests show that the reference tracking of such controllers is superior to the performance of PI controllers, but the disturbance rejection is of poor quality. Then a *LQR state feedback controller* is designed based on a extended discrete-time model that includes an integral action to reject disturbances. Furthermore, due to the incomplete state measurement, a discrete-time *observer* must be developed. Analysis and experiments show that reference tracking as well as disturbance rejection are satisfactory for step changes.

For the preparation of this experiment at home you need to download the following files from the IfA Fachpraktikum website `http://control.ee.ethz.ch/~ifa-fp/` (provided as a zip-file on the individual experiment's page):

`updateVarParamFlexShaft.m`	MATLAB script for the controller design. Here you have to insert your solutions to all the tasks; in the other files no modifications are required.
`simFlexShaft.mdl`	SIMULINK model of the flexible shaft used to measure step responses.
`continuousTimeSsModelFlexShaft.m`	Continuous-time state space model.
`ConstantParameterFlexibleShaft`	Folder with MATLAB functions containing constant parameters for the SIMULINK model.
`initFcnFlexShaft.m`	MATLAB script to initialize the SIMULINK model.
`stopFcnFlexShaft.m`	MATLAB script to display the data recorded by the SIMULINK model.

During the lab session you will need additional files that are provided locally on your machine.

Remark:

- The tasks in Chapter 2 have to be completed before the lab session.
- The tasks in Chapter 3 are covered during the lab session.

Contents

Chapter 1

Problem Setup and Notation

1.1 Setup

Figure 1 shows the hardware setup for this lab experiment. The flexible shaft consists of a shaft with two flywheels that are connected by a torsional spring. At both ends of the shaft a brushed DC motor is mounted. Motor 1 is used to manipulate the shaft, whereas motor 2 is used to disturb the shaft by simulating a load torque. Both motor angles can be measured independently by an incremental encoder, although in the present experiment only encoder 1 is employed. The angular velocity of motor 1 is measured indirectly by differentiating the corresponding angle.

The flexible shaft is controlled by a real time application running on the control unit. The control unit consists of (from left to right in Figure 1) a CPU module, two counter modules and a motor module, and works with 24V DC voltage provided by the power supply. The motor module generates a PWM signal for each motor; for motor 2 we use the available current controller mode. The real-time application was built from a SIMULINK model. The same model also serves as a interface to interact with the control unit from the lab PC over Ethernet.

1.2 Modeling

In this section, the modeling of the flexible shaft is presented. It is not necessary to understand the modeling in detail, although this enhances the overall comprehension. Figure 1.1 shows the schematic for the flexible shaft including the two brushed DC motors. Table 1.1 introduces the used notation.

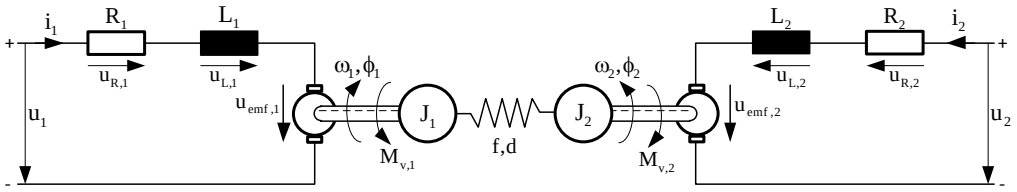

Figure 1.1: Schematic of Flexible Shaft

Notation	Meaning	Unit
u_j	motor input voltage	V
i_j	motor current	A
R_j	motor resistance	Ω
L_j	motor inductance	H = Vs/A
$u_{\text{emf},j}$	back-emf voltage (electro-motive force)	V
$K_{u,j}$	back-emf constant	Vs/rad
$M_{t,j}$	motor torque	Nm
$K_{t,j}$	torque constant	Nm/A
J_j	mass inertia of flywheel and shaft	kgm^2
ω_j	angular velocity	rad/s
$\omega_{m,j}$	measured angular velocity	rad/s
ϕ_j	angle	rad
$\Delta\phi$	angular difference	rad
$M_{v,j}$	viscous friction in bearings	Nm
$K_{v,j}$	coefficient of viscous friction	Nms/rad
f	spring constant	Nm/rad
d	damping coefficient of spring	Nms/rad

Table 1.1: Notation. Note that $j \in \{1, 2\}$.

Mechanical Part

To get the equations of the mechanical part of the flexible shaft, the principle of angular momentum is used. In terms of friction we only consider the viscous friction given as

$$M_{v,1}(t) = K_{v,1} \cdot w_1(t) \ , \quad M_{v,2}(t) = K_{v,2} \cdot w_2(t).$$

This yields the following differential equations for the flywheels and the angles:

$$J_1 \cdot \frac{d}{dt}\omega_1(t) = -f \cdot (\phi_1(t) - \phi_2(t)) - d \cdot (\omega_1(t) - \omega_2(t)) - K_{v,1} \cdot \omega_1(t) + M_{t,1}(t) \tag{1.1a}$$

$$J_2 \cdot \frac{d}{dt}\omega_2(t) = f \cdot (\phi_1(t) - \phi_2(t)) + d \cdot (\omega_1(t) - \omega_2(t)) - K_{v,2} \cdot \omega_2(t) + M_{t,2}(t) \tag{1.1b}$$

$$\frac{d}{dt}\phi_1(t) = \omega_1(t) \tag{1.1c}$$

$$\frac{d}{dt}\phi_2(t) = \omega_2(t) \tag{1.1d}$$

This is obviously a fourth-order system (with states ω_1, ω_2, ϕ_1, ϕ_2). But if we look more closely at the mechanical part of our plant, then we actually see only three energy mass storages present in the system:

1. The right flywheel storing kinetic energy.
2. The torsional spring storing potential energy.
3. The left flywheel storing kinetic energy.

Therefore, there must be an algebraic relation which reduces our system to a third-order system. Examining the differential equations (1.1a) and (1.1b), we see that we can substitute the angular difference

$$\Delta\phi(t) = \phi_1(t) - \phi_2(t), \tag{1.2}$$

which replaces the absolute angles ϕ_1, ϕ_2 as states. Combining (1.1c) and (1.1d) then leads to

$$\frac{d}{dt}\Delta\phi(t) = \frac{d}{dt}(\phi_1(t) - \phi_2(t)) = \omega_1(t) - \omega_2(t). \tag{1.3}$$

From here on we assume that the spring damping is neglectable and that both flywheels (including the shaft they are mounted on) are symmetric, i.e.

$$\begin{aligned} d &= 0 \\ J &= J_1 = J_2 \\ K_v &= K_{v,1} = K_{v,2}. \end{aligned}$$

Finally, we obtain the following differential equations for the mechanical part:

$$J \cdot \frac{d}{dt}\omega_1(t) = -f \cdot \Delta\phi(t) - K_v \cdot \omega_1(t) + M_{t,1}(t) \tag{1.4a}$$

$$J \cdot \frac{d}{dt}\omega_2(t) = f \cdot \Delta\phi(t) - K_v \cdot \omega_2(t) + M_{t,2}(t) \tag{1.4b}$$

$$\frac{d}{dt}\Delta\phi(t) = \omega_1(t) - \omega_2(t) \tag{1.4c}$$

Electrical Part

We assume that the parameters of both motors are equal, i.e.

$$\begin{aligned} R &= R_1 = R_2 \\ L &= L_1 = L_2 \\ K_u &= K_{u,1} = K_{u,2} \\ K_t &= K_{t,1} = K_{t,2}. \end{aligned}$$

Motor 1. As we are using permanent magnet excited brushed DC motors, the following general relations hold:

$$M_{t,1}(t) = K_t \cdot i_1(t) \tag{1.5}$$

$$u_{\text{emf},1}(t) = K_u \cdot \omega_1(t) \tag{1.6}$$

Starting with Kirchhoff's second law we have

$$u_{L,1}(t) = u_1(t) - u_{R,1}(t) - u_{\text{emf},1}(t)$$

and derive the following differential equation for the current:

$$L \cdot \frac{d}{dt} i_1(t) = u_1(t) - R \cdot i_1(t) - K_u \cdot \omega_1(t). \tag{1.7}$$

Figure 1.2 shows the block diagram for motor 1 resulting from (1.5) and (1.7).

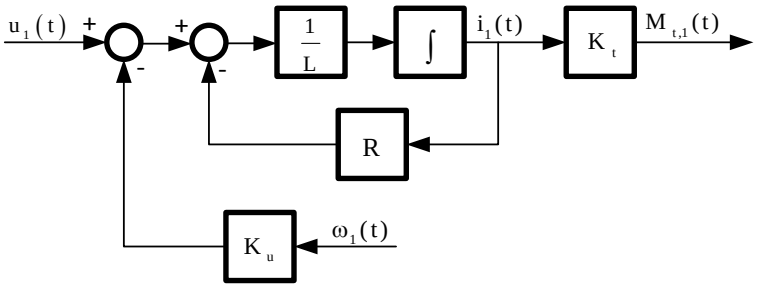

Figure 1.2: Block Diagram of Motor 1

Assuming that the motor current dynamics can be neglected compared to the dynamics of the mechanical system, we set $\frac{d}{dt} i_1(t) = 0$ in (1.7) and obtain for the current

$$i_1(t) = \frac{1}{R} \cdot \left(u_1(t) - K_u \cdot \omega_1(t) \right),$$

which leads by inserting into (1.5) to the algebraic relation

$$M_{t,1}(t) = \frac{K_t}{R} \cdot \left(u_1(t) - K_u \cdot \omega_1(t) \right). \tag{1.8}$$

Motor 2. For motor 2, the same equations as for motor 1 would apply. However, as motor 2 is current-controlled by the motor module of the control unit and we assume that this current controller is ideal, we only use the algebraic relation

$$M_{t,2}(t) = K_t \cdot i_2(t). \tag{1.9}$$

Measurement

The angular velocity $\omega_1(t)$ is measured indirectly by differentiating the angle $\phi_1(t)$. As the angle is measured with an incremental encoder with limited resolution, the velocity measurement carries noise and must therefore be lowpass filtered. We assume that this can be approximated (in the frequency range of interest) by a first-order lowpass filter

$$\frac{d}{dt} \omega_{m,1}(t) = \frac{1}{T_1} \cdot \left(\omega_1(t) - \omega_{m,1}(t) \right), \tag{1.10}$$

where $\omega_{m,1}(t)$ is the measured angular velocity.

Complete Model

With the definitions

$$\begin{aligned} x(t) &\triangleq \begin{bmatrix} \omega_{m,1}(t) & \omega_1(t) & \omega_2(t) & \Delta\phi(t) \end{bmatrix}^T \\ w(t) &\triangleq \begin{bmatrix} u(t) & z(t) \end{bmatrix}^T \triangleq \begin{bmatrix} u_1(t) & i_2(t) \end{bmatrix}^T \\ y(t) &\triangleq \omega_{m,1}(t) \end{aligned}$$

and by using the equations (1.4), (1.8), (1.9) and (1.10), the **state space model** of the flexible shaft becomes:

$$\begin{aligned} \frac{d}{dt} x(t) &= \underbrace{\begin{bmatrix} -\frac{1}{T_1} & \frac{1}{T_1} & 0 & 0 \\ 0 & -\frac{1}{J}\left(K_v + \frac{K_t \cdot K_u}{R}\right) & 0 & -\frac{f}{J} \\ 0 & 0 & -\frac{K_v}{J} & \frac{f}{J} \\ 0 & 1 & -1 & 0 \end{bmatrix}}_{=A_c} \cdot x(t) + \underbrace{\begin{bmatrix} 0 & 0 \\ \frac{K_t}{J \cdot R} & 0 \\ 0 & \frac{K_t}{J} \\ 0 & 0 \end{bmatrix}}_{\substack{=[B_{c,u} \;\; B_{c,z}] \\ =B_c}} \cdot \underbrace{\begin{bmatrix} u(t) \\ z(t) \end{bmatrix}}_{=w(t)} \\ y(t) &= \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}}_{=C_c} \cdot x(t) + \underbrace{\begin{bmatrix} 0 & 0 \end{bmatrix}}_{\substack{=[D_{c,u} \;\; D_{c,z}] \\ =D_c}} \cdot w(t) \; . \end{aligned} \tag{1.11}$$

This model is illustrated in Figure 1.3.

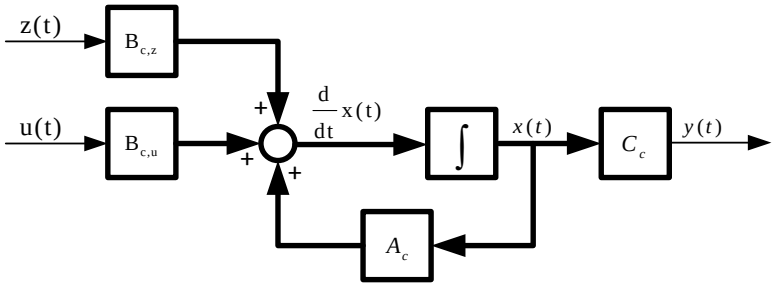

Figure 1.3: Block Diagram of Continuous-Time State Space Model

For a better understanding, the inputs and outputs of the system are discussed again:

- $u(t)$ denotes the manipulating input, i.e. the voltage of motor 1.
- $z(t)$ denotes the disturbance input, i.e. the current of motor 2.
- $y(t)$ denotes the output, i.e. the measured angular velocity of motor 1.

The state space representation (1.11) can be used to derive the transfer function models. The **input-output transfer function** $G(s)$ can be computed as

$$\frac{Y(s)}{U(s)} = G(s) = C_c \cdot (s \cdot I - A_c)^{-1} \cdot B_{c,u} + D_{c,u} \ , \tag{1.12}$$

and the **disturbance transfer function** $G_z(s)$ as

$$\frac{Y(s)}{Z(s)} = G_z(s) = C_c \cdot (s \cdot I - A_c)^{-1} \cdot B_{c,z} + D_{c,z} \ . \tag{1.13}$$

This finally yields the block diagram shown in Figure 1.4.

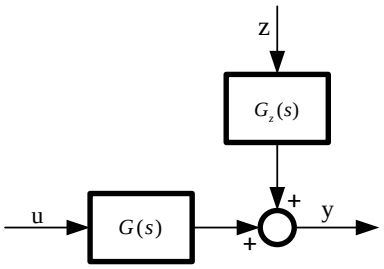

Figure 1.4: Block Diagram of Continuous-Time Transfer Function Model

Chapter 2

Preparation@Home B

In this exercise, we first design a feedback controller using the so called loop-shaping method. Then, we transform the continuous-time state space model into a discrete time model, and based on this model, we finally design a state feedback controller with observer.

2.1 Instructions

1. The exercises of this chapter must be completed before the lab session of Chapter 2.
2. Open MATLAB and go to the folder created in Part A of the home exercise. If you have adapted the script `updateVarParamFlexShaft.m` during the lab session, copy it to the current MATLAB folder.
3. Open the script `updateVarParamFlexShaft.m`. Make sure that the following values are set: `modeSel=0`, `cmd=1`, `J=402e-6` (in `CONFIGURATION`).
4. Now you are ready to solve the tasks given in the next section of this chapter step by step. Write your code directly into `updateVarParamFlexShaft.m` at the marked positions. In this file, the continuous-time state space model is available in the structure `syscr`.
5. In some tasks you should run the simulator to measure some responses. In this case, proceed as follows:
   (a) Open the SIMULINK model `simFlexShaft.mdl`.
   (b) Configure `updateVarParamFlexShaft.m` according to the given task, i.e. set the function selector `fcnSel` and the controller selector `controllerSel` to the appropriate value (in `CONFIGURATION`). Then, save the file.
   (c) Start the simulation in `simFlexShaft.mdl`. This automatically loads the parameters of `updateVarParamFlexShaft.m` into the model and starts with measuring the desired response.
   (d) As soon as the measurements are completed, the simulation stops automatically and the recorded data is displayed. View the plot and save it for your records.
6. After having completed all the tasks: Store the file `updateVarParamFlexShaft.m` safely; you will need it in the lab session to test your controller on the real hardware.

2.2 Loop-Shaping Controller

Before we start with the actual controller design, the key points of the loop-shaping method are highlighted based on the book of Skogestad and Postletwhaite [2, §2.6]. Examine now the feedback structure shown in Figure 2.1, where

- u is the input-output input, hence the input voltage of motor 1,
- z is the disturbance input, hence the current of motor 2,
- y is the output, hence the measured angular velocity of motor 1,
- r is the reference input, hence the set point value of the angular velocity of motor 1,
- n is some additive measurement noise.

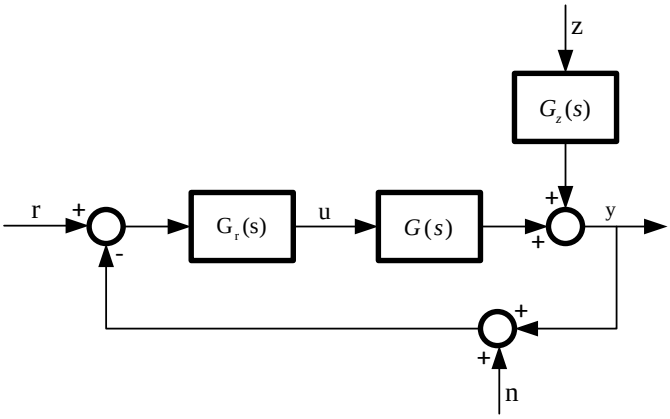

Figure 2.1: Block Diagram of Feedback Structure

In the classical loop-shaping approach to feedback controller design, "loop-shape" refers to the magnitude of the loop transfer function $L(s) = G(s) \cdot G_r(s)$ as a function of frequency, i.e. $|L(j\omega)|$.

First, we have a look at the control error which is defined as

$$e = y - r, \tag{2.1}$$

and yields the closed-loop response

$$e = -(I + L(s))^{-1} \cdot r + (I + L(s))^{-1} \cdot G_z(s) \cdot z - (I + L(s))^{-1} \cdot L(s) \cdot n. \tag{2.2}$$

For "perfect control" we want $e = y - r = 0$ or in other words

$$e \approx 0 \cdot r + 0 \cdot z + 0 \cdot n. \tag{2.3}$$

To achieve approximately perfect control, the following design objectives must be fulfilled:

1. Good disturbance rejection: needs large controller gains, i.e. $|L(j\omega)|$ large.
2. Good command following (reference tracking): $|L(j\omega)|$ large.
3. Mitigation of measurement noise: $|L(j\omega)|$ small.
4. Small magnitude of input signal u: $|G_r(j\omega)|$ and $|L(j\omega)|$ small.

5. Physical system must be strictly proper (relative degree of $|L(j\omega)| > 0$, i.e. more poles than zeros): $|L(j\omega)|$ approaches 0 at high frequencies.

6. Closed-loop stability (stable plant): $|L(j\omega)|$ small.

We see that feedback controller design involves a trade-off between conflicting objectives. Fortunately, they are generally in different frequency ranges, and we can meet most of the objectives by using a large loop gain ($|L(j\omega)| > 1$) at low frequencies below crossover, and a small gain ($|L(j\omega)| < 1$) at high frequencies above crossover.

In conclusion, we specify a reasonable loop transfer function $L(s)$ for our problem as follows:

1. The gain crossover frequency should be smaller than the antiresonance and resonance frequencies.

2. The slope of $|L(j\omega)|$ must be at least -1 at low frequencies, meaning zero steady-state error to step changes in the reference or disturbance input.

3. The phase margin should be larger than $50°$.

4. The controller must be proper (relative degree of $L(s) \geqslant 0$).

5. The bounds $[-18\text{V} + 18\text{V}]$ of the control action u shall be respected.

Task 1: Controller Design

1. As the system is minimum phase, we might want to follow the idea to cancel the poles and zeros of the plant by an inverse-based controller design approach ($G_r(s) = G(s)^{-1}$). Is it reasonable to compensate the poles and zeros originating from the resonance respectively the antiresonance point? In particular, consider the case, where the poles and zeros of the plant are nearer to the imaginary axis of the complex s-plane than assumed, e.g. due to an increased moment of inertia J.

2. Design a loop-shaping controller that meets the specified requirements, and plot the Bode diagram of the resulting loop transfer function $L(s)$. Write your code into `updateVarParamFlexShaft.m` at marking `Task B.1` and run the file.
   Hint: In industrial drive applications, the following approach is widely used:

$$G_r(s) = \underbrace{K_p \frac{s + K_i}{s}}_{\text{PI controller}} \cdot \underbrace{\frac{s^2 + s \cdot 2 \cdot \omega_0 \cdot \frac{d}{c} + \omega_0^2}{s^2 + s \cdot 2 \cdot \omega_0 \cdot \frac{1}{c} + \omega_0^2}}_{\text{notch filter}} .$$

   You can use your PI controller of Part A as a basis and design a notch filter to damp the resonance peak.

3. Use the provided simulator (as described in step 5 of Section 2.1) to plot the reference step response and the closed-loop disturbance step response. Set the corresponding function selector (`4` or `5`) and make sure that the controller selector is set to `loopShaping`.

4. Keep the controller parameters, increase the moment of inertia J to the value $1002 \cdot 10^{-6}\text{kgm}^2$ and plot again the Bode diagram of the loop transfer function $L(s)$. Afterwards, use the simulator to plot the step responses and comment on the results.

Remark: We do the design in continuous time, despite the fact that we are applying digital control. In the script `updateVarParamFlexShaft.m`, the continuous-time transfer function of the controller, $G_r(s)$, is automatically transformed into discrete time using the function `c2d` with Tustin's method. This method essentially substitutes the approximation $s \approx \frac{2}{T} \cdot \frac{z-1}{z+1}$, where T is the sampling interval, into the continuous-time transfer function, [1, §8.3].

2.3 State Feedback Controller

In this section, a LQR state feedback controller will be designed. We know from theory that for state feedback control all states must be available. But in this experiment only one state is measured. Therefore, the remaining states must be estimated by an observer. Furthermore, to reject disturbances and model uncertainties, the control loop should be extended by an integral action.

We will do a direct digital design, i.e. a design that is based on a discrete model of the plant. So first of all it is necessary to convert the continuous-time state space model to a discrete-time model (§2.3.1). Knowing that for LTI systems the separation principle holds, the problem of designing a state feedback controller with integral action and an observer can be split into two separate problems: one of designing the full-state feedback controller with integral action (§2.3.2) and another one of designing the observer (§2.3.3). Finally the resulting combined system is analyzed (§2.3.4).

2.3.1 Discrete Time State Space Model

Sampling the solution of the continuous-time model (1.11) with sampling period T and using a zero-order hold (ZOH) to keep the inputs to the plant $w(kT)$ constant over each sampling period, results in the discrete-time model

$$\begin{aligned} x(k+1) = A_d \cdot x(k) + B_d \cdot w(k) \quad &= A_d \cdot x(k) + \begin{bmatrix} B_{d,u} & B_{d,z} \end{bmatrix} \cdot \begin{bmatrix} u(k) \\ z(k) \end{bmatrix} \\ y(k) = C_d \cdot x(k) + D_d \cdot w(k) \quad &= C_d \cdot x(k) + \begin{bmatrix} D_{d,u} & D_{d,z} \end{bmatrix} \cdot \begin{bmatrix} u(k) \\ z(k) \end{bmatrix}, \end{aligned} \tag{2.4}$$

where

$$\begin{aligned} A_d &= e^{A_c \cdot T} & B_d &= \int_0^T e^{A_c \cdot (T-\tau)} d\tau \cdot B_c \\ C_d &= C_c & D_d &= D_c \ . \end{aligned}$$

Task 2: Discrete-Time State Space Model

Set the moment of inertia J in `updateVarParamFlexShaft.m` back to the value $402 \cdot 10^{-6}\text{kgm}^2$. For each subtask, write your code into `updateVarParamFlexShaft.m` at marking `Task B.2` and run the file.

1. Derive the discrete-time state space model of the flexible shaft by using the MATLAB function `c2d`.
2. Calculate the eigenvalues of the matrix A_d using the MATLAB function `eig` and compare the results with the eigenvalues of the matrix A_c transformed into the z-domain by the relation $z = e^{s \cdot T}$, where T is the sampling interval.
3. Show that the system represented by the discrete-time state space model is controllable. To do this, you might use the MATLAB functions `rank` and `ctrb`.
4. Show that the system is observable. Among other things, you might use the MATLAB function `obsv`.

2.3.2 LQR Full-State Feedback Controller with Integral Action

Let us consider the full-state feedback controller with integral action as depicted in Figure 2.2. For the integral action, we used the forward Euler method to approximate the continuous-time integrator in discrete time, yielding

$$\frac{d}{dt} x_I(t) \approx \frac{x_I(k+1) - x_I(k)}{T} = -\left(r(k) - C_d \cdot x(k)\right).$$

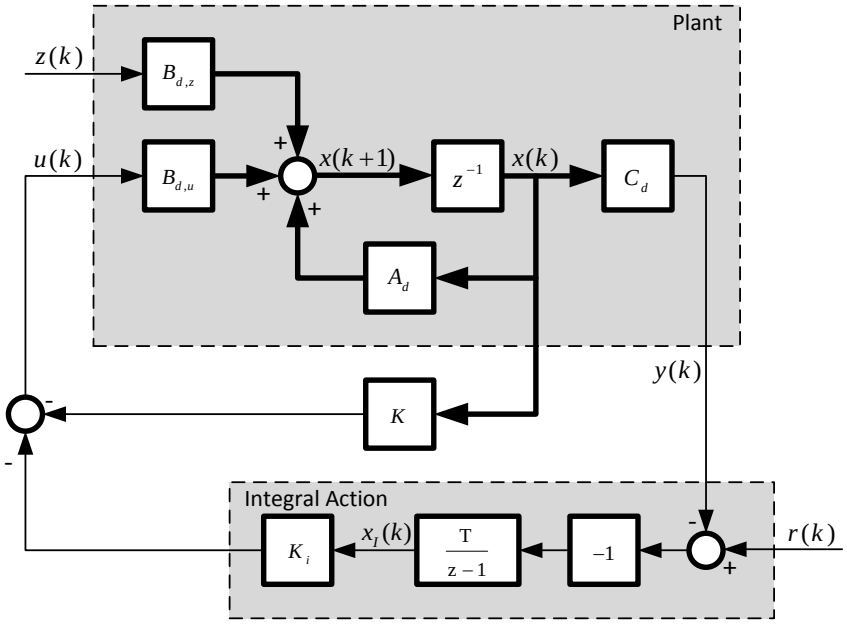

Figure 2.2: Block Diagram of Full-State Feedback Controller

The discrete-time state space model for the plant is according to (2.4). Extending this model with the integral action yields the extended model

$$\underbrace{\begin{bmatrix} x_I(k+1) \\ x(k+1) \end{bmatrix}}_{=\bar{x}(k+1)} = \underbrace{\begin{bmatrix} 1 & T \cdot C_d \\ 0 & A_d \end{bmatrix}}_{=\bar{A}_d} \cdot \underbrace{\begin{bmatrix} x_I(k) \\ x(k) \end{bmatrix}}_{=\bar{x}(k)} + \underbrace{\begin{bmatrix} 0 & -T & 0 \\ B_{d,u} & 0 & B_{d,z} \end{bmatrix}}_{\substack{=[\bar{B}_{d,u} \;\; \bar{B}_{d,r} \;\; \bar{B}_{d,z}] \\ =\bar{B}_d}} \cdot \underbrace{\begin{bmatrix} u(k) \\ r(k) \\ z(k) \end{bmatrix}}_{=\bar{w}(k)}$$
$$y = \underbrace{\begin{bmatrix} 0 & C_d \end{bmatrix}}_{=\bar{C}_d} \cdot \underbrace{\begin{bmatrix} x_I(k) \\ x(k) \end{bmatrix}}_{=\bar{x}(k)}, \tag{2.5}$$

which represents the gray part of Figure 2.2.

We will determine the static feedback gain $[K_i, K]$ optimization-based, namely by an LQR design. For this, consider the discrete-time linear system

$$\bar{x}(k+1) = \bar{A}_d \cdot \bar{x}(k) + \bar{B}_{d,u} \cdot u(k)$$

and the quadratic cost

$$J = \sum_{k=0}^{\infty} \bar{x}(k)^T \cdot Q \cdot \bar{x}\,(k) + u(k)^T \cdot R \cdot u(k),$$

where Q is a symmetric positive semidefinite matrix and R a positive scalar. Then the optimal control law minimizing the quadratic cost is given by

$$u(k) = -\underbrace{\begin{bmatrix} K_i & K \end{bmatrix}}_{=\bar{K}} \cdot \bar{x}(k) \tag{2.6}$$

where

$$\bar{K} = (R + \bar{B}_{d,u}^T \cdot P \cdot \bar{B}_{d,u})^{-1} \cdot \bar{B}_{d,u}^T \cdot P \cdot \bar{A}_d$$

and P is the unique positive definite solution of the discrete algebraic Riccati equation (DARE)

$$P = Q + \bar{A}_d^T \cdot \left(P - P \cdot \bar{B}_{d,u} \cdot \left(R + \bar{B}_{d,u}^T \cdot P \cdot \bar{B}_{d,u} \right)^{-1} \cdot \bar{B}_{d,u}^T \cdot P \right) \cdot \bar{A}_d \ .$$

If the pair $[\bar{A}_d, \bar{B}_{d,u}]$ is stabilizable and the pair $[Q^{\frac{1}{2}}, \bar{A}_d]$ detectable, then the DARE has a unique solution. Furthermore, the closed-loop system is stable, meaning that all eigenvalues of the matrix $\left(\bar{A}_d - \bar{B}_{d,u} \cdot \bar{K}\right)$ lie within the unit circle.

Task 3: State Feedback Gain

For each subtask, write your code into `updateVarParamFlexShaft.m` at marking `Task B.3` and run the file.

1. Define the matrices of the extended system (2.5) in MATLAB.
2. Show that the extended system is still controllable (using MATLAB).
3. Derive an optimal feedback gain $\bar{K}$ using the MATLAB function `dlqr`. For this you have to define the matrix Q (penalty on the states, usually diagonal) and the scalar R (penalty on the input). Note that a large Q results in small states, whereas a large R results in small inputs, and that there is a trade-off between both. Tune Q and R such that
   (i) the closed-loop system is well damped, and
   (ii) the amplitude of the control action u becomes not too high, meaning that closed loop poles lie not too far left in the left half of the complex s-plane.

   Hint: You may use the relation $z = e^{s \cdot T}$ to check the eigenvalues of the matrix $\left(\bar{A}_d - \bar{B}_{d,u} \cdot \bar{K}\right)$.

2.3.3 Observer

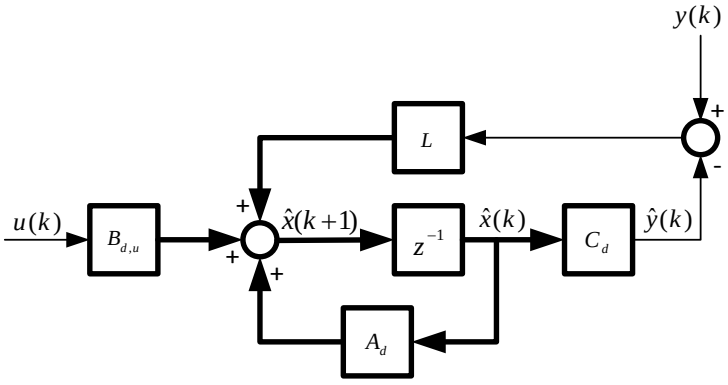

Figure 2.3: Block Diagram of Observer

We use the observer shown in Figure 2.3 to get an estimate $\hat{x}(k)$ of the full state vector $x(k)$. According to the Figure, the dynamics of the observer are given as

$$\hat{x}(k+1) = A_d \cdot \hat{x}(k) + B_{d,u} \cdot u(k) + L \cdot \left(y(k) - C_d \cdot \hat{x}(k) \right),$$

or rewritten as

$$\hat{x}(k+1) = \left(A_d - L \cdot C_d \right) \cdot \hat{x}(k) + B_{d,u} \cdot u(k) + L \cdot y(k). \tag{2.7}$$

We define the estimation error as $\epsilon(k) = \hat{x}(k) - x(k)$, and then use equations (2.7) and (2.4) to derive the error dynamics as

$$\begin{aligned}\epsilon(k+1) = \hat{x}(k+1) - x(k) &= (A_d - L \cdot C_d)\,(\hat{x}(k) - x(k)) - B_{d,z} \cdot z(k) \\ &= (A_d - L \cdot C_d) \cdot \epsilon(k) - B_{d,z} \cdot z(k).\end{aligned}$$

When the pair (A_d, C_d) is observable, then by an appropriate choice of the observer gain L, the eigenvalues of the matrix $(A_d - L \cdot C_d)$ can be placed within the unit circle, meaning that the error dynamics are stable.

Task 4: Observer Gain

Derive the vector L using the MATLAB function `dlqr`. Here, Q corresponds to the covariance of the state (model quality), whereas R corresponds to the covariance of the measurement (noise level). Thus, with a small Q compared to R we trust the model more than the measurement (noisy measurement), whereas with a large Q compared to R we have a good measurement but a poor model. Write your code into `updateVarParamFlexShaft.m` at marking `Task B.4` and run the file.

Hint: With the property $\det(P) = \det\left(P^T\right)$ of determinants we get:

$$\det\left(\lambda \cdot I - (A_d - L \cdot C_d)\right) = \det\left(\lambda \cdot I - \left(A_d^T - C_d^T \cdot L^T\right)\right)$$

2.3.4 State Feedback Controller with Observer

Putting the equations (2.5) and (2.7) together we get the following state space description for the open-loop system with integral action and observer:

$$\begin{aligned}\underbrace{\begin{bmatrix} x_I(k+1) \\ x(k+1) \\ \hat{x}(k+1) \end{bmatrix}}_{=\tilde{x}(k+1)} &= \underbrace{\begin{bmatrix} 1 & T \cdot C_d & 0 \\ 0 & A_d & 0 \\ 0 & L \cdot C_d & A_d - L \cdot C_d \end{bmatrix}}_{=\tilde{A}_d} \cdot \underbrace{\begin{bmatrix} x_I(k) \\ x(k) \\ \hat{x}(k) \end{bmatrix}}_{=\tilde{x}(k)} + \underbrace{\begin{bmatrix} 0 & -T & 0 \\ B_{d,u} & 0 & B_{d,z} \\ B_{d,u} & 0 & 0 \end{bmatrix}}_{=[\tilde{B}_{d,u}\ \ \tilde{B}_{d,r}\ \ \tilde{B}_{d,z}]} \cdot \underbrace{\begin{bmatrix} u(k) \\ r(k) \\ z(k) \end{bmatrix}}_{=\tilde{w}(k)} \\ y(k) &= \underbrace{\begin{bmatrix} 0 & C_d & 0 \end{bmatrix}}_{=\tilde{C}_d} \cdot \underbrace{\begin{bmatrix} x_I(k) \\ x(k) \\ \hat{x}(k) \end{bmatrix}}_{=\tilde{x}(k)}\end{aligned} \tag{2.8}$$

With the control law

$$u(k) = -\underbrace{\begin{bmatrix} K_i & 0 & K \end{bmatrix}}_{=\tilde{K}} \cdot \underbrace{\begin{bmatrix} x_I(k) \\ x(k) \\ \hat{x}(k) \end{bmatrix}}_{=\tilde{x}(k)}, \tag{2.9}$$

where the term $K \cdot x(k)$ of (2.6) was replaced by $K \cdot \hat{x}(k)$, the closed-loop system can finally be written as

$$\begin{aligned}\tilde{x}(k+1) &= \left(\tilde{A}_d - \tilde{B}_{d,u} \cdot \tilde{K}\right) \cdot \tilde{x}(k) + \begin{bmatrix} \tilde{B}_{d,r} & \tilde{B}_{d,z} \end{bmatrix} \cdot \begin{bmatrix} r(k) \\ z(k) \end{bmatrix} \\ y(k) &= \tilde{C}_d \cdot \tilde{x}(k).\end{aligned} \tag{2.10}$$

Task 5: State Feedback Controller with Observer

For each subtask, write your code into `updateVarParamFlexShaft.m` at marking `Task B.5` and run the file.

1. Define the matrices of the closed-loop system with observer using MATLAB.
2. Plot the frequency response of the reference transfer function and the closed-loop disturbance transfer function using the MATLAB function `dbode`.
3. Use the simulator (as described in step 5 of Section 2.1) to plot the reference step response and the closed-loop disturbance step response. Do not forget to set the corresponding function selector (`4` or `5`) and make sure that the controller selector is set to `state`. In the obtained results, check if the bounds $[-18\text{V} + 18\text{V}]$ for the control action u hold. If not then adjust the feedback and observer gain (i.e. change the matrices Q and R of the LQR design).

Chapter 3

Lab Session B

In this lab session, we verify the loop-shaping controller and the state feedback controller designed in Chapter 2.

3.1 Instructions

1. Double-click the icon `ifa_3_7.ps1` on the desktop of the lab computer. This will download all necessary files into `C:\Scratch\Flex_Shaft` and start MATLAB automatically.

2. Copy the MATLAB script `updateVarParamFlexShaft.m` generated during the home exercises to the current MATLAB folder and open it. Set the mode selector to `modeSel=1` and make sure that the inertia is set to `J=402e-6`.

3. Open the SIMULINK model `targetSysFlexShaft.mdl`, which serves as graphical user interface (GUI) to interact with the control unit. Figure 3.1 shows the GUI with some highlighted elements.

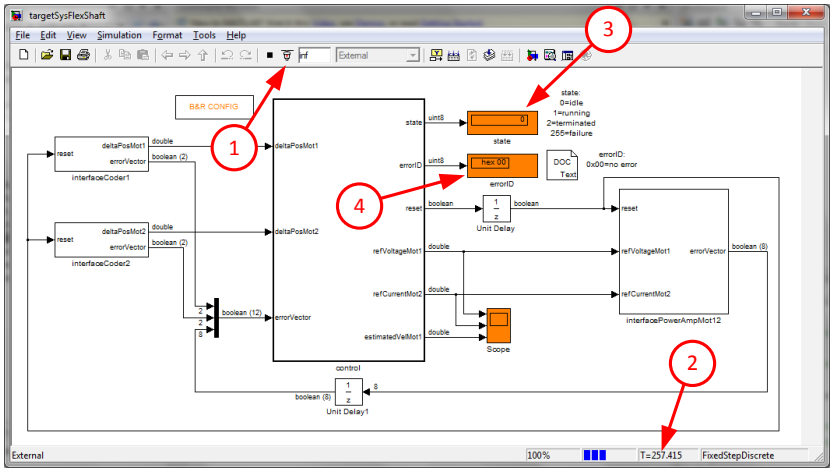

Figure 3.1: GUI Flexible Shaft

4. Power on the control unit of the flexible shaft with the switch on the back side, and wait until booting is completed and the application is running, which is indicated by a green shining `R/E` light (this is the top most) on the front side of the CPU module.

5. Now you are ready to complete the tasks according to the description in the next section of this chapter. For each task proceed as follows:

   (a) Connect (Fig. 3.1(1)) the GUI to the control unit, and wait until the absolute timer (Fig. 3.1(2)) is running.

   (b) Configure `updateVarParamFlexShaft.m` according to the given task, i.e. set the function selector `fcnSel` and the controller selector `controllerSel` to the appropriate value. Set `cmd=1` and run the file. The control unit should go to state *running* (Fig. 3.1(3)). The flexible shaft starts rotating, and the measurements are in progress.

   (c) Wait until the control unit is in state *terminated*, meaning that the measurements are completed. In the meantime you can observe the real system and look at the scope of the GUI. Note:

   - You can stop the flexible shaft at any time by proceeding with step 5d!
   - If the state goes to *failure*, then a hardware error occurred. For more information, check the `errorID` (Fig. 3.1(4)). To reset the error, proceed with step 5d.
   - If SIMULINK shows an error message box, then restart from step 5a.

   (d) Set `cmd=0` in `updateVarParamFlexShaft.m` and run the file. The control unit goes back to state *idle*, and the flexible shaft stops rotating.

   (e) Disconnect (Fig. 3.1(1)) the GUI from the control unit. The recorded data is automatically loaded and displayed. View the plot and save it for your records.

   (f) If the obtained results are not satisfactory enough, adjust the parameters of your controller in `updateVarParamFlexShaft.m` and repeat from step 5a.

6. After having completed all the tasks: Power off the control unit of the flexible shaft.

3.2 Loop-Shaping Controller

Task 6: Verification of Loop-Shaping Controller

1. Measure the reference step response. Do not forget to set the corresponding function selector and make sure that the controller selector is set to `loopShaping`. As usual, the recorded data is plotted into a diagram. Analyze the diagram and summarize the observations you make. You might want to compare the recorded data with the simulated results.
2. Do the same for the closed-loop disturbance step response.

3.3 State Feedback Controller

Task 7: Verification of State Feedback Controller

1. Measure the reference step response. Do not forget to set the corresponding function selector and make sure that the controller selector is set to `state`. Of course the recorded data is plotted into a diagram too. Examine the diagram and based on the observations you make, adjust the feedback and the observer gain respectively. This is usually an iterative process.
2. Do the same for the closed-loop disturbance response.

3.4 Conclusion

Task 8: Conclusion

1. Fill in the online feedback form found in `MyExperiments` on the laboratory courses registration page. Each participant has to give his/her own feedback. Thank you for your inputs! This helps us to improve our experiments.
2. Discuss the experiment with your supervisor to get the confirmation (Testat).

Bibliography

[1] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback control of dynamic systems*. Pearson Prentice Hall, 5th edition, 2006.

[2] Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. John Wiley & Sons, 2nd edition, 2007.