Chapter 3

Models of dynamical systems

Dynamical systems can be modelled from different viewpoints. This chapter summarises the main notions. Each of the succeeding chapters uses one of these models for fault diagnosis and fault-tolerant control.

3.1 Fundamental notions

Fault-tolerant control is based on models. These models have to describe the nominal as well as the faulty system. The following introduces the different models which can be used for fault-tolerant control, starting with the definition of a system as a set of interconnected components, and introducing faults as events which prevent the system components to perform the function they have been designed for.

Dynamical systems. A system is a set of interconnected components. Each of the components has been chosen (or designed) by the system engineer so as to achieve some function of interest. A function describes what the design engineer expects the component to perform, independently of how it is performed. A component performs some function because it has been designed so as to exploit some physical principles, which in general are expressed by some relationships between the time evolution of some system variables. Such relationships are called *constraints*, and the time evolution of a variable is called its *trajectory*.

The components are interconnected by energy or information flows. Energy flows characterise physical systems, which are called "process". Information flows characterise information and control systems.

To illustrate these notions, consider for example a tank. "Storage", which is the *function* classically associated with it, refers to a special operating mode in which the input and the output flows are both equal to zero. In that mode, the mass in the

tank stays constant, which indeed justifies the "storage" denomination. However, many different functions could be assigned to a tank, for example the decoupling (smoothing) of the output flow from some variations of the input flow. This example shows that the notion of function is not univoque, unless the function is understood through the mathematical expression of the constraint that it introduces. In the tank example the tank function would be the "integration" one, since the associated constraint is

$$\frac{dh(t)}{dt} = q_i(t) - q_o(t),$$

where $h(t)$ is the level of the liquid contained in the tank at time t and $q_i(t)$ and $q_o(t)$ are the inflow and outflow at time t.

Controlled systems. Some of the components may have been introduced with the aim of controlling the process, i.e. being able to choose, between all the possible system trajectories, the one which will bring some expected result (cf. Fig. 3.1). Those components which allow to impose the trajectory of a given variable (or to influence the trajectory of a given variable) are called actuators. They establish some constraint between the variables of the process and some control variable, which is called "control signal".

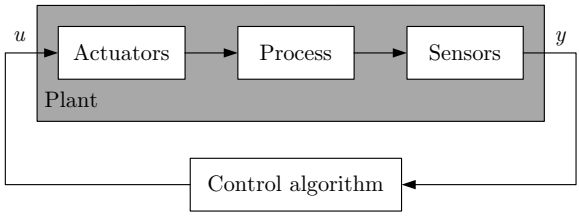


Fig. 3.1. Controlled system

For example, the function of an input valve is to control the input flow in some tank. An analog input valve is associated with the constraint

$$q_i(t) = ku(t)\sqrt{\Delta p}(t),$$

where k is some constant parameter, $u(t)$ is the control signal and $\Delta p(t)$ is the differential pressure on both sides of the valve. Note that, as it is seen from the expression of the constraint, the input valve actually controls the input flow only when the differential pressure $\Delta p(t)$ is controlled (or fixed) by another means. In practice, the signal $u(t)$ controls the ratio $\frac{q_i(t)}{\sqrt{\Delta p(t)}}$.

If instead of a continuous valve an on-off valve is used, a different constraint is associated, namely

$$\begin{aligned} u(t) &= 0 \Longrightarrow q_i(t) = 0 \\ u(t) &= 1 \Longrightarrow q_i(t) = \alpha, \end{aligned}$$

where α is a given constant and $0, 1$ are two logic values which stand for the control signals "closed valve" or "open valve", respectively.

Actuators may be driven (i.e. control signals may be generated) by human operators or by control algorithms. In both cases, closed-loop control demands some information about the actual values of some system variables to be known. Sensors are components which are designed so as to provide this information. An example of such a component is a level sensor, whose function is to provide an image of the actual level in the tank. An analog sensor is associated with the constraint

$$y(t) = h(t) + \varepsilon(t),$$

where $y(t)$ is the signal provided by the sensor at time t and $\varepsilon(t)$ is some stochastic process which models the measurement noise, e.g. with normal distribution $N(0, \sigma)$. A discrete sensor is associated with a constraint expressed by a set of rules, an example of which is given in the following table

$$\begin{array}{lcll} h(t) & \in & [0, \alpha[& \Rightarrow \quad y(t) = 0 \\ h(t) & \in & [\alpha, \beta[& \Rightarrow \quad y(t) = a \\ h(t) & \in & [\beta, \gamma[& \Rightarrow \quad y(t) = b \\ h(t) & \in & \geq \gamma & \Rightarrow \quad y(t) = c, \end{array}$$

where a, b, c and α, β, γ are given constants.

Thus a controlled system is a quadruple

$$<\text{process, actuators, sensors, control devices and algorithms}>.$$

Example 3.1 *Single-tank system*

Consider the following controlled system
$<$(tank, output pipe), input valve, level sensor, level controller$>$.

- Component 1: Tank
  Function: integration
  Constraint: $\frac{dh(t)}{dt} = q_i(t) - q_o(t)$

- Component 2: On-off input valve
  Function: control the input flow
  Constraint: $q_i(t) = \alpha$ if $u(t) = 1$
  $q_i(t) = 0$ if $u(t) = 0$

- Component 3: Output pipe
  Function: deliver the output flow
  Constraint: $q_0(t) = k\sqrt{h(t)}$ where k is some parameter (the output pressure is supposed to be known).

- Component 4: Analog level sensor
  Function: provide an image of the actual level in the tank
  Constraint: $y(t) = h(t) + \varepsilon(t), \varepsilon \sim N(0, \sigma)$

- Component 5: On-off control algorithm
  Function: regulate the level in the tank
  Constraint: if $y(t) \leq h_0 - r$ then $u(t) = 1$,
  if $y(t) \geq h_0 + r$ then $u(t) = 0$, where h_0 and r are given constants. □

Faults. Systems are designed in order to achieve some objectives. Normal operation is an operating mode in which the system's objectives are achieved. Normal operation is defined as the simultaneous occurrence of two situations:

1. The components perform properly the functions they have been assigned. This means that they really behave as the designer expected when he designed them, i.e. the constraints they apply to the system variables are the nominal ones.
2. The variables occuring in the component constraints have values in some domain that are compatible with the system's objectives.

From this, it follows that two kinds of faults can be distinguished. *Internal faults* change the constraints describing the components. *External faults* are associated with variables whose value does not allow to achieve the system's objectives. It can be noticed that internal faults refer to the system's state while external faults refers to the system objectives. Indeed, healthy systems might be unable to achieve the objectives they have been assigned, as the result of inadequate input signals or strong disturbances. On the contrary, faulty systems might still be able to achieve their objective through fault accommodation procedures.

Example 3.2 *Internal faults of the tank*

Consider the single-tank system. Examples of internal faults are the following:

- Process fault: The tank is leaking.
  Then the description of Component 1 introduced in Example 3.1 is replaced by:
  Component 1: Leaking tank
  Constraint: $\frac{dh(t)}{dt} = q_i(t) - q_o(t) - q_l(t)$ where $q_l(t)$ is some (unknown) leakage flow.

- Actuator fault: The input valve is blocked open.
  Then Component 2 is described by:
  Component 2: Blocked-open input valve
  Constraint: $q_i(t) = \alpha$ whatever the value of $u(t)$.

- Sensor fault: The measurement noise has improper statistical characteristics.
  Then Component 3 is described by:
  Component 3: Level sensor
  Constraint: $y(t) = h(t) + \varepsilon(t)$ with $\varepsilon \sim N(0, \Sigma)$ (instead of $N(0, \sigma)$). □

Example 3.3 *External fault of the tank*

Component 4 defined in Example 3.1 is a control algorithm whose function is to regulate the level in the tank, the objective being to keep $h(t)$ within the interval $(h_0 - r, h_0 + r)$ for any

initial value of the level which belongs to this interval. Note that this objective is expressed in terms of two inequality constraints:

$$\begin{aligned} h(t) &\leq h_0 + r \\ h(t) &\geq h_0 - r. \end{aligned}$$

The control signal generated by this algorithm is the input of Component 2 (the actuator) which delivers an input flow α if $y(t) \leq h_0 - r$ holds. It is easily seen that even in the absence of any internal fault, the system objective cannot be achieved for output flows $q_0(t)$ which satisfy, in some time interval (t_1, t_2), the relation

$$\int_{t_1}^{t_2} q_0(t)dt > h(t_1) + \alpha(t_2 - t_1) - h_0 + r.$$

One can also notice that in the presence of a leakage in the tank (internal fault), the system objective may still be achieved provided that the above inequality does not hold when $q_0(t)$ is replaced by $q_0(t) + q_l(t)$. □

3.2 Modelling the system architecture

Generic component models describe the system architecture, by describing the system components and their interconnection. For example, sensors, actuators, unitary process devices are elementary components, but higher level ones can be built from their interconnection. A set of interconnected components can be seen, at a higher hierarchical level, as one single complex aggregated component. For example, the aggregation of a tank, an input valve, an output pipe, a level sensor and a regulator (with consistent connection between them) is a high-level component, namely the single-tank system. Thus, components can be considered at any level in the system hierarchical decomposition, and any subsystem (including the whole system itself) can be considered as a component.

Therefore, the system architecture can be described by instantiating a generic component model, at any level of the system hierarchical description. The aim of the generic component model is to provide a common formal modelling frame for every system component, so as to perform systematic manipulations for the purpose of fault diagnosis and fault-tolerant control design. It is not intended to describe the behaviour of the variables which are associated with the component (this is the aim of the behaviour model), but the services that the component provides, seen from the user viewpoint. In that context, the user is either another component or the human operator.

Services. A component S is first described by the list of the services that it provides to its users, $S = \{s_i, i \in I_s\}$. A service s_i is a transformation of some *consumed variables* $(cons_i)$ into some *produced variables* $(prod_i)$, that is performed by the

component according to a given procedure $(proc_i)$, either in a systematic way, or only upon some specific *request* $(rqst_i)$.

For example, a tank consumes input and output mass flows, and produces a stored mass, using an integration procedure (note that the output flow is indeed an input variable for the integration procedure), thus providing an *integration* service (whose behaviour model is $\dot{h}(t) = q_i(t) - q_o(t)$). This transformation does obviously not need to be requested. On the contrary, the *measurement* service of a sensor consumes (an often neglected amount of) energy from the outside world and produces a signal which is the image of the measured variable, by means of the transducer, at each system clock pulse which requests the sensor analog to digital converter operation.

In general, the transformation procedure needs some resources (res_i) to be available, and it may be enabled or disabled at different times $(enable_i)$. Therefore, the description of a service is a 6-tuple

$$s_i = \{cons_i,\ prod_i,\ proc_i,\ rqst_i,\ enable_i,\ res_i\} \tag{3.1}$$

For example, the integration service of the tank is defined by the 6-tuple

$cons = \{q_i(t), q_o(t)\}$
$prod = \{h(t)\}$
$proc : \dot{h}(t) = q_i(t) - q_o(t)$
$rqst = 1$ (which means that it is always true)
$enable = 1$
$res = \{tank,\ input\ pipe,\ output\ pipe\}$

Versions. Some components exhibit built-in fault tolerance possibilities, which means that they are still able to provide some services even if the associated resources are faulty and no longer available. This is only possible if there are different means to perform the same transformation, among which at least one does not use the faulty resources. In that case, the service is said to exist under several *versions*, where each version is a 6-tuple like (3.1), which can be used indifferently for the same purpose. It is worth noting that all the versions of the same service share the same request, and produce the same output value, but they cannot be simultaneously enabled, and at least one among the input signals, procedures and hardware resources is different from one version to another one. Moreover, since several versions might be able to provide the same result at a given time, there is the need for a mechanism which enables only one of them when the request for the service is issued.

For example, consider a sensor which includes two redundant transducers to measure the same variable. Let

$$\begin{aligned} y_1(t) &= x(t) + \varepsilon_1(t), \quad \varepsilon_1(t) \sim N(0, \sigma_1) \\ y_2(t) &= x(t) + \varepsilon_2(t), \quad \varepsilon_2(t) \sim N(0, \sigma_2) \end{aligned}$$

be the two measurement equations, where $x(t)$ is the unknown variable to be measured, $y_i(t), i = 1, 2$ are respectively the two transducers output, and $\varepsilon_i, i = 1, 2$

are the measurement noises, with the two Gaussian distributions $N(0, \sigma_i), i = 1, 2$. The *measurement* service of this sensor could obviously be provided under different versions, namely

Version	Procedure
1	$y(t) = \frac{\sigma_2}{\sigma_1+\sigma_2}\, y_1(t) + \frac{\sigma_1}{\sigma_1+\sigma_2}\, y_2(t)$
2	$y(t) = y_1(t)$
3	$y(t) = y_2(t)$

where version 1 could be the nominal one, version 2 could be used when transducer 2 is faulty, and version 3 would be used when transducer 1 is faulty.

Use-modes. Not all the services provided by a component are enabled at any time. For that reason, subsets of services are gathered into *use-modes*, whose evolution is described by an automaton, which shows the possible transitions from one use-mode to another one, and the conditions under which these transitions are fired.

For example, a typical controller could be described by three use-modes, namely *Off, Initialise, On*, whose content (services) and evolution (automaton) are given in the following table:

Mode	Possible transitions	Enabled services
Off	*To_Initialise, To_On*	
On	*To_Off, To_Initialise*	*Compute_control, Display_set_point*
Initialise	*To_Off, To_On*	*Enter_set_point, Display_set_point*

Building systems from components. As already mentioned, systems (or subsytems) are high-level components, which are built by the aggregation of lower level ones, following a bottom-up approach. Whatever the component level, its generic model includes its use-mode automaton, and the services which are available in each use-mode. Systematic aggregation procedures are defined in order to compose the generic models of low-level components and obtain the generic models of high-level ones.

Fault tolerance analysis. A use-mode is associated with one or several objectives that the component or system must achieve. At any time, the current use-mode defines the current objective, and the enabled services (requests for other services are rejected). The system fault tolerance results from the fact that, in spite of the failure of some resources, the services which are necessary to achieve the objectives of the current use-mode still exist (under at least one version).

3.3 System behaviour – basic modelling features

Variables. A first question which arises is to select those variables which are of interest to describe the system behaviour. Process components generally introduce power and energy variables, while control systems introduce control and information signals. Therefore, the system variables to be considered are all quantities which are constrained by system components (process, actuators, sensors, control and estimation algorithms). Note that for systems that obey the Markov property, there is a minimal set of variables which summarise the whole past history of the system untill time t (the state variables). The evolution of the state at time t only depends on its value at time t and on the values of the input at time t.

Once the system variables are defined, a second question is to decide about the set of values they can be assigned. Quantitative variables take their values in a subset of the real numbers (which is totally ordered, and provided with the four classical operations), while qualitative variables take their values in a given finite set of symbols, which may be ordered or not. It can be useful to define variables whose values are the segments of some partition of the real line. The coarser the partition, the coarser the granularity of the variable. A symbol is often associated with each segment of the partition, e.g. small, medium, large. Abrupt transitions from one value to another one can be avoided using fuzzy segments instead of crisp ones.

Time. The most classical time variable takes its values in the set of positive real numbers (continuous time). In discrete time systems, the set of positive integers (or any set isomorphic to that one) is used when sampled data systems are considered. This time representation is called synchronous since practical sampling systems are driven by a clock. On the contrary, in event driven systems, time is considered only at each event occurence.

Constraints. The evolution of the system is described by a set of constraints which apply to the system variables. The constraints can be classified according to what they represent and to the form they take.

What constraints represent. In the basic modelling step, each system component is described by its own (local) constraints, and the overall system formed by the interconnection of the components is described by the concatenation of all constraints. In further steps, it may be interesting to solve some constraints and to summarise them within a more compact model.

Example 3.4 *Single-tank system*

For example, the tank associated with an input pump and an output pipe is a three component system described by the three local constraints

$$M_0 : \begin{cases} \text{Tank:} & \dot{h}(t) = q_i(t) - q_0(t) \\ \text{Pump:} & q_i(t) = \alpha.u(t) \\ \text{Pipe:} & q_0(t) = k\sqrt{h(t)} \end{cases}$$

where $u(t)$ is the control signal, and α and k are two parameters. More condensed models may be created as follows

$$M_1 : \begin{cases} \text{Tank + pump:} & \dot{h}(t) = \alpha.u(t) - q_0(t) \\ \text{Pipe:} & q_0(t) = k\sqrt{h(t)} \end{cases}$$

$$M_2 : \begin{cases} \text{Tank + pipe:} & \dot{h}(t) = q_i(t) - k\sqrt{h(t)} \\ \text{Pump:} & q_i(t) = \alpha.u(t) \end{cases}$$

$$M_3 : \left\{ \text{Tank + pump + pipe:} \quad \dot{h}(t) = \alpha.u(t) - k\sqrt{h(t)} \right.$$

Note that the last model uses the minimal number of variables (but it condenses the three components into one single constraint). In fact, $h(t)$ is the system state: the knowledge of $h(t_0)$ and of the input $u(\tau)$, $\tau \in [t_0, t]$ is the only knowledge that is necessary to produce $h(t)$ for any time t. □

The form constraints take. According to the different descriptions of the variables and of time, the constraints have different forms:

- The evolution of continuous variables (whose values are in the set of real numbers) can be described in continuous or in discrete time. Continuous time descriptions basically use algebraic and differential equations and transfer functions (Laplace transform). Discrete time descriptions are useful when computer controlled systems are considered, since the data are sampled at a constant rate by the system clock. They basically use algebraic and difference equations, and transfer functions (based on z-transform). Continuous-variable models will be described in Section 3.4.
- The evolution of qualitative (or symbolic) variables is best described using discrete-event models such as automata, Petri nets, sets of rules. Such models will be described in Section 3.6. Fuzzy variables (and models) can be used when it is wished to avoid abrupt transitions from one qualitative value to another one.
- In many real life systems, continuous variables and qualitative variables co-exist.

Example 3.5 *On/off-temperature control system*

For example, an on/off temperature control system would be described by the continuous model

$$\frac{d\theta}{dt} = -a\theta + b$$

when the heater is on, and by the model

$$\frac{d\theta}{dt} = -a\theta$$

when the heater is off (θ is the temperature to be controlled, and a, b are system parameters). The time evolution of such a system is described not only by the temperature (which is a

continuous variable) but also by the heating mode (on/off) which is a qualitative one. Such systems are described by so-called "hybrid models", which will be developed in Section 3.7. □

3.4 Continuous-variable systems

In continuous-variable systems, the input, state and output variables are defined for a continuum of values in $\mathbb{R}$. Two types of signals can enter such systems: continuous-time or discrete-time signals. For the first, the independent variable t is continuous, and thus such signals are defined over a continuum of time values. Discrete-time signals, on the other hand, are only defined over a time variable k, which belongs to a discrete set. A continuous-time (discrete-time) system processes continuous-time (discrete-time) input signals and generates a continuous-time (discrete-time) output. Models for these two classes of systems are presented next.

Continuous-time model. A quite general state-space model for a continuous-time continuous-variable nonlinear system can be written as

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \tag{3.2}$$

$$\boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)), \tag{3.3}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^m$, $\boldsymbol{y} \in \mathbb{R}^p$ denote the state vector, the vector of known input signals, and the vector of measured output values. $\boldsymbol{d} \in \mathbb{R}^{n_d}$ stands for the vector of unknown input signals or disturbances acting on the process. The functions $\boldsymbol{g}$ and $\boldsymbol{h}$ are respectively $\mathbb{R}^n$-valued and $\mathbb{R}^p$ -valued and they are assumed to be smooth. A model of the form (3.2), (3.3) can be obtained by using physical laws to describe the considered process.

Example 3.6 *Single-tank model*

A continuous-valued signal $u(t)$ is considered instead of a binary one like in Example 3.4. Introducing the pipe constraint into the tank model and assuming a noise free measurement of the level yields

$$\frac{dh(t)}{dt} = -k\sqrt{h(t)} + \alpha u(t) \tag{3.4}$$

$$y(t) = h(t), \tag{3.5}$$

which has the form indicated above with $\boldsymbol{d}(t) = 0$. □

Linear time-invariant systems can be used to describe the behaviour of a nonlinear system of the form (3.2), (3.3) around a specific set-point. Linearisation around a point of operation $\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}}, \bar{\boldsymbol{d}}$ is obtained by introducing $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \bar{\boldsymbol{x}}$, $\tilde{\boldsymbol{u}} = \boldsymbol{u} - \bar{\boldsymbol{u}}$, $\tilde{\boldsymbol{d}} = \boldsymbol{d} - \bar{\boldsymbol{d}}$ and $\tilde{\boldsymbol{y}} = \boldsymbol{y} - \bar{\boldsymbol{y}}$ by performing the Taylor expansion

$$\begin{aligned}\frac{d\tilde{\boldsymbol{x}}}{dt} &= \frac{\partial \boldsymbol{g}(\bar{\boldsymbol{x}},\bar{\boldsymbol{u}},\bar{\boldsymbol{d}})}{\partial \boldsymbol{x}}\tilde{\boldsymbol{x}} + \frac{\partial \boldsymbol{g}(\bar{\boldsymbol{x}},\bar{\boldsymbol{u}},\bar{\boldsymbol{d}})}{\partial \boldsymbol{u}}\tilde{\boldsymbol{u}} + \frac{\partial \boldsymbol{g}(\bar{\boldsymbol{x}},\bar{\boldsymbol{u}},\bar{\boldsymbol{d}})}{\partial \boldsymbol{d}}\tilde{\boldsymbol{d}} \\ \tilde{\boldsymbol{y}} &= \frac{\partial \boldsymbol{h}(\bar{\boldsymbol{x}},\bar{\boldsymbol{u}},\bar{\boldsymbol{d}})}{\partial \boldsymbol{x}}\tilde{\boldsymbol{x}} + \frac{\partial \boldsymbol{h}(\bar{\boldsymbol{x}},\bar{\boldsymbol{u}},\bar{\boldsymbol{d}})}{\partial \boldsymbol{u}}\tilde{\boldsymbol{u}} + \frac{\partial \boldsymbol{h}(\bar{\boldsymbol{x}},\bar{\boldsymbol{u}},\bar{\boldsymbol{d}})}{\partial \boldsymbol{d}}\tilde{\boldsymbol{d}}\end{aligned}$$

to obtain a set of linear equations (see Appendix 1)

$$\frac{d\tilde{\boldsymbol{x}}(t)}{dt} = \boldsymbol{A}_{ct}\,\tilde{\boldsymbol{x}}(t) + \boldsymbol{B}_{ct}\,\tilde{\boldsymbol{u}}(t) + \boldsymbol{E}_{x,ct}\,\tilde{\boldsymbol{d}}(t), \quad \tilde{\boldsymbol{x}}(0) = \boldsymbol{x}_0 \tag{3.6}$$

$$\tilde{\boldsymbol{y}}(t) = \boldsymbol{C}_{ct}\,\tilde{\boldsymbol{x}}(t) + \boldsymbol{D}_{ct}\,\tilde{\boldsymbol{u}}(t) + \boldsymbol{E}_{y,ct}\,\tilde{\boldsymbol{d}}(t), \tag{3.7}$$

where the variables $\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{y}(t)$ and $\boldsymbol{d}(t)$, are defined as above, and $\boldsymbol{A}_{ct}$, $\boldsymbol{B}_{ct}, \ldots$ are matrices of appropriate dimensions with constant entries in $\mathbb{R}$. In the sequel, $\boldsymbol{x}(t)$ is used instead of $\tilde{\boldsymbol{x}}(t)$ following the tradition of linear systems theory.

Example 3.6 (cont.) *Single-tank system*

Let h_0 denote the nominal level around which the tank is normally operated. The nominal control signal u_0 is obtained by looking for the steady state solution of (3.4) at $h = h_0$, which yields $u_0 = k\sqrt{h_0}/\alpha$. Define $\tilde{h}(t) = h(t) - h_0$ and $\tilde{u}(t) = u(t) - u_0$. A straigthforward computation yields

$$\frac{d\tilde{h}(t)}{dt} = -\beta\tilde{h}(t) + \alpha\tilde{u}(t) \tag{3.8}$$

$$\tilde{y}(t) = \tilde{h}(t), \tag{3.9}$$

where $\beta = k/(2\sqrt{h_0})$. In the sequel, $\tilde{h}(t)$, $\tilde{u}(t)$ and $\tilde{y}(t)$ are replaced by $h(t)$, $u(t)$ and $y(t)$ respectively, keeping in mind that the latter represent discrepancies with respect to their nominal value. □

Discrete-time model. Discrete-time models can be used to model sampled-data systems. All signals are assumed to be sampled synchronously at a fixed sampling period T_s. The traditional theory of sampled-data systems relies on the assumption that the input signals are constant over T_s. This holds true for the control variables, as the output of digital to analog converters has this property. It is, however, an approximation for disturbances and faults.

By integrating the state Eq. (3.6) over one sampling period, the following discrete-time model can be deduced from (3.6), (3.7)

$$\begin{aligned}\boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{E}_x\,\boldsymbol{d}(k), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{u}(k) + \boldsymbol{E}_y\,\boldsymbol{d}(k),\end{aligned}$$

where k (actually standing for kT_s) denotes the discrete-time instants,

$$\begin{aligned}\boldsymbol{A} &= \exp\left(\boldsymbol{A}_{ct}T_s\right) \\ \boldsymbol{B} &= \int_0^{T_s} \exp\left(\boldsymbol{A}_{ct}t\right)\boldsymbol{B}_{ct}dt.\end{aligned}$$

$\boldsymbol{E}_x$ is defined in a similar way as $\boldsymbol{B}$. Furthermore, the relations $\boldsymbol{C}{=}\boldsymbol{C}_{ct}$, $\boldsymbol{D}{=}\boldsymbol{D}_{ct}$, $\boldsymbol{E}_y = \boldsymbol{E}_{y,ct}$ hold.

Example 3.6 (cont.) *Discrete-time model of the single-tank system*

Letting $\phi = \exp(-\beta T_s)$ and

$$\gamma = \int_0^{T_s} \alpha \ \exp(-\beta t) dt = \frac{\alpha}{\beta}(1 - \exp(-\beta T_s)),$$

the discrete-time model deduced from (3.8), (3.9) is written as

$$\begin{aligned} h(k+1) &= \phi h(k) + \gamma u(k) \\ y(k) &= h(k). \ \square \end{aligned}$$

Stochastic disturbances and measurement noise. For discrete-time stochastic models, measurement noise and stochastic disturbances possibly acting on the state variables are described by stochastic sequences (cf. Appendix 2). A discrete-time state-space model for the system then takes the form

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{E}_x \boldsymbol{d}(k) + \boldsymbol{w}(k), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{u}(k) + \boldsymbol{E}_y \boldsymbol{d}(k) + \boldsymbol{v}(k). \end{aligned}$$

The only new notations are $\boldsymbol{v}(k)$ and $\boldsymbol{w}(k)$. They are samples of white noise sequences with zero mean and covariance matrix

$$E\left[\begin{pmatrix} \boldsymbol{w}(k) \\ \boldsymbol{v}(k) \end{pmatrix} (\boldsymbol{w}(\ell)' \ \ \boldsymbol{v}(\ell)')\right] = \begin{pmatrix} \boldsymbol{Q}_w & \boldsymbol{Q}_{wv} \\ \boldsymbol{Q}'_{wv} & \boldsymbol{Q}_{\boldsymbol{v}} \end{pmatrix} \delta_{k\ell}.$$

Fault model. Fault signals are usually separated into two classes: additive and non-additive (or multiplicative) faults. *Additive faults* appear as additional terms in the state equations of a linear time-invariant system. For stochastic models, they result in changes of the mean value of the measured signals only. *Multiplicative faults* correspond to changes in the parameters of the state equations, namely changes in the entries of the matrices $\boldsymbol{A}_{ct}$, $\boldsymbol{B}_{ct}$, $\boldsymbol{C}_{ct}$, $\boldsymbol{D}_{ct}$ for a continuous-time model (or $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ for a discrete-time model) or changes in the variance of the stochastic disturbance and noise.

A continuous-time linear system subject to additive faults can thus be modeled by

$$\begin{aligned} \frac{d\boldsymbol{x}(t)}{dt} &= \boldsymbol{A}_{ct}\, \boldsymbol{x}(t) + \boldsymbol{B}_{ct}\, \boldsymbol{u}(t) + \boldsymbol{E}_{x,ct}\, \boldsymbol{d}(t) + \boldsymbol{F}_{x,ct}\, \boldsymbol{f}(t), \ \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \boldsymbol{y}(t) &= \boldsymbol{C}_{ct}\, \boldsymbol{x}(t) + \boldsymbol{D}_{ct}\, \boldsymbol{u}(t) + \boldsymbol{E}_{y,ct}\, \boldsymbol{d}(t) + \boldsymbol{F}_{y,ct}\, \boldsymbol{f}(t). \end{aligned}$$

The type of faults that are accounted for in the above model include sensor faults, actuator faults, and some component faults.

A continuous-time model subject to non-additive faults can be written as

$$\begin{aligned} \frac{d\boldsymbol{x}(t)}{dt} &= \boldsymbol{A}_{ct}(\boldsymbol{\theta})\boldsymbol{x}(t) + \boldsymbol{B}_{ct}(\boldsymbol{\theta})\boldsymbol{u}(t), \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \boldsymbol{y}(t) &= \boldsymbol{C}_{ct}(\boldsymbol{\theta})\boldsymbol{x}(t) + \boldsymbol{D}_{ct}(\boldsymbol{\theta})\boldsymbol{u}(t), \end{aligned}$$

where the entries in the different matrices are smooth functions of the parameter vector $\boldsymbol{\theta}$. Under healthy working conditions, the relation $\boldsymbol{\theta} = \boldsymbol{\theta}_0$ and in faulty conditions the relation $\boldsymbol{\theta} \neq \boldsymbol{\theta}_0$ hold. An example of multiplicative fault is an abnormal change in the armature resistance of a DC motor.

Example 3.6 (cont.) *Single-tank system*

Consider again the continuous-time model (3.8), (3.9), and assume the process can be subject to sensor and actuator faults denoted respectively f_s and f_a. Equations (3.8), (3.9) are modified as follows to account for such faults:

$$\begin{aligned} \frac{dh(t)}{dt} &= -\beta h(t) + \alpha u(t) + \alpha f_a(t) \\ y(t) &= h(t) + f_s(t). \end{aligned}$$

Assume now that a leakage at the bottom of the tank occurs. To account for this phenomenon, (3.4) becomes

$$\frac{dh(t)}{dt} = -k\sqrt{h(t)} - k_{leak}(t)\sqrt{h(t)} + \alpha u(t)\ . \tag{3.10}$$

If Eq. (3.10) is linearised around the nominal level h_0 and the nominal parameter $k_{leak,0} = 0$, the fault appears to be additive in the linear approximation. Indeed, the latter can be written as

$$\frac{dh(t)}{dt} = -\beta h(t) + \alpha u(t) - \sqrt{h_0}k_{leak}(t)\ . \qquad \square$$

3.5 System structure

Detailed behaviour models are seldom available in the first phases of system design, and/or are very expensive to develop, especially when complex processes, with hundreds of variables, are considered, and simpler models have to be used. In such situations, structural models provide an interesting approach to the system analysis, since they only need a very primitive level of knowledge about the system behaviour.

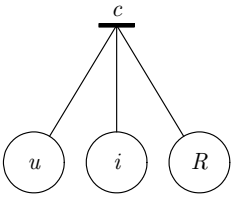
Structural model. The structural model of a system is an abstraction of its behaviour model. For continuous-variable systems, the behaviour is described by a set of algebraic and differential equations. Analysing the structure of these equations resumes to analysing the links which exist between variables and parameters, independently on the form of the underlying equations.

For example, consider a system described by Ohm's law

$$u - Ri = 0. \tag{3.11}$$

The structural model associated with this system says: "There exists one constraint (call it c), which links two variables (u, i) and one parameter (R)". It is represented

by a bi-partite graph $(\mathcal{C}, \mathcal{Z}, \mathcal{A})$ where $\mathcal{C}$ is the set of constraints, $\mathcal{Z}$ the set of variables or parameters, and $\mathcal{A}$ the set of edges between $\mathcal{C}$ and $\mathcal{Z}$ or, equivalently, by this graph's adjacency matrix, as shown on Fig. 3.2, where bars represent constraints and circles represent variables or parameters.



	u	i	R
c	1	1	1

Fig. 3.2. Structure of Ohm's law

Structural properties. Structural properties of a system are properties of its structure graph. Two systems which have the same structure are said to be structurally equivalent. This is possible, since the structure of a set of constraints is independent of the nature of these constraints, of the variables, and of the value of the parameters. Indeed, the structural model would be the same if, instead of Eq. (3.11), Ohm's law was expressed by a look-up table, or if another system, which obeys e.g. the numerical model $u(i^2 + 3i + 1) = R$, was considered.

Since structural properties are properties of the structure graph, they are obviously shared by all the systems which have the same structure. Thus, structural properties are properties of a system which are independent of the values of its parameters.

Known and unknown variables. Two kinds of variables appear in the system constraints, namely the known and the unknown ones. Therefore, the set of variables $\mathcal{Z}$ is decomposed into two subsets $\mathcal{Z} = \mathcal{X} \cup \mathcal{K}$. Control and measurement signals are known variables, while the systems states are unknown. Known variables obey measurement equations, which are introduced in the structural model. Assuming the voltage u is measured by a sensor whose output is y_1, the previous system obeys the two constraints

$$\begin{aligned} u - Ri &= 0 \\ y_1 - u &= 0 \end{aligned}$$

and, dropping the parameter R, its structure becomes

$\nearrow$	y_1	u	i
c	1	1	**1**
m_1	1	**1**	

which shows that all the unknown variables in the system can be computed, since u can be computed from y_1 by using the measurement equation m_1 (this is symbolised by the bold **1**), and therefore i can be computed from y_1 and u. Structural observability is indeed one of the properties that structural analysis allows to study. This is of course only a potential property, since constraints m_1 and c might be more complex ones, and the numerical computations might be impossible in some particular cases (change, for example, constraint m_1 into $y_1(1-u)=1$ and suppose that the known value of y_1 is zero!).

Faults. When faults occur, the system components do not any longer obey the equations which define their nominal behaviour. Therefore, a given fault mode is described in structural analysis by the subset of constraints which do no longer hold when this fault occurs. These constraints are said to be violated. For example, the short circuit of the previous resistor is described by constraint c being violated when the nominal value of R is used, while a malfunction of the voltage sensor would be described by constraint m_1 being violated.

3.6 Discrete-event systems

From a global viewpoint, some dynamical systems can be seen as systems whose signals switch from one value to another one rather than changing their value continously. In fault diagnosis, systems with discrete measurements occur naturally in the process industry where alarm messages represent discrete information, because the alarm can only be alerted or not and, hence, the corresponding signal is only known to exceed a given threshold or not. As the dynamical behaviour of such systems is described by events denoting the switches of the signal from one discrete value to the next, these systems are called discrete-event systems.

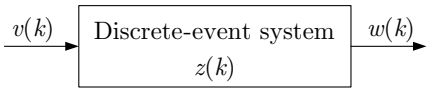


Fig. 3.3. Discrete-event dynamical system

Discrete-valued signals. Due to the symbolic nature of the input, state and output, the symbols v, z and w are used for them. The discrete value sets are enumerated such that

$$\begin{aligned} v &\in \mathcal{N}_v = \{1,2,...,M\} \\ z &\in \mathcal{N}_z = \{1,2,...,N\} \\ w &\in \mathcal{N}_w = \{1,2,...,R\} \end{aligned}$$

hold (Fig. 3.3). Every change of the symbolic value of v, z or w is called an *event*. For example, if the state jumps from the value j to the value i, a state event denoted by e_{ij} occurs. In Fig. 3.4 the events e_{13} and e_{32} are marked.

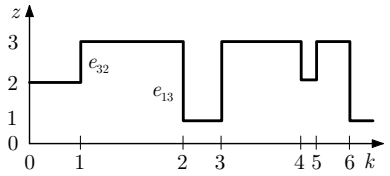


Fig. 3.4. Symbolic signal values and event sequence

The model that will be introduced now describes in which order the events occur but it says nothing about the temporal distance of these events. The sequences of discrete values that the input, state or output assume for a time horizon k_h are denoted by

$$\begin{aligned} V(0\,...\,k_h) &= (v(0),\, v(1),\, v(2),\, ...,\, v(k_h)) \\ Z(0\,...\,k_h) &= (z(0),\, z(1),\, z(2),\, ...,\, z(k_h)) \\ W(0\,...\,k_h) &= (w(0),\, w(1),\, w(2),\, ...,\, w(k_h)). \end{aligned}$$

In diagnosis, k_h denotes the current time instant and $V(0\,...\,k_h)$ and $W(0\,...\,k_h)$ the measured sequences to be processed.

Deterministic automata. A standard form for describing discrete-event systems is the deterministic automaton

$$\mathcal{A} = (\mathcal{N}_z,\, \mathcal{N}_v,\, \mathcal{N}_w,\, G,\, H,\, z_0),$$

which has the set of states $\mathcal{N}_z$, the input set $\mathcal{N}_v$ and the output set $\mathcal{N}_w$. G and H are the state transition function and the output function, which determine the successor state or output in the following way:

$$z(k+1) = G(z(k),\, v(k)), \qquad z(0) = z_0 \tag{3.12}$$

$$w(k) = H(z(k),\, v(k)). \tag{3.13}$$

z_0 is the initial state. k denotes the place that the input, state and output values have in the corresponding sequence.

Obviously, for a given initial state z_0 and input sequence $V(0\,...\,k_h)$ the state and output sequences $Z(0\,...\,k_h)$ and $W(0\,...\,k_h)$ can be generated by applying Eqs. (3.12) and (3.13) k_h times. The automaton is deterministic because the initial state and the input sequence unambiguously determine the state and output sequence.

Non-deterministic automaton. In the non-deterministic automaton

$$\mathcal{N}(\mathcal{N}_z,\, \mathcal{N}_v,\, \mathcal{N}_w,\, L_n,\, z_0)$$

the functions G and H of the deterministic automaton are replaced by the *behavioural relation* L_n

$$L_n : \mathcal{N}_z \times \mathcal{N}_w \times \mathcal{N}_z \times \mathcal{N}_v \;\in\; \{0,1\}$$

which for every given state $z(k)$ and input $v(k)$ describes which successor state $z(k+1)$ can be assumed while generating the output $w(k)$. Hence, the dynamical behaviour of the automaton is described by all 4-tuples for which

$$L_n(z(k+1),\, w(k),\, z(k),\, v(k)) = 1 \tag{3.14}$$

holds. Equation (3.14) replaces Eqs. (3.12), (3.13) of the deterministic automaton. Obviously, for z_0 and $V(0\,...\,k_h)$ the sequences $Z(0\,...\,k_h)$ and $W(0\,...\,k_h)$ are not unique.

If the probabilities with which the automaton assumes the different 4-tuples on the left-hand side of Eq. (3.14) are known, instead of the non-deterministic automaton a stochastic automaton

$$\mathcal{S} = (\mathcal{N}_z,\, \mathcal{N}_v,\, \mathcal{N}_w,\, L,\, \mathrm{Prob}(z(0)))$$

can be used to describe the discrete-event system. The *behavioural relation*

$$L : \mathcal{N}_z \times \mathcal{N}_w \times \mathcal{N}_z \times \mathcal{N}_v \;\longrightarrow\; [0,1]$$
$$L(z',\, w,\, z,\, v) = \mathrm{Prob}\,(z_p(1) = z',\, w_p(0) = w \mid z_p(0) = z,\, v_p(0) = v)$$

describes the probability that the automaton steps from state z towards state z' while generating the output w if it gets the input v. Hence, a probability measure can be associated with each state sequence Z and output sequence W.

Model of a faulty discrete-event system. In order to describe the behaviour of a discrete-event system under the influence of faults, the fault $f(k)$ is introduced as an additional discrete-valued input. The fault may change over time and, thus, generate the sequence

$$F(0\,...\,k_h) = (f(0),\, f(1),\, ...,\, f(k_h)).$$

The additional input f extends the stochastic automaton, which becomes

$$\mathcal{S} = (\mathcal{N}_z,\, \mathcal{N}_v,\, \mathcal{N}_f,\, \mathcal{N}_w,\, L,\, \mathrm{Prob}\,(z_0))$$

with $\mathcal{N}_f$ denoting the set of possible fault values. The behavioural relation L is now a function of five arguments:

$$L(z',\, w,\, z,\, f,\, v) =$$
$$\mathrm{Prob}\;(z_p(1) = z',\, w_p(0) = w | z_p(0) = z,\, f_p(0) = f,\, v_p(0) = v)\;\;.$$

Example 3.7 *Discrete-event model of the two-tank system*

The question whether a given system should be dealt with as a continous-variable or a discrete-event system depends not only on its properties but also on the task to be solved

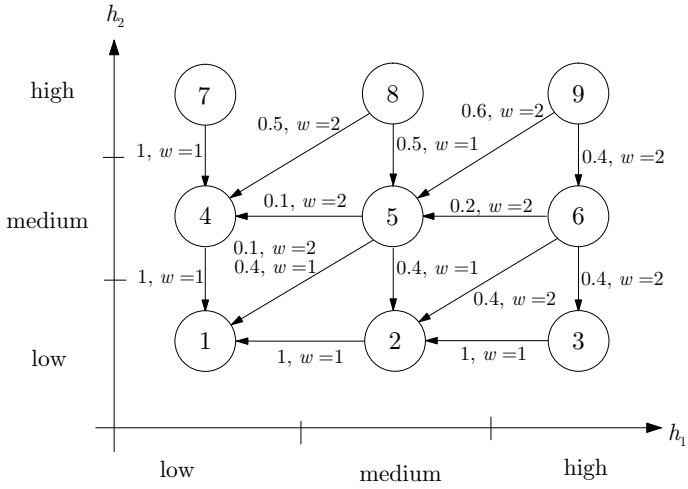


Fig. 3.5. Stochastic automaton describing the tank system for faulty pump ($q_P = 0$)

with the model. If for the two-tank system the tank levels should simply remain in a "high" region, it is sufficient to distinguish the levels "high", "medium" and "low" and to describe the behaviour of the system as a switching among these qualitative levels. The graph of the stochastic automaton describing this behaviour is depicted in Fig. 3.5. The automaton state $z = 1$ corresponds to the tank state $(h_1, h_2)'$, where both tank levels are "low", i.e. do not exceed a given threshold. The other states are defined in a similar way as illustrated by Fig. 3.5. The automaton graph is drawn for faulty pump (no inflow to Tank 1) which makes the input useless. The output $w = 1$ denotes a small and $w = 2$ a large outflow from Tank 2. The labels of the arcs describe the outflow together with the probability with which the state transition described by the arc occurs. The automaton says, for example, that if the tank system is in state 6 (h_1 is high, h_2 medium) then it assumes next one of the states 5, 2 or 3 and that it goes from state 6 towards state 2 while generating the output $w = 2$ with the probability 0.4. All paths through the automaton symbolise a possible state sequence Z and define an associated output sequence W. □

3.7 Hybrid systems

For many technological systems both continuous and discrete phenomena play important roles. The mixture of discrete and continuous signals and discrete and continuous forms of the models used is typical for supervisory control tasks and plays a particular role in diagnosis and fault-tolerant control. As the system possesses both real-valued and discrete-valued signals, combinations of differential equations and automata have to be used for its description (Fig. 3.6).

The main problem in dealing with hybrid systems result from the different range of the signals. The transition between these different ranges are represented by quan-

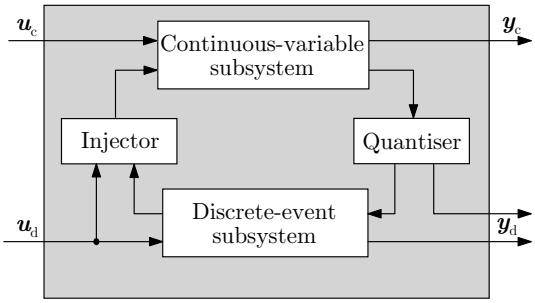


Fig. 3.6. Hybrid dynamical system

tisers and injectors. The *quantiser* transforms a real-valued signal into a sequence of symbols, where the real-valued signal or signal vector is denoted by a lower-case letter like y or $\boldsymbol{u}$ and the corresponding quantised signals by $[y]$ or $[\boldsymbol{u}]$. If, in the simplest case, the quantiser decides to which real interval of a given set of intervals the current value $y(t)$ belongs, the value of the quantised signal $[y(t)]$ at the same time instant t is the number of the corresponding interval. Clearly, this interval can be associated with symbolic names like "normal", "high" or "low", which give a semantic signal value. As long as the signal does not leave a given interval, the quantised value remains the same. Hence, a continuous change of $y(t)$ is transformed into a sequence of discrete changes of $[y(t)]$, where the quantiser does not only determine the symbolic value of the signal but also the time instants at which these symbolic values change.

The *injector* carries out the inverse mapping. Its input is a symbolic signal like $[\boldsymbol{u}]$, which is associated with a real-valued signal $\boldsymbol{u}$. The relation between $[\boldsymbol{u}]$ and $\boldsymbol{u}$ can be either deterministic where every symbolic value is associated with a unique real value or non-deterministic where the associated real value is randomly selected from a given set of signal values or may vary within this set as long as the symbolic value does not change. In any case, the injector is the interface between symbolic and real-valued signals.

A standard structure of hybrid systems is shown in Fig. 3.6. The system has continuous input and output (u_c and y_c) as well as discrete input and output (u_d and y_d), where the attribute "discrete" refers to the signal value. In addition to that, the system may be considered as a discrete-time system where all signals are known only at given sampling time instants.

Quantised systems, which are considered in Chapter 9 represent an important class of hybrid systems. These systems exhibit principal phenomena that characterise hybrid systems.

3.8 Links between the different models

Since different models can be built in order to describe the same system, there must be some relations between them. The aim of this section is to present and discuss those relations.

Relation among the models. The most important difference of the models introduced so far concerns the value set of the signals. Continuous-variable descriptions refer to signals with real signal values whereas discrete-event models use signals with discrete signals values. The question which model is appropriate for a particular application depends upon the question whether the continuous movement of the system or a sequence of discrete events generated by the given system have to be investigated for solving the given task. Therefore, a given system may be considered simultaneously as a continuous system or a discrete system if different problems have to be solved.

For example, a tank system has to be considered as a continuous system if the level of the tank or a concentration of a certain substance in the liquid filling the tank has to be controlled. Level or concentration controllers measure the numerical value of the level or the concentration with a given sampling rate and fix the control input to be applied at the next time instant. The same tank system may be considered as a discrete-event system if it is part of a batch process. Then a certain recipe is realised by imposing a discrete control sequence on the tank where the control command opens or closes valves to fill or empty the tank, heat or cool the liquid etc. The controller, which is usually a programmable logic controller reacts only on events, which are generated if the liquid or the temperature crosses given thresholds. The temporal distance of these events is of minor importance and, therefore, not described by the model.

Architecture and functions. Although functional models have not been developed in this chapter, it may be worth to discuss the link between architecture and function. The architecture model describes the system as a network of interconnected components. The reason why a given component belongs to the system is that it has been chosen to perform a specific function, in a given system operating mode. Thus, each service of a component is associated with a given function the component is expected to fulfil in some operating mode. For example, the "open" service of valve V_a in the tank example is associated with the function "increase the level in Tank T_2" when the level in Tank T_1 is higher than the level in Tank T_2 and higher than the level of the connecting pipe.

Architecture and behaviour. Components provide services which transform consumed variables into produced variables, according to some given procedure. Variables which are processed by services may be quantitative or qualitative. In any case, the procedures which describe the services of a component are nothing else

than constraints which link the values of the variables associated with this component. The temporal behaviour of these variables is thus defined once the procedures are given. Note that these procedures introduce algebraic and differential constraints for quantitative variables and discrete-event models for qualitative variables.

Example 3.8 *Different models of the tank system*

For example, the "open" service of valve V_{12} considered above introduces an algebraic constraint between the flow from tank T_1 to tank T_2 and the two levels h_1 and h_2

$$\begin{array}{ll} q_{12} = k(u)\sqrt{h_1 - h_2} & \text{if } h_1 \geq \max(h_{12}, h_2) \\ q_{12} = 0 & \text{if } \max(h_1, h_2) \leq h_{12} \\ q_{12} = -k(u)\sqrt{h_1 - h_2} & \text{if } h_2 \geq \max(h_{12}, h_1), \end{array}$$

where $k(u)$ is some coefficient which depends on u, the opening position of the valve, while the "close" service introduces the constraint

$$q_{12} = 0, \quad \forall\, h_1, h_2.$$

Also note that since the set of services (i.e. the set of constraints, and therefore the behaviour model) depends on the system operating mode, the generic component model directly introduces a hybrid model for the system description. In the valve example, three operating modes should be considered to describe the behaviour model, namely

$$\begin{array}{lcl} \text{valve is open} & \text{and} & \max(h_1, h_2) \leq h_{12} \\ \quad \text{then } q_{12} & = & 0 \\ \text{valve is open} & \text{and} & h_1 \geq \max(h_{12}, h_2) \text{ or } h_2 \geq \max(h_{12}, h_1) \\ \quad \text{then } q_{12} & = & \text{sign}(h_1 - h_2) k(u) \sqrt{|h_1 - h_2|} \\ \text{valve is closed} & & \\ \quad \text{then } q_{12} & = & 0. \end{array}$$

If the functions are considered, two different operating modes have to be associated with the situation $q_{12} \neq 0$, namely

$$\begin{array}{lcl} \text{valve is open} & \text{and} & h_1 \geq \max(h_{12}, h_2) \\ \quad \text{then } q_{12} & = & k(u) \sqrt{h_1 - h_2} \text{ and level } h_2 \text{ increases} \\ \text{valve is open} & \text{and} & h_2 \geq \max(h_{12}, h_1) \\ \quad \text{then } q_{12} & = & -k(u) \sqrt{h_2 - h_1} \text{ and level } h_2 \text{ decreases.} \end{array}$$

It can be checked that a discrete-event model of this system can be built by considering the following events

e_1: valve V_{12} opens
e_2: valve V_{12} closes
e_3: both levels h_1 and h_2 become lower than h_{12}
e_4: h_1 becomes higher than $\max(h_{12}, h_2)$
e_5: h_2 becomes higher than $\max(h_{12}, h_1)$ □

Behaviour and structure. The link between the behaviour model and the structural model is obvious, since the structural model is nothing but an abstraction of the behaviour model. In each operating mode, there is a set of constraints $\mathcal{C}$ which link the values of the system variables $\mathcal{Z} = \mathcal{X} \cup \mathcal{K}$. The structure of these constraints is

directly represented by the set of edges $\mathcal{A}$ in the bi-partite graph $(\mathcal{C}, \mathcal{Z}, \mathcal{A})$ whose nodes are respectively $\mathcal{C}$ and $\mathcal{Z}$.

Example 3.9 *Structure of a valve in different operation modes*

In the valve example, the are two different structure associated with the four different operating modes which appear on the hybrid description of the system behaviour:

- **Structure 1:** If the valve is open and $\max(h_1, h_2) \leq h_{12}$ or if the valve is closed, the following relation holds:

$\nearrow$	h_1	h_2	q_{12}	u
c_1				1
c_2			1	

  Constraint c_1 expresses that the control u is known, and constraint c_2 expresses that the flow q_{12} is also known (since $q_{12} = 0$).

- **Structure 2:** If the valve is open and $h_1 \geq \max(h_{12}, h_2)$ or $h_2 \geq \max(h_{12}, h_1)$,

$\nearrow$	h_1	h_2	q_{12}	u
c_1				1
c_2	1	1	1	1

  where constraint c_1 expresses that the control u is known, and constraint c_2 expresses the relation between the flow q_{12}, the control u, and the two levels $h_1$and h_2. □

3.9 Exercises

Exercise 3.1 *Model of ship dynamics*

Using the notation from Section 2.2, the dynamic model of a ship is

$$\begin{aligned} \dot{\omega}_3 &= b(\eta_1\omega_3 + \eta_3\omega_3^3) + b\delta \\ \dot{\psi} &= \omega_3 + \omega_w \\ y_1 &= \psi \\ y_2 &= \dot{\psi} \\ y_3 &= \delta \end{aligned}$$

1. Derive a linear model in state-space form, linearising about the point of operation
   $$\bar{\omega}_3 = \omega_o, \quad \bar{\psi} = \psi_o, \quad \bar{\delta} = \delta_o$$
2. Find also the model for the special case
   $$\bar{\omega}_3 = 0, \quad \bar{\psi} = 0, \quad \bar{\delta} = 0. \; \square$$

Exercise 3.2 *Model of industrial actuator*
A block diagram of an industrial actuator is shown in Fig. 3.7. It consists of the following components:

- DC motor with input current i and motor speed n
- power drive with known current command $i_{\rm com}$
- gear with gear ratio N efficiency η and output angle θ
- unknown load torque Q_l.

Measurements are θ_m the angle after the gear and n_m the shaft speed at the motor.

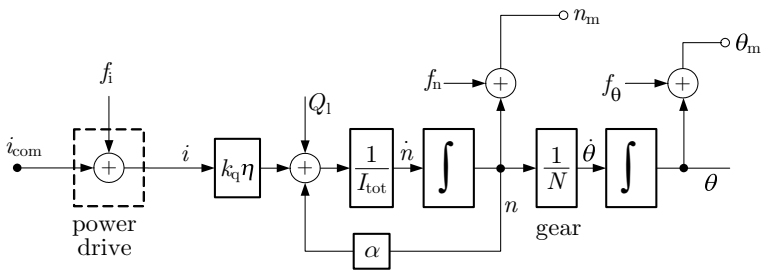


Fig. 3.7. Block diagram of actuator with additive faults - open loop

The faults concern

- f_θ – position sensor fault
- f_n – tachometer fault
- f_i – actuator fault.

With $\boldsymbol{x} = (n, \theta)'$, $u = i_{\rm com}$, $d = Q_l$, $\boldsymbol{f} = (f_i, f_n, f_\theta)'$, $\boldsymbol{y} = (n_m, \theta_m)'$, the actuator has the following state-space representation

$$\begin{aligned} \frac{d}{dt}\boldsymbol{x} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}u + \boldsymbol{E}_x d + \boldsymbol{F}_x \boldsymbol{f} \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{x} + \boldsymbol{F}_y \boldsymbol{f}. \end{aligned} \tag{3.15}$$

1. Show that
$$\boldsymbol{A} = \begin{pmatrix} -\frac{\alpha}{I_{tot}} & 0 \\ \frac{1}{N} & 0 \end{pmatrix}$$
and determine the remaining matrices in the state-space model.
2. Show that the system transfer function (Laplace domain) is
$$\begin{aligned} \boldsymbol{x}(s) &= (s\boldsymbol{I} - \boldsymbol{A})^{-1}(\boldsymbol{B}u(s) + \boldsymbol{E}_x d(s) + \boldsymbol{F}_x \boldsymbol{f}(s)) \\ \boldsymbol{y}(s) &= \boldsymbol{C}(s\boldsymbol{I} - \boldsymbol{A})^{-1}(\boldsymbol{B}u(s) + \boldsymbol{E}_x d(s) + \boldsymbol{F}_x \boldsymbol{f}(s)) + \boldsymbol{F}_y \boldsymbol{f}(s). \end{aligned}$$
3. Using the shorthand notation
$$\boldsymbol{y}(s) = \boldsymbol{H}_{yu}(s)u(s) + \boldsymbol{H}_{yd}(s)d(s) + \boldsymbol{H}_{yf}\boldsymbol{f}(s),$$
determine the three transfer function matrices $\boldsymbol{H}_{yu}, \boldsymbol{H}_{yd}$ and $\boldsymbol{H}_{yf}$. Verify what is apparent from the block diagram,
$$n_m(s) = \frac{1}{sI_{tot} + \alpha}(k_q\eta i_{\rm com}(s) + Q_l(s) + k_q\eta f_i(s)) + f_n(s). \ \square$$

Exercise 3.3 *Discrete-time model of industrial actuator*

The following parameters apply to the industrial actuator explained in Exercise 3.2: $k_q = 0.5\ \mathrm{Nm/A}$, $\eta = 0.8$, $N = 100$, $I_{tot} = 2 \cdot 10^{-3}\ \mathrm{kgm^2}$, $\alpha = 10^{-4}\ \mathrm{Nms/rad}$.

1. Determine the numerical transfer function matrices $\boldsymbol{H}_{yu}, \boldsymbol{H}_{yd}, \boldsymbol{H}_{yf}$. Find the values of gains and the location of poles and zeros.
2. Make a discrete-time model using a sampling time of 2 ms. Note values of gains, discrete-time (z-plane) poles and zeros in the discrete-time model.
3. Determine the steady-state properties of the change in measurement values when step-wise faults and disturbance are applied. Faults or load steps appear one at a time and are not simultaneously present. □

Exercise 3.4 *Industrial actuator with speed control*

Figure 3.8 shows an actuator with speed control, a limit in the maximum current from the power drive and measurement of motor current i_m. The speed controller is $i_{\mathrm{com}} = k_t(n_{\mathrm{ref}} - n_m)$. The power drive has a gain of 1 in the linear range $|i| \leq i_{\max}$, otherwise $i = i_{\max}$ sign (i_{com}).

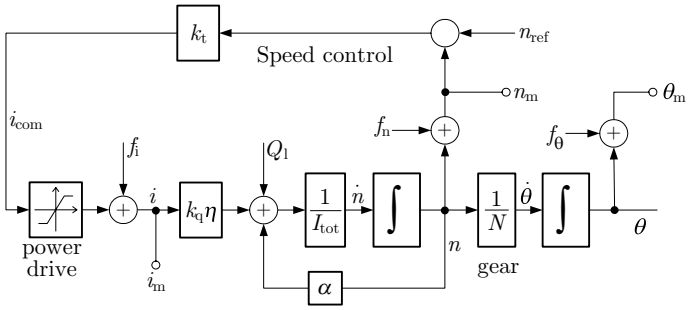


Fig. 3.8. Actuator with angular velocity feedback

1. Write a dynamic model of the actuator in the form

$$\frac{d}{dt}\begin{pmatrix} n \\ \theta \end{pmatrix} = \boldsymbol{A}_{cl}\begin{pmatrix} n \\ \theta \end{pmatrix} + \boldsymbol{B}_{cl} n_{\mathrm{ref}} + \boldsymbol{E}_{x,cl} Q_l + \boldsymbol{F}_{x,cl}\boldsymbol{f}$$

$$\begin{pmatrix} n_m \\ \theta_m \\ i_m \end{pmatrix} = \boldsymbol{C}_{cl}\begin{pmatrix} n \\ \theta \end{pmatrix} + \boldsymbol{D}_{cl} n_{\mathrm{ref}} + \boldsymbol{E}_{y,cl} Q_l + \boldsymbol{F}_{y,cl}\boldsymbol{f}$$

   and determine the elements of all parameter matrices $\boldsymbol{A}_{cl}$, $\boldsymbol{B}_{cl}$, ... in the model.
2. Implement a simulation of the continuous-time model of the actuator of Fig. 3.8. Use $k_t = 1.0\ \mathrm{As/rad}$ and $i_{\max} = \pm 20\ \mathrm{A}$ in the current limiter block in the simulation.
3. Validate that the responses to step-wise changes in reference, load torque and fault signals and compare with those of the theoretical model. □

Exercise 3.5 *Model of a coffee machine*
Describe the steps to produce a coffee with milk by means of a coffee maschine by a deterministic automaton. How can this automaton be extended to hybrid model if the differential equations describing the continuous processes are associated to some of the automaton states? □

3.10 Bibliographical notes

Modular and object-oriented models have been developed to describe architectures of automated systems [2], [110], [165]. Such models are used in the description and the validation of real-time and distributed systems [253], and specific tools, like state charts have been developed to describe their real-time operation [90]. Generic component models are architecture models, first developed for the description of intelligent sensors and actuators [225], [230]. They have been used for the interoperability analysis of distributed architectures [1], [9], [11], [30]. There exists a bridge between SyncCharts and generic component models, which provides a means of analysing both the system architecture and its associated real-time behaviour [8].

Behaviour models based on "first principles" describe the power exchanges and transformations which take place in a process. Bond graphs, first developed in [197], describe power as the product of an effort (e.g. voltage, pressure, force) and a flow (e.g. current, volume flow, velocity), and use a graphical representation to describe the exchanges of power between different components of a process. They provide a unified modelling approach for different engineering disciplines, since power is a concept which is shared by all of them. Bond graph modelling has been further developed in [112] and [252], and recently extended to thermal and chemical engineering in [251]. Bond graph models are used for simulation and control design, as well as for the design of fault detection and isolation algorithms [247].

For an introduction into the wide field of system identification, the reader is referred to the monographs [132], [221] and [261].

Good introductions to discrete-event systems are [39] and [148].