# 2

# Actuator and Sensor Fault-tolerant Control Design

## 2.1 Introduction

Many industrial systems are complex and nonlinear. When it is not easy to deal with the nonlinear models, systems are usually described by linear or linearized models around operating points. This notion of operating point is very important when a linearized model is considered, but it is not always easily understood. The objective in this chapter is to highlight the way to seek an operating point and to show a complete procedure which includes the identification step, the design of the control law, the FDI, and the FTC. In addition to the detailed approach dealing with linearized systems around an operation point, a nonlinear approach will be presented.

## 2.2 Plant Models

### 2.2.1 Nonlinear Model

Many dynamical system can be described either in continuous-time domain by differential equations:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)) \\ y(t) = h(x(t), u(t)) \end{cases}, \tag{2.1}$$

or in discrete-time domain by recursive equations:

$$\begin{cases} x(k+1) = f(x(k), u(k)) \\ y(k) = h(x(k), u(k)) \end{cases}, \tag{2.2}$$

where $x \in \Re^n$ is the state vector, $u \in \Re^m$ is the control input vector, and $y \in \Re^q$ is the system output vector. $f$ and $h$ are nonlinear functions.

These forms are much more general than their standard linear counterparts which are described in the next section.

There is a particular class of nonlinear systems – named input-linear or affine systems – which is often considered, as many real systems can be described by these equations:

$$\begin{cases} \dot{x}(t) = f(x(t)) + \sum_{j=1}^{m} (g_j(x(t))u_j(t)) \\ \qquad\quad = f(x(t)) + G(x(t))u(t) \\ y_i(t) = h_i(x(t)), \qquad\qquad 1 \le i \le q \end{cases} . \tag{2.3}$$

$f(x)$ and $g_j(x)$ can be represented in the form of $n$-dimensional vector of real-valued functions of the real variables $x_1, \ldots, x_n$, namely

$$f(x) = \begin{bmatrix} f_1(x_1, \ldots, x_n) \\ f_2(x_1, \ldots, x_n) \\ \vdots \\ f_n(x_1, \ldots, x_n) \end{bmatrix}; \qquad g_j(x) = \begin{bmatrix} g_{1j}(x_1, \ldots, x_n) \\ g_{2j}(x_1, \ldots, x_n) \\ \vdots \\ g_{nj}(x_1, \ldots, x_n) \end{bmatrix}. \tag{2.4}$$

Functions $h_1, \ldots, h_q$ which characterize the output equation of system (2.3) may be represented in the form

$$h_i(x) = h_i(x_1, \ldots, x_n). \tag{2.5}$$

The corresponding discrete-time representation is

$$\begin{cases} x(k+1) = f_d(x(k)) + \sum_{i=j}^{m} (g_{d_j}(x(k))u_j(k) \\ \qquad\quad = f_d(x(k)) + G_d(x(k))u(k) \\ y(k) = h_d(x(k)) \end{cases} . \tag{2.6}$$

### 2.2.2 Linear Model: Operating Point

An operating point is usually defined as an equilibrium point. It has to be chosen first when one has to linearize a system. The obtained linearized model corresponds to the relationship between the *variation of the system output* and the *variation of the system input* around this operating point. Let us consider a system associated with its actuators and sensors, with the whole range of the operating zone of its inputs $U$ and measurements $Y$ (Fig. 2.1).

If the system is linearized around an operating point $(U_0, Y_0)$, the linearized model corresponds to the relationship between the variations of the system inputs $u$ and outputs $y$ (Fig. 2.2) such that

$$u = U - U_0 \qquad \text{and} \qquad y = Y - Y_0. \tag{2.7}$$

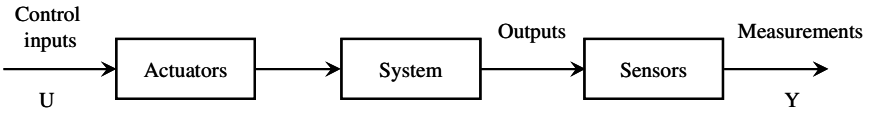


**Fig. 2.1.** System representation considering the whole operating zone

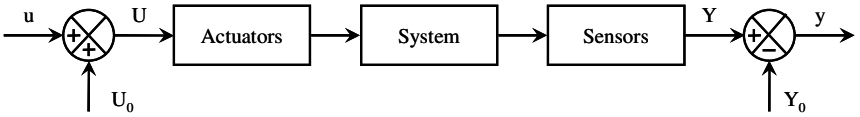


**Fig. 2.2.** System representation taking into account the operating point

Then the model describing the relationship between the input $u$ and the output $y$ can be given by a Laplace transfer function for single-input single-output (SISO) systems:

$$\Theta(s) = \frac{y(s)}{u(s)}, \tag{2.8}$$

or by a state-space representation given in continuous-time for SISO or multiple-input multiple-output (MIMO) systems:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}, \tag{2.9}$$

where $x \in \Re^n$ is the state vector, $u \in \Re^m$ is the control input vector, and $y \in \Re^q$ is the output vector. $A$, $B$, $C$, and $D$ are matrices of appropriate dimensions.

Very often, in real applications where a digital processor is used (microcontroller, programmable logic controller, computer, and data acquisition board, *etc.*), it may be more convenient to consider a discrete-time representation:

$$\begin{cases} x(k+1) = A_d x(k) + B_d u(k) \\ y(k) = C_d x(k) + D_d u(k) \end{cases}. \tag{2.10}$$

$A_d$, $B_d$, $C_d$, and $D_d$ are the matrices of the discrete-time system of appropriate dimensions.

In the sequel, linear systems will be described in discrete-time, whereas nonlinear systems will be considered in continuous-time. For the simplicity of notation and without loss of generality, matrix $D_d$ is taken as a zero matrix, and the subscript $d$ is removed.

### 2.2.3 Example: Linearization Around an Operating Point

To illustrate the notion of the operating point, let us consider the following example. In the tank presented in Fig. 2.3, the objective is to study the behavior of the water level $L$ and the outlet water temperature $T_o$. An inlet flow

rate $Q_i$ is feeding the tank. An electrical power $P_u$ is applied to an electrical resistor to heat the water in the tank.

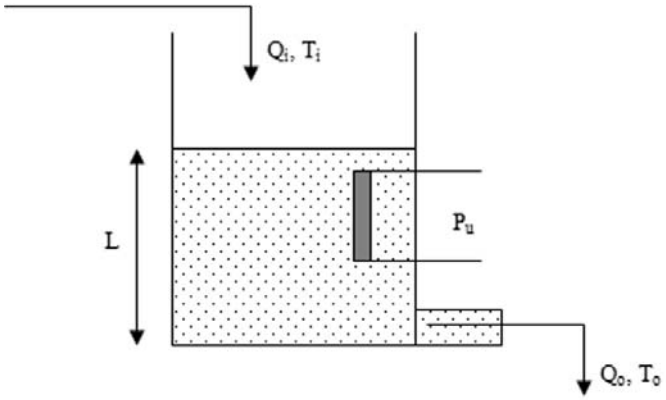


**Fig. 2.3.** Tank with heater

- $Q_i$ is the inlet water flow rate
- $T_i$ is the inlet water temperature considered as constant
- $Q_o$ is the outlet water flow rate
- $T_o$ is the outlet water temperature
- $L$ is the water level in the tank
- $P_u$ is the power applied to the electrical resistor
- $S$ is the cross section of the tank

The outputs of this MIMO system are $L$ and $T_o$. The control inputs are $Q_i$ and $P_u$. The block diagram of this system is given in Fig. 2.4.

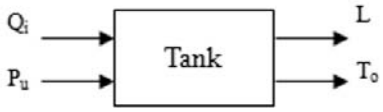


**Fig. 2.4.** The input/output block diagram

Assuming that the outlet flow rate $Q_o$ is proportional to the square root of the water level in the tank ($Q_o = \alpha\sqrt{L}$), the water level $L$ will be given by the following nonlinear differential equation:

$$\frac{dL(t)}{dt} = \frac{1}{S}(Q_i(t) - Q_o(t)) = \frac{1}{S}(Q_i(t) - \alpha\sqrt{L(t)}). \tag{2.11}$$

Based on the thermodynamics equations, the outlet water temperature $T_o$ is described by the following nonlinear differential equation:

$$\frac{dT_o(t)}{dt} = \frac{P_u(t)}{SL(t)\mu c} - \frac{T_o(t) - T_i(t)}{SL(t)} Q_i(t), \tag{2.12}$$

where $c$ is the specific heat capacity, and $\mu$ is the density of the water.

The objective now is to linearize these equations around a given operating point: $OP = (Q_{i0}, P_{u0}, Q_{o0}, T_{o0}, L_0)$. Around this operating point, the system variables can be considered as

$$\begin{aligned} &Q_i(t) = Q_{i0} + q_i(t); \quad P_u(t) = P_{u0} + p_u(t); \quad Q_o(t) = Q_{o0} + q_o(t); \\ &T_o(t) = T_{o0} + t_o(t); \quad L(t) = L_0 + l_o(t). \end{aligned} \tag{2.13}$$

The linearization of (2.11) around the operating point $OP$ is given by

$$\frac{dl(t)}{dt} = \frac{1}{S}\, q_i(t) - \frac{\alpha}{2S\sqrt{L_0}}\, l(t). \tag{2.14}$$

Similarly, the linearization of (2.12) around the operating point $OP$ is given by

$$\begin{aligned} \frac{dt_o(t)}{dt} = &-\frac{T_{o0} - T_i}{SL_0}\, q_i(t) + \frac{1}{SL_0\mu c}\, p_u(t) \\ &- \frac{1}{L_0^2}\left(\frac{P_{u0}}{S\mu c} - \frac{T_{o0} - T_i}{S}\, Q_{i0}\right) l(t) \\ &- \frac{Q_{i0}}{SL_0}\, t_o(t). \end{aligned} \tag{2.15}$$

Considering the following state vector $x = \begin{bmatrix} l & t_o \end{bmatrix}^T$, the linearized state-space representation of this system around the operating point is then given by

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} \dot{l}(t) \\ \dot{t}_o(t) \end{bmatrix} = \begin{bmatrix} -\dfrac{\alpha}{2S\sqrt{L_0}} & 0 \\ a & b \end{bmatrix} \begin{bmatrix} l(t) \\ t_o(t) \end{bmatrix} + \begin{bmatrix} \dfrac{1}{S} & 0 \\ c & d \end{bmatrix} \begin{bmatrix} q_i(t) \\ p_u(t) \end{bmatrix} \\ y(t) = \begin{bmatrix} l(t) \\ t_o(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} l(t) \\ t_o(t) \end{bmatrix} \end{cases}, \tag{2.16}$$

where

$$a = -\frac{1}{L_0^2}\left(\frac{P_{u0}}{S\mu c} - \frac{T_{o0} - T_i}{S}\, Q_{i0}\right); \qquad b = -\frac{Q_{i0}}{SL_0};$$

$$c = -\frac{T_{o0} - T_i}{SL_0}; \qquad d = \frac{1}{SL_0\mu c}.$$

**Numerical Application**

Study the response of the system to the *variation of the input variables* as follows: $q_i = 10\ l/h$ and $p_u = 2\ kW$. The numerical values of the system lead to the following state-space representation:

$$\dot{x}(t) = \begin{bmatrix} \dot{l}(t) \\ \dot{t}_o(t) \end{bmatrix} = \begin{bmatrix} -0.01 & 0 \\ 0 & -0.02 \end{bmatrix} \begin{bmatrix} l(t) \\ t_o(t) \end{bmatrix} + \begin{bmatrix} 0.01 & 0 \\ -0.03 & 0.04 \end{bmatrix} \begin{bmatrix} q_i(t) \\ p_u(t) \end{bmatrix}. \quad (2.17)$$

The simulation results of the linearized system in response to the variation of the system inputs in open-loop around the operating point are shown in Fig. 2.5.

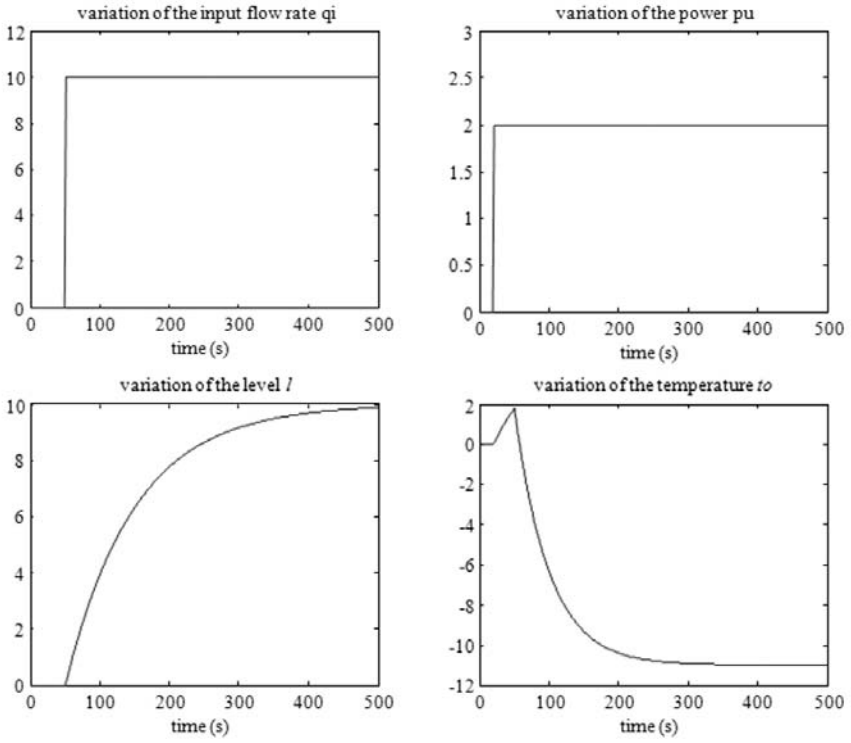


**Fig. 2.5.** The variation of the system inputs/outputs

It can be seen that initial values of these variables are zero. The zero here corresponds to the value of the operating point. However, the real variables $Q_i$, $P_u$, $L$, and $T_o$ are shown in Fig. 2.6.

Later on, if a state-feedback control has to be designed, it should be based on the linearized equations given by (2.16).

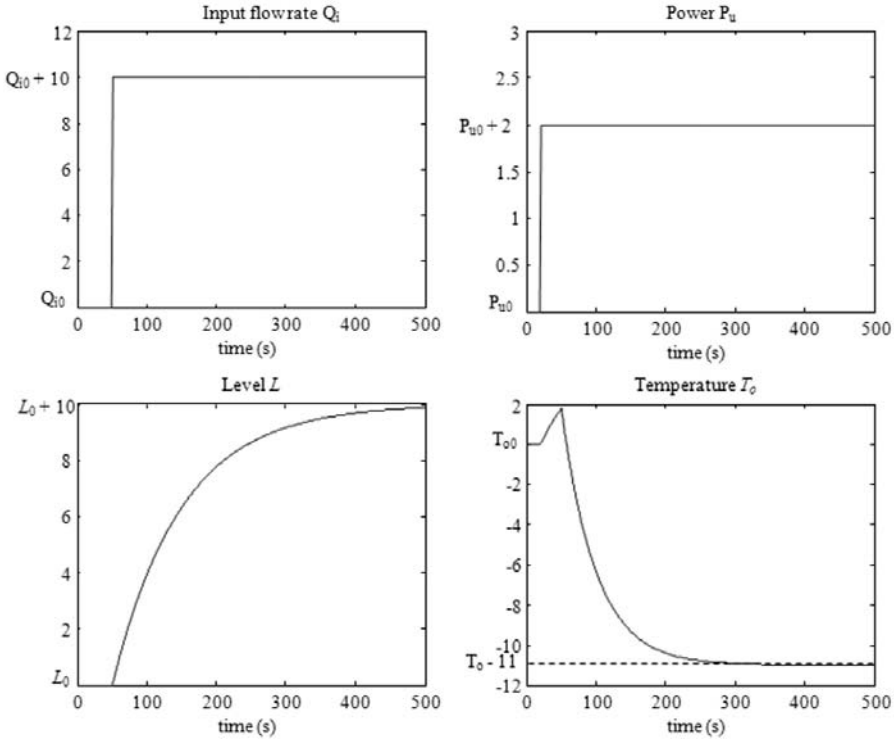


**Fig. 2.6.** The actual values of the system inputs/outputs

## 2.3 Fault Description

During the system operation, faults or failures may affect the sensors, the actuators, or the system components. These faults can occur as additive or multiplicative faults due to a malfunction or equipment aging.

For FDI, a distinction is usually made between additive and multiplicative faults. However, in FTC, the objective is to compensate for the fault effect on the system regardless of the nature of the fault.

The faults affecting a system are often represented by a variation of system parameters. Thus, in the presence of a fault, the system model can be written as

$$\begin{cases} x_f(k+1) = A_f x_f(k) + B_f u_f(k) \\ y_f(k) = C_f x_f(k) \end{cases}, \tag{2.18}$$

where the new matrices of the faulty system are defined by

$$A_f = A + \delta A; \quad B_f = B + \delta B; \quad C_f = C + \delta C. \tag{2.19}$$

$\delta A$, $\delta B$, and $\delta C$ correspond to the deviation of the system parameters with respect to the nominal values. However, when a fault occurs on the system, it is very difficult to get these new matrices on-line.

Process monitoring is necessary to ensure effectiveness of process control and consequently a safe and a profitable plant operation. As presented in the next paragraph, the effect of actuator and sensor faults can also be represented as an additional unknown input vector acting on the dynamics of the system or on the measurements.

The effect of actuator and sensor faults can also be represented using an unknown input vector $f_j \in \Re^l, j = a$ (for actuators), $s$ (for sensors) acting on the dynamics of the system or on the measurements.

### 2.3.1 Actuator Faults

It is important to note that an actuator fault corresponds to the variation of the global control input $U$ applied to the system, and not only to $u$:

$$U_f = \Gamma U + U_{f0}, \tag{2.20}$$

where

- $U$ is the global control input applied to the system
- $U_f$ is the global faulty control input
- $u$ is the variation of the control input around the operating point $U_0$, ($u = U - U_0$, $u_f = U_f - U_0$ )
- $U_{f0}$ corresponds to the effect of an additive actuator fault
- $\Gamma U$ represents the effect of a multiplicative actuator fault

with $\Gamma = diag(\alpha)$, $\alpha = \begin{bmatrix} \alpha_1 & \cdots & \alpha_i & \cdots & \alpha_m \end{bmatrix}^T$ and $U_{f0} = \begin{bmatrix} u_{f01} & \cdots & u_{f0i} & \cdots & u_{f0m} \end{bmatrix}^T$. The $i^{\text{th}}$ actuator is faulty if $\alpha_i \neq 1$ or $u_{f0i} \neq 0$ as presented in Table 2.1 where different types of actuator faults are described.

**Table 2.1.** Actuator fault

| | Constant offset $u_{f0i} = 0$ | Constant offset $u_{f0i} \neq 0$ |
|---|---|---|
| $\alpha_i = 1$ | Fault-free case | Bias |
| $\alpha_i \in ]0;1[$ | Loss of effectiveness | Loss of effectiveness |
| $\alpha_i = 0$ | Out of order | Actuator blocked |

In the presence of an actuator fault, the linearized system (2.10) can be given by

$$\begin{cases} x(k+1) = Ax(k) + B(\Gamma U(k) + U_{f0} - U_0) \\ y(k) = Cx(k) \end{cases}. \tag{2.21}$$

The previous equation can also be written as

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + B[(\Gamma - I)U(k) + U_{f0}] \\ y(k) = Cx(k) \end{cases}. \tag{2.22}$$

By defining $f_a(k)$ as an unknown input vector corresponding to actuator faults, (2.18) can be represented as follows:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + F_a f_a(k) \\ y(k) = Cx(k) \end{cases}, \tag{2.23}$$

where $F_a = B$ and $f_a = (\Gamma - I)U + U_{f0}$. If the $i^{\text{th}}$ actuator is declared to be faulty, then $F_a$ corresponds to the $i^{\text{th}}$ column of matrix $B$ and $f_a$ corresponds to the magnitude of the fault affecting this actuator.

In the nonlinear case and in the presence of actuator faults, (2.3) can be described by the following continuous-time state-space representation:

$$\begin{cases} \dot{x}(t) = f(x(t)) + \sum_{j=1}^{m} (g_j(x(t))u_j(t)) + \sum_{j=1}^{m} (F_{a,j}(x(t))f_{a,j}(t)) \\ y_i(t) = h_i(x(t)) \qquad\qquad 1 \leq i \leq q \end{cases}, \tag{2.24}$$

where $F_{a,j}(x(t))$ corresponds to the $j^{\text{th}}$ column of matrix $G(x(t))$ in (2.3) and $f_{a,j}(t)$ corresponds to the magnitude of the fault affecting the $j^{\text{th}}$ actuator.

### 2.3.2 Sensor Faults

In a similar way, considering $f_s$ as an unknown input illustrating the presence of a sensor fault, the linear faulty system will be represented by

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) + F_s f_s(k) \end{cases}. \tag{2.25}$$

The affine nonlinear systems can be defined in continuous-time through an additive component such as

$$\begin{cases} \dot{x}(t) = f(x(t)) + G(x(t))u(t) \\ y_i(t) = h_i(x(t)) + F_{s,i} f_{s,i}(t) \end{cases}, \quad 1 \leq i \leq q, \tag{2.26}$$

where $F_{s,i}$ is the $i^{\text{th}}$ row of matrix $F_s$ and $f_{s,i}$ is the fault magnitude affecting the $i^{\text{th}}$ sensor.

This description of actuator and sensor faults is a structured representation of these faults. Matrices $F_a$ and $F_s$ are assumed to be known and $f_a$ and $f_s$ correspond, respectively, to the magnitudes of the actuator fault and the sensor fault.

An FTC method is based on a nominal control law associated with a fault detection and estimation, and a modification of this control law. This is used in order to compensate for the fault effect on the system.

## 2.4 Nominal Tracking Control Law

The first step in designing an FTC method is the setup of a nominal control. In the sequel, a multi-variable linear tracking control is first addressed, then a case of nonlinear systems is presented.

### 2.4.1 Linear Case

The objective in this section is to describe a nominal tracking control law able to make the system outputs follow pre-defined reference inputs.

Consider a MIMO system given by the following discrete-time state-space representation:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases}, \tag{2.27}$$

where $x \in \Re^n$ is the state vector, $u \in \Re^m$ is the control input vector, and $y \in \Re^q$ is the output vector. $A$, $B$, and $C$ are matrices of appropriate dimensions.

The tracking control law requires that the number of outputs to be controlled must be less than or equal to the number of the control inputs available on the system [29].

If the number of outputs is greater than the number of control inputs, the designer of the control law selects the outputs that must be tracked and breaks down the output vector $y$ as follows:

$$y(k) = Cx(k) = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} x(k) = \begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix}. \tag{2.28}$$

The feedback controller is required to cause the output vector $y_1 \in \Re^p$ ($p \leq m$) to track the reference input vector $y_r$ such that in steady-state:

$$y_r(k) - y_1(k) = 0. \tag{2.29}$$

To achieve this objective, a comparator and integrator vector $z \in \Re^p$ is added to satisfy the following relation:

$$\begin{cases} z(k+1) = z(k) + T_s(y_r(k) - y_1(k)) \\ \qquad\quad = z(k) + T_s(y_r(k) - C_1 x(k)) \end{cases}, \tag{2.30}$$

where $T_s$ is the sample period to be chosen properly. Careful consideration should be given to the choice of $T_s$. If $T_s$ is too small, the processor will not

have enough time to calculate the control law. The system may be unstable if $T_s$ is too high because the system is operating in open-loop during a sample period.

The open-loop system is governed by the augmented state and output equations, where $I_p$ is an identity matrix of dimension $p$ and $0_{n,p}$ is a null matrix of $n$ rows and $p$ columns:

$$\begin{cases} \begin{bmatrix} x(k+1) \\ z(k+1) \end{bmatrix} = \begin{bmatrix} A & 0_{n,p} \\ -T_s C_1 & I_p \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix} + \begin{bmatrix} B \\ 0_{p,m} \end{bmatrix} u(k) + \begin{bmatrix} 0_{n,p} \\ T_s I_p \end{bmatrix} y_r(k) \\ y(k) = \begin{bmatrix} C & 0_{q,p} \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix} \end{cases}. \tag{2.31}$$

This state-space representation can also be written in the following form:

$$\begin{cases} X(k+1) = \bar{A}X(k) + \bar{B}_u u(k) + \bar{B}_r y_r(k) \\ y(k) = \bar{C}X(k) \end{cases}. \tag{2.32}$$

The nominal feedback control law of this system can be computed by

$$u(k) = -KX(k) = -\begin{bmatrix} K_1 & K_2 \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix}. \tag{2.33}$$

$K = \begin{bmatrix} K_1 & K_2 \end{bmatrix}$ is the feedback gain matrix obtained, for instance, using a pole placement technique, linear-quadratic (LQ) optimization, and so on [6,78, 119,125,130,133]. To achieve this control law, the state variables are assumed to be available for measurement. Moreover, the state-space considered here is that where the outputs are the state variables ($C$ is the identity matrix $I_n$). Otherwise, the control law is computed using the estimated state variables obtained, for instance, by an observer or a Kalman filter.

Figure 2.7 summarizes the design of the nominal tracking control taking into account the operating point with $x = y_1$.

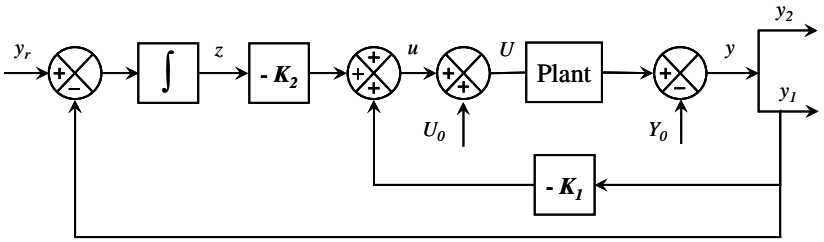


**Fig. 2.7.** Nominal tracking control taking into account the operating point

### 2.4.2 Nonlinear Case

The need for nonlinear control theory arises from the fact that systems are nonlinear in practice. Although linear models are simple and easy to analyze, they are not valid except around a certain operating region. Outside this region, the linear model is not valid and the linear representation of the process is insufficient. Control of nonlinear systems has been extensively considered in the literature where plenty of approaches have been proposed for deterministic, stochastic, and uncertain nonlinear systems (see for instance [46, 52, 58, 116]). In this book, two control methods are used: the exact input-output linearization and the sliding mode controller (SMC).

#### Exact Linearization and Decoupling Input-Output Controller

According to the special class of input-linear systems given by (2.3), a nonlinear control law is commonly established to operate in closed-loop. To perform this task, an exact linearization and decoupling input-output law via a static state-feedback $u(t) = \alpha(x(t)) \ + \ \beta(x(t))\ v(t)$ is designed. It is assumed here that the system has as many outputs as inputs (*i.e.*, $q = m$). For the general case where $q \neq m$, the reader can refer to [41, 73, 98].

The aim of this control law is to transform (2.3) into a linear and controllable system based on the following definitions

**Definition 2.1.** *Let* $(r_1, r_2, \ldots, r_m)$ *be the set of the relative degree per row of (2.3) such as*

$$r_i = \{min\ l \in \aleph / \exists j \in [1, \ldots, m], L_{g_j} L_f^{l-1} h_i(x(t)) \neq 0\}, \tag{2.34}$$

*where* $L$ *is the Lie derivative operator.*

The Lie derivative of $h_i$ in the direction of $f$, denoted $L_f h_i(x)$, is the derivative of $h_i$ in $t = 0$ along the integral curve of $f$, such that

$$L_f h_i(x) = \sum_{j=1}^{n} f_j(x) \frac{\partial h_i}{\partial x_j}(x). \tag{2.35}$$

The operation $L_f$, Lie derivative in the direction of $f$, can be iterated. $L_f^k h$ is defined for any $k \geq 0$ by

$$L_f^0 h(x) = h(x) \ \text{ and } \ L_f^k h(x) = L_f(L_f^{k-1} h(x)) \ \ \forall k \geq 1\,. \tag{2.36}$$

**Definition 2.2.** *If all* $r_i$ *exist* $(i = 1, \ldots, m)$*, the following matrix* $\Delta$ *is called "decoupling matrix" of (2.3):*

$$\Delta(x) = \begin{bmatrix} L_{g_1} L_f^{r_1-1} h_1(x) & \cdots & L_{g_m} L_f^{r_1-1} h_1(x) \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{r_m-1} h_m(x) & \cdots & L_{g_m} L_f^{r_m-1} h_m(x) \end{bmatrix}. \tag{2.37}$$

*A vector $\Delta_0$ is also defined such as*

$$\Delta_0(x) = \begin{bmatrix} L_f^{r_1} h_1(x) \\ \vdots \\ L_f^{r_m} h_m(x) \end{bmatrix}. \tag{2.38}$$

According to the previous definition, the nonlinear control is designed as follows.

**Theorem:**
*a) The system defined by (2.3) is statically decouplable on a subset $M_0$ of $\Re^n$ if and only if*

$$\text{rank } \Delta(x) = m, \qquad \forall x \in M_0. \tag{2.39}$$

*b) The control law computed using the state-feedback is defined by*

$$u(t) = \alpha(x(t)) + \beta(x(t))v(t), \tag{2.40}$$

*where*

$$\begin{cases} \alpha(x) = -\Delta^{-1}(x)\Delta_0(x) \\ \beta(x) = \Delta^{-1}(x) \end{cases}. \tag{2.41}$$

*This control law is able to decouple (2.3) on $M_0$.*
*c) This closed-loop system has a linear input-output behavior described by*

$$y_i^{(r_i)}(t) = v_i(t), \qquad \forall i \in [1, \ldots, m], \tag{2.42}$$

*where $y_i^{(r_i)}(t)$ is the $r_i^{th}$ derivative of $y_i$.*

Two cases may be observed:

- $\sum_{i=1}^{m} r_i = n$: the closed-loop system characterized by the $m$ decoupled linear subsystems is linear, controllable and observable.
- $\sum_{i=1}^{m} r_i < n$: a subspace made unobservable by the nonlinear feedback (2.40). The stability of the unobservable subspace must be studied. This subspace must have all modes stable. More details about this case can be found in [41, 73, 98].

Since each SISO linear subsystem is equivalent to a cascade of integrators, a second feedback control law should be considered in order to stabilize and to set the performance of the controlled nonlinear system. This second feedback

is built using linear control theory [29]. The simplest feedback consists of using a pole placement associated with $\tau_i$ such as

$$\frac{y_i(s)}{y_{ref,i}(s)} = \frac{1}{(1+\tau_i s)^{r_i}} \tag{2.43}$$

where $y_{ref,i}$ is the reference input associated with output $y_i$.

The advantage of this approach is that the feedback controllers are designed independently of each other. Indeed, nonlinear feedback (2.40) is built from model (2.3) according to the theorem stated previously. The stabilized feedback giving a closed-loop behavior described by (2.43) is designed from the $m$ decoupled linear equivalent subsystems (2.42) written in the Brunovsky canonical form [65] such as

$$\begin{cases} \dot{z}_i(t) = A_i z_i(t) + B_i v_i(t) \\ y_i(t) = C_i z_i(t) \end{cases}, \qquad \forall i \in [1, \ldots, m], \tag{2.44}$$

with

$$A_i = \begin{bmatrix} 0 & & \\ \vdots & I_{r_i-1} & \\ 0 & & \\ 0 & 0 \quad \cdots \quad 0 & \end{bmatrix}, \qquad B_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \qquad C_i = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}. \tag{2.45}$$

The link between both state-feedbacks is defined by a diffeomorphism $z(t) = \Phi(x(t))$ where $z(t)$ is the state vector of the decoupled linear system written in the controllability canonical form.

When there is no unobservable state subspace, the diffeomorphism is defined by

$$z(t) = \Phi(x(t)) = \begin{bmatrix} \Phi_1(x(t)) \\ \vdots \\ \vdots \\ \vdots \\ \Phi_m(x(t)) \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} h_1(x(t)) \\ \vdots \\ L_f^{r_1-1} h_1(x(t)) \end{bmatrix} \\ \vdots \\ \begin{bmatrix} h_m(x(t)) \\ \vdots \\ L_f^{r_m-1} h_m(x(t)) \end{bmatrix} \end{bmatrix}. \tag{2.46}$$

The exact linearization and decoupling input-output controller with both state-feedback control laws may be illustrated in Fig. 2.8.

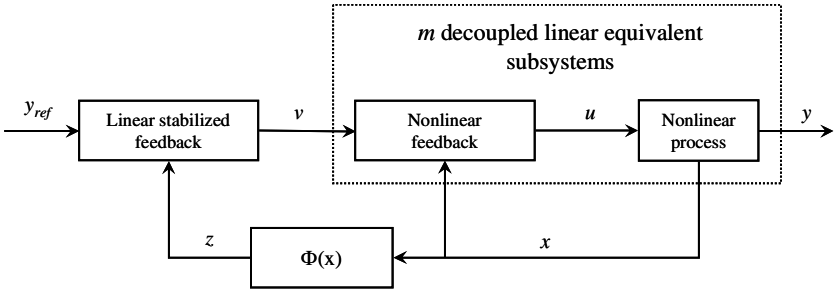


**Fig. 2.8.** Nonlinear control scheme

### Sliding Mode Controller

The main advantage of the SMC over the other nonlinear control laws is its robustness to external disturbances, model uncertainties, and variations in system parameters [135, 136]. In order to explain the SMC concept, consider a SISO second order input affine nonlinear system:

$$\ddot{x} = f(x, \dot{x}) + g(x, \dot{x})u + d_f, \tag{2.47}$$

where $u$ is the control input and $d_f$ represents the uncertainties and external disturbances which are assumed to be upper bounded with $|d_f| < D$. Note that this section considers continuous-time systems but the time index is omitted for simplicity. Defining the state variables as $x_1 = x$ and $x_2 = \dot{x}$, (2.47) leads to

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f(x, \dot{x}) + g(x, \dot{x})u + d_f \end{cases}. \tag{2.48}$$

If the desired trajectory is given as $x_1^d$, then the error between the actual $x_1$ and the desired trajectory $x_1^d$ can be written as

$$e = x_1 - x_1^d. \tag{2.49}$$

The time derivative of $e$ is given by

$$\dot{e} = \dot{x}_1 - \dot{x}_1^d = x_2 - x_2^d. \tag{2.50}$$

The switching surface $s$ is conventionally defined for second order systems as a combination of the error variables $e$ and $\dot{e}$:

$$s = \dot{e} + \lambda e, \tag{2.51}$$

where $\lambda$ sets the dynamics in the sliding phase ($s = 0$).

The control input $u$ should be chosen so that trajectories approach the switching surface and then stay on it for all future time instants. Thus, the time derivative of $s$ is given by

$$\dot{s} = f(x, \dot{x}) + g(x, \dot{x})u + d_f - \ddot{x}_1{}^d + \lambda \dot{e}. \tag{2.52}$$

The control input is expressed as the sum of two terms [114]. The first one, called the equivalent control, is chosen using the nominal plant parameters ($d_f = 0$), so as to make $\dot{s} = 0$ when $s = 0$. It is given by [114]

$$u_{eq} = g(x, \dot{x})^{-1}(\ddot{x}_1{}^d - f(x, \dot{x}) - \lambda \dot{e}). \tag{2.53}$$

The second term is chosen to tackle the uncertainties in the system and to introduce a reaching law; the constant ($Msign(s)$) plus proportional ($ks$) rate reaching law is imposed by selecting the second term as [114]

$$u^* = g(x, \dot{x})^{-1}[-ks - Msign(s)], \tag{2.54}$$

where $k$ and $M$ are positive numbers to be selected and $sign(.)$ is the *signum* function. The function $g(x, \dot{x})$ must be invertible for (2.53) and (2.54) to hold.

Then, the control input $u = u_{eq} + u^*$ becomes

$$u = g(x, \dot{x})^{-1}[\ddot{x}_1{}^d - f(x, \dot{x}) - \lambda \dot{e} - ks - Msign(s)]. \tag{2.55}$$

Substituting input $u$ of (2.55) in (2.52) gives the derivative $\dot{s}$ of the sliding surface

$$\dot{s} = -ks - Msign(s) + d_f. \tag{2.56}$$

The necessary condition for the existence of conventional sliding mode for (2.47) is given by

$$\frac{1}{2}\frac{d}{dt}s^2 < 0, \quad \text{or} \quad s\dot{s} < 0. \tag{2.57}$$

This condition states that the squared distance to the switching surface, as measured by $s^2$, decreases along all system trajectories. However, this condition is not feasible in practice because the switching of real components is not instantaneous and this leads to an undesired phenomenon known as chattering in the direction of the switching surface. Thus (2.57) is expanded by a boundary layer in which the controller switching is not required:

$$s\dot{s} < -\eta \left| s \right|. \tag{2.58}$$

Multiplying (2.56) by $s$ yields

$$s\dot{s} = -ks^2 - Msign(s)s + d_f s = -ks^2 - M \left| s \right| + d_f s. \tag{2.59}$$

With a proper choice of $k$ and $M$, (2.58) will be satisfied.

The elimination of the chattering effect produced by the discontinuous function *sign* is ensured by a saturation function *sat*. This saturation function is defined as follows:

$$sat(s) = \begin{cases} sign(s) & \text{if } |s| > \phi_s \\ s/\phi_s & \text{if } |s| < \phi_s \end{cases}, \tag{2.60}$$

where $\phi_s$ is a boundary layer around the sliding surface $s$.

## 2.5 Model-based Fault Diagnosis

After designing the nominal control law, it is important to monitor the behavior of the system in order to detect and isolate any malfunction as soon as possible. The FDI allows us to avoid critical consequences and helps in taking appropriate decisions either on shutting down the system safely or continuing the operation in degraded mode in spite of the presence of the fault.

The fault diagnostic problem from raw data trends is often difficult. However, model-based FDI techniques are considered and combined to supervise the process and to ensure appropriate reliability and safety in industry. The aim of a diagnostic procedure is to perform two main tasks: fault detection, which consists of deciding whether a fault has occurred or not, and fault isolation, which consists of deciding which element(s) of the system has (have) indeed failed. The general procedure comprises the following three steps:

- Residual generation: the process of associating, with the measured and estimated output pair $(y, \widehat{y})$, features that allow the evaluation of the difference, denoted $r$ ($r = y - \widehat{y}$), with respect to normal operating conditions
- Residual evaluation: the process of comparing residuals $r$ to some predefined thresholds according to a test and at a stage where symptoms $S(r)$ are produced
- Decision making: the process of deciding through an indicator, denoted $I$ based on the symptoms $S(r)$, which elements are faulty (*i.e.*, isolation)

This implies the design of residuals $r$ that are close to zero in the fault-free situations ($f = 0$), while they will clearly deviate from zero in the presence of faults ($f \neq 0$). They will possess the ability to discriminate between all possible modes of faults, which explains the use of the term isolation. A short historical review of FDI can also be found in [71] and current developments are reviewed in [44].

While a single residual may be enough to detect a fault, a set of structured residuals is required for fault isolation . In order to isolate a fault, some residuals with particular sensitivity properties are established. This means that $r = 0$ if $f^* = 0$ and $r \neq 0$ if $f^* \neq 0$ regardless of the other faults defined through $f^d = 0$. In this context, in order to isolate and to estimate both actuator and sensor faults, a bank of structured residuals is considered where each residual vector $r$ may be used to detect a fault according to a statistical test. Consequently, it involves the use of statistical tests such as the Page-Hinkley test, limit checking test, generalized likelihood ratio test, and trend analysis test [8].

An output vector of the statistical test, called *coherence vector* $S_r$, can then be built from the bank of $\nu$ residual generators:

$$S_r = [S(||r_1||) \cdots S(||r_\nu||)]^T, \tag{2.61}$$

where $S(||r_j||)$ represents a symptom associated with the norm of the residual vector $r_j$. It is equal to 0 in the fault-free case and set to 1 when a fault is detected.

The coherence vector is then compared to the fault signature vector $S_{ref,f_j}$ associated with the $j^{\text{th}}$ fault according to the residual generators built to produce a signal sensitive to all faults except one as represented in Table 2.2.

**Table 2.2.** Fault signature table

| $S_r$ | No faults | $S_{ref,f_1}$ | $S_{ref,f_2}$ | $\cdots$ | $S_{ref,f_\nu}$ | Other faults |
|---|---|---|---|---|---|---|
| $S(\|r_1\|)$ | 0 | 0 | 1 | $\cdots$ | 1 | 1 |
| $S(\|r_2\|)$ | 0 | 1 | 0 | $\cdots$ | 1 | 1 |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ | $\vdots$ |
| $S(\|r_\nu\|)$ | 0 | 1 | 1 | $\cdots$ | 0 | 1 |

The decision is then made according to an elementary logic test [86] that can be described as follows: an indicator $I(f_j)$ is equal to 1 if $S_r$ is equal to the $j^{\text{th}}$ column of the incidence matrix ($S_{ref,f_j}$) and otherwise it is equal to 0. The element associated with the indicator equals to 1 is then declared to be faulty.

Moreover, the FDI module can also be exploited in order to estimate the fault magnitude.

Based on a large diversity of advanced model-based methods for automated FDI [22,31,53,69], the problem of actuator and/or sensor fault detection and magnitude estimation for both linear time-invariant (LTI) and nonlinear systems has been considered in the last few decades. Indeed, due to difficulties inherent in the on-line identification of closed-loop systems, parameter estimation techniques are not considered in this book. The parity space technique is suitable to distinguish between different faults in the presence of uncertain parameters, but is not useful for fault magnitude estimation.

In this section, the FDI problem is first considered, then in Sect. 2.6 the fault estimation is treated before investigating the FTC problem in Sect. 2.7.

### 2.5.1 Actuator/Sensor Fault Representation

Let us recall the state-space representation of a system that may be affected by actuator and/or sensor fault:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + F_a f_a(k) \\ y(k) = Cx(k) + F_s f_s(k) \end{cases}, \tag{2.62}$$

where matrices $F_a$ and $F_s$ are assumed to be known and $f_a$ and $f_s$ correspond to the magnitude of the actuator and the sensor faults, respectively. The

magnitude and time occurrence of the faults are assumed to be completely unknown.

In the presence of sensor and actuator faults, (2.62) can also be represented by the unified general formulation

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + F_x f(k) \\ y(k) = Cx(k) + F_y f(k) \end{cases}, \tag{2.63}$$

where $f = [f_a^T \quad f_s^T]^T \in \Re^\nu \quad (\nu = m+q)$ is a common representation of sensor and actuator faults. $F_x \in \Re^{n\times\nu}$ and $F_y \in \Re^{q\times\nu}$ are respectively the actuator and sensor faults matrices with $F_x = [B \quad 0_{n\times q}]$ and $F_y = [0_{q\times m} \quad I_q]$.

The objective is to isolate faults. This is achieved by generating residuals sensitive to certain faults and insensitive to others, commonly called structured residuals . The fault vector $f$ in (2.63) can be split into two parts. The first part contains the "$d$" faults to be isolated $f^0 \in \Re^d$. In the second part, the other "$\nu - d$" faults are gathered in a vector $f^* \in \Re^{\nu-d}$. Then, the system can be written by the following equations:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + F_x^0 f^0(k) + F_x^* f^*(k) \\ y(k) = Cx(k) + F_y^0 f^0(k) + F_y^* f^*(k) \end{cases}. \tag{2.64}$$

Matrices $F_x^0$, $F_x^*$, $F_y^0$, and $F_y^*$, assumed to be known, characterize the distribution matrices of $f^*$ and $f^0$ acting directly on the system dynamics and on the measurements, respectively.

As indicated previously, an FDI procedure is developed to enable the detection and the isolation of a particular fault $f^0$ among several others. In order to build a set of residuals required for fault isolation, a residual generation using an unknown input decoupled scheme is considered such that the residuals are sensitive to fault vector $f^*$ and insensitive to $f^0$. Only a single fault (actuator or sensor fault) is assumed to occur at a given time, because simultaneous faults can hardly be isolated. Hence, vector $f^0$ is a scalar ($d = 1$) and it is considered as an unknown input. It should be noted that the necessary condition of the existence of decoupled residual generator is fulfilled according to Hou and Muller [66]: the number of unknown inputs must be less than the number of measurements ($d \leq q$).

In case of an $i^{\text{th}}$ actuator fault, the system can be represented according to (2.64) by

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + B_i f^0(k) + [\overline{B}_i \quad 0_{n\times q}] f^*(k) \\ y(k) = Cx(k) + [0_{q\times(p-1)} \quad I_q] f^*(k) \end{cases}, \tag{2.65}$$

where $B_i$ is the $i^{\text{th}}$ column of matrix $B$ and $\overline{B}_i$ is matrix $B$ without the $i^{\text{th}}$ column.

In order to generate a unique representation, (2.65) can be described as:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + F_d f_d(k) + F_x^* f^*(k) \\ y(k) = Cx(k) + F_y^* f^*(k) \end{cases}, \tag{2.66}$$

where $f^0$ is denoted as $f_d$.

Similarly, for a $j^{\text{th}}$ sensor fault, the system is described as follows:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + [B \;\; 0_{n\times(q-1)}] f^*(k) \\ y(k) = Cx(k) + E_j f^0(k) + [0_{q\times p} \;\; \overline{E}_j] f^*(k) \end{cases}, \tag{2.67}$$

where $E_j = [0 \cdots 1 \cdots 0]^T$ represents the $j^{\text{th}}$ sensor fault effect on the output vector and $\overline{E}_j$ is the identity matrix without the $j^{\text{th}}$ column.

According to Park *et al.* [100], a system affected by a sensor fault can be written as a system represented by an actuator fault. Assume the dynamic of a sensor fault is described as

$$f^0(k+1) = f^0(k) + T_s \xi(k), \tag{2.68}$$

where $\xi$ defines the sensor error input and $T_s$ is the sampling period.

From (2.67) and (2.68), a new system representation including the auxiliary state can be introduced:

$$\begin{cases} \begin{bmatrix} x(k+1) \\ f^0(k+1) \end{bmatrix} = \begin{bmatrix} A & 0_{n\times 1} \\ 0_{1\times n} & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ f^0(k) \end{bmatrix} + \begin{bmatrix} B \\ 0_{1\times m} \end{bmatrix} u(k) + \begin{bmatrix} 0_{n\times 1} \\ T_s \end{bmatrix} \xi(k) \\ \qquad\qquad + \begin{bmatrix} B & 0_{n\times(q-1)} \\ 0_{1\times m} & 0_{1\times(q-1)} \end{bmatrix} f^*(k) \\ y(k) = \begin{bmatrix} C & E_j \end{bmatrix} \begin{bmatrix} x(k) \\ f^0(k) \end{bmatrix} + \begin{bmatrix} 0_{q\times m} & \overline{E}_j \end{bmatrix} f^*(k) \end{cases}. \tag{2.69}$$

Consequently, for actuator or sensor faults representation ((2.65) and (2.69)), a unique state-space representation can be established to describe the faulty system as follows:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) + F_d f_d(k) + F_x^* f^*(k) \\ y(k) = Cx(k) + F_y^* f^*(k) \end{cases}, \tag{2.70}$$

where $f_d$ is the unknown input vector. For simplicity, the same notation for vectors and matrices has been used in (2.66) and (2.70).

Under the FTC framework, once the FDI module indicates which sensor or actuator is faulty, the fault magnitude should be estimated and a new control law will be set up in order to compensate for the fault effect on the system.

As sensor and actuator faults have different effects on the system, the control law should be modified according to the nature of the fault. In this book, only one fault is assumed to occur at a given time. The presence of simultaneous multiple faults is still rare, and the FDI problem in this case is considered as a specific topic and is dealt with in the literature. Here, the objective is to deal with a complete FTC problem for a single fault.

### 2.5.2 Residual Generation

#### Unknown Input Observer – Linear Case

Based on the previous representation, several approaches have been suggested by [43, 53] to generate a set of residuals called structural residuals, in order to detect and isolate the faulty components. The theory and the design of unknown input observers developed in [22] is considered in this book due to the fact that a fault magnitude estimation can be generated but also that the unknown observers concept can be extended to nonlinear systems. A full-order observer is built as follows:

$$\begin{cases} w(k+1) = Ew(k) + TBu(k) + Ky(k) \\ \widehat{x}(k) = w(k) + Hy(k) \end{cases}, \tag{2.71}$$

where $\widehat{x}$ is the estimated state vector and $w$ is the state of this full-order observer. $E$, $T$, $K$, and $H$ are matrices to be designed for achieving unknown input decoupling requirements. The state estimation error vector ($e = \widehat{x} - x$) of the observer goes to zero asymptotically, regardless of the presence of the unknown input in the system. The design of the unknown input observer is achieved by solving the following equations:

$$(HC - I)F_d = 0, \tag{2.72}$$

$$T = I - HC, \tag{2.73}$$

$$E = A - HCA - K_1C, \tag{2.74}$$

$$K_2 = EH, \tag{2.75}$$

and

$$K = K_1 + K_2. \tag{2.76}$$

$E$ must be a stable matrix in order to guarantee a state error estimation equal to zero.

The system defined by (2.71) is an unknown input observer for the system given by (2.70) if the necessary and sufficient conditions are fulfilled:

- $\text{Rank}(CF_d) = \text{rank}(F_d)$
- $(C, A_1)$ is a detectable pair, where $A_1 = E + K_1C$

If these conditions are fulfilled, an unknown input observer provides an estimation of the state vector, used to generate a residual vector $r(k) = y(k) - C\widehat{x}(k)$ independent of $f_d(k)$. This means that $r(k) = 0$ if $f^*(k) = 0$ and $r(k) \neq 0$ if $f^*(k) \neq 0$ for all $u(k)$ and $f_d(k)$.

**Unknown Input Observer – Affine Case**

Among all algebraic methods, several methods consist of the generation of fault decoupling residual for special class of nonlinear systems such as bilinear systems [80]. Other methods focus more on general nonlinear systems where an unknown input decoupling input-output model is obtained [138]. Exact fault decoupling for nonlinear systems is also synthesized with geometric approach by [60, 101]. A literature review is detailed in [81].

Consider the state-space representation of the affine system affected by an actuator fault:

$$\begin{cases} \dot{x}(t) = f(x(t)) + \sum_{j=1}^{m} g_j(x(t))u_j(t) + \sum_{j=1}^{m} F_j(x(t))f_j(t) \\ y_i(t) = h_i(x(t)), \qquad 1 \leq i \leq q \end{cases} . \tag{2.77}$$

The approach presented in this section is an extension of the synthesis of unknown input linear observers to affine nonlinear systems. The initial work on this problem can be found in [49, 50].

The original system described by (2.77) should be broken down into two subsystems where one subsystem depends on the fault vector $f$ and the second is independent of $f$ by means of a diffeomorphism $\Phi_f$ such as

$$\begin{cases} \dot{\widetilde{x}}_1(t) = \widetilde{f}_1(\widetilde{x}_1(t), \widetilde{x}_2(t)) + \sum_{j=1}^{m} \widetilde{g}_{1j}(\widetilde{x}_1(t), \widetilde{x}_2(t))u_j(t) \\ \qquad\qquad + \sum_{j=1}^{m} \widetilde{F}_j(\widetilde{x}_1(t), \widetilde{x}_2(t))f_j(t) \\ \dot{\widetilde{x}}_2(t) = \widetilde{f}_2(\widetilde{x}_1(t), \widetilde{x}_2(t)) + \sum_{j=1}^{m} \widetilde{g}_{2j}(\widetilde{x}_1(t), \widetilde{x}_2(t))u_j(t) \end{cases} , \tag{2.78}$$

where $\widetilde{x}(t) = \begin{bmatrix} \widetilde{x}_1(t) \\ \widetilde{x}_2(t) \end{bmatrix} = \Phi_f(x(t), u(t))$.

The diffeomorphism $\Phi_f$ is defined by

$$\sum_{j=1}^{m} \frac{\partial}{\partial x_j(t)} \Phi_f(x(t), u(t)) \times F_j(x(t))f_j(t) = 0 \ . \tag{2.79}$$

This transformation is solved using the Frobenius theorem [73]. Equation (2.79) is not always satisfied. In order to simplify the way to solve this transformation, only one component $f_j(t)$ of the fault vector $f$ is considered with the objective of building a bank of observers. Each observer is dedicated to one single fault $f_j$ as proposed in the generalized observer scheme.

A subsystem insensitive to a component $f_j$ of the fault vector $f(t)$ is extracted for each observer by deriving the output vector $y(t)$. A characteristic index is associated with each fault $f_j$. This index corresponds to the necessary derivative number so that the fault $f_j$ appears in $y_i$. This index is also called *the detectability index* and is defined by

$$\rho_i = \min\{\zeta \in \mathbb{N} | L_F L_g^{\zeta-1} h_i(x(t)) \neq 0\}. \tag{2.80}$$

If $\rho_i$ exists, only component output $y_i$ is affected by $f_j$. It is then possible to define a new state-space representation where a subsystem is insensitive to fault $f_j$, such as

$$\widetilde{x}(t) = \Phi_{f_j}(x(t), u(t)) = \begin{bmatrix} \widetilde{x}_1(t) \\ \widetilde{x}_2(t) \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} y_i(t) \\ \dot{y}_i(t) \\ \vdots \\ y_i^{\rho_i - 1}(t) \end{bmatrix} \\ \phi_i(x(t), u(t)) \end{bmatrix}. \tag{2.81}$$

It is always possible to find $\phi_i(x(t), u(t))$ satisfying the following conditions [41]:

$$\operatorname{rank}\left( \frac{\partial}{\partial x} \begin{bmatrix} \begin{bmatrix} y_i(t) \\ \dot{y}_i(t) \\ \vdots \\ y_i^{\rho_i - 1}(t) \end{bmatrix} \\ \phi_i(x(t), u(t)) \end{bmatrix} \right) = \dim(x(t)), \tag{2.82}$$

where $\dfrac{d}{dt}\left(\phi_i(x(t), u(t))\right)$ is independent of $f_j(t)$.

System (2.77) can now be written by means of the new coordinates system defined in (2.81) and a subsystem insensitive to $f_j$ can be represented as

$$\begin{cases} \dot{\widetilde{x}}_1(t) = \phi_i(\widetilde{x}_1(t), \widetilde{x}_2(t), u(t)) \\ \widetilde{y}_i(t) = \widetilde{h}_i(\widetilde{x}_1(t), \widetilde{x}_2(t)) \end{cases}, \tag{2.83}$$

where $\widetilde{y}_i(t)$ is the output vector $y(t)$ without the $i^{\text{th}}$ component $y_i(t)$. $\widetilde{x}_2(t)$ is considered as an input vector for (2.83).

Considering all the components of the fault vector $f(t)$, a bank of observers is built where each observer is insensitive to a unique fault $f_j$. Nonlinear subsystem (2.83), which is insensitive to $f_j$, is used in order to synthesize a nonlinear observer as an extended Luenberger observer [97].

Based on [100], the proposed decoupled observer method applied to an affine system also provides an efficient FDI technique for sensor faults as developed in the linear case. In the presence of a sensor fault, the observer insensitive to the fault estimates state vector $\widetilde{x}_1(t)$ and consequently estimates the output corrupted by the fault. On the other hand, no estimation of an actuator fault can be computed from (2.83).

**Fault Diagnosis Filter Design**

Some control methods such as observers have been considered or modified to solve FDI problems. Among various FDI methods, filters have been successfully considered to provide new tools to detect and isolate faults.

To detect and estimate the fault magnitude, a fault detection filter is designed such that it does not decouple the residuals from the fault but rather assigns the residuals vector in particular directions to guarantee the identification of the fault [28, 79, 122].

Under the condition that $(A, C)$ is observable from (2.66) or (2.70), the projectors are designed such that the residual vector is sensitive only to a particular fault direction. In order to determine the fault magnitude and the state vector estimations, a gain is synthesized such that the residual vector $r(k) = y(k) - C\widehat{x}(k)$ is insensitive to specific faults according to some projectors $P$. These projectors are designed such that the projected residual vector $p(k) = Pr(k)$ is sensitive only to a particular fault direction. Hence, the specific fault filter is defined as follows:

$$\begin{cases} \widehat{x}(k+1) = A\widehat{x}(k) + Bu(k) + (K_A + K_C)\left(y(k) - C\widehat{x}(k)\right) \\ \quad \widehat{y}(k) = C\widehat{x}(k) \end{cases}, \tag{2.84}$$

where

- $K_A$ should be defined in order to obtain $AF_d - K_ACF_d = 0$, so $K_A$ is equivalent to
$$K_A = \omega\Xi, \tag{2.85}$$
$\omega = AF_d$, $\Xi = (CF_d)^+$ and $+$ defines the pseudo-inverse
- $K_C$ should be defined in order to obtain $K_CCF_d = 0$ which is solved as follows:
$$K_C = K\Psi, \tag{2.86}$$
where $\Psi = \beta\left[I - (CF_d)(CF_d)^+\right]$ and $K$ is a constant gain

It must be noted that $\beta$ is chosen as a matrix with appropriate dimensions whose elements are equal to 1. The reduced gain $K$ defines the unique free parameter in this specific filter.

Based on (2.85) and (2.86), (2.84) becomes equivalent to the following:

$$\begin{cases} \widehat{x}(k+1) = (\mathcal{A} - K\mathcal{C})\,\widehat{x}(k) + Bu(k) + K_Ay_k + K\Psi y_k \\ \quad \widehat{y}(k) = C\widehat{x}(k) \end{cases}, \tag{2.87}$$

where $\mathcal{A} = A\left[I - F_d\Xi C\right]$ and $\mathcal{C} = \Psi C$.

The gain $K$ is calculated using the eigenstructure assignment method such that $(\mathcal{A} - K\mathcal{C})$ is stable.

The gain breakdown $K_A + K_C$ and associated definitions involve the following matrices properties:

$$\Xi C F_d = 0, \qquad \text{and} \qquad \Psi C F_d = I, \tag{2.88}$$

and enable the generation of projected residual vector as follows:

$$p(k) = Pr(k) = \begin{bmatrix} \Psi \\ \Xi \end{bmatrix} r(k) = \begin{bmatrix} \Sigma r(k) \\ \Xi r(k) + f_d(k-1) \end{bmatrix} = \begin{bmatrix} \gamma(k) \\ \eta(k) \end{bmatrix}. \tag{2.89}$$

It is worth noting that $\gamma$ is a residual insensitive to faults and $\eta$ is calculated in order to be sensitive to $f_d$.

As sensor and actuator faults do not affect the system similarly, the control law should be modified according to the nature of the fault. In the sequel, different methods for estimating the actuator and sensor faults are presented.

## 2.6 Actuator and Sensor Faults Estimation

### 2.6.1 Fault Estimation Based on Unknown Input Observer

According to the fault isolation, the fault magnitude estimation of the corrupted element is extracted directly from the $j^{\text{th}}$ unknown input observer which is built to be insensitive to the $j^{\text{th}}$ fault ($f^*(k) = 0$). Based on the unknown input observer, the substitution of the state estimation in the faulty description (2.70) leads to

$$F_d f_d(k) = \widehat{x}(k+1) - A\widehat{x}(k) - Bu(k). \tag{2.90}$$

In the presence of an actuator fault, $F_d$ is a matrix of full column rank. Thus, the estimation of the fault magnitude $\widehat{f_0}(k) = \widehat{f_d}(k)$ makes use of the singular-value decomposition (SVD) [54].

Let $F_d = U \begin{bmatrix} R \\ 0 \end{bmatrix} V^T$ be the SVD of $F_d$. Thus, $R$ is a diagonal and non-singular matrix and $U$ and $V$ are orthonormal matrices.

Using the SVD and substituting it in (2.90) results in

$$\overline{\widehat{x}}(k+1) = \overline{A\widehat{x}}(k) + \overline{B\widehat{u}}(k) + \begin{bmatrix} R \\ 0 \end{bmatrix} V^T f_d(k), \tag{2.91}$$

where

$$\widehat{x}(k) = U\overline{\widehat{x}}(k) = U \begin{bmatrix} \overline{\widehat{x}}_1(k) \\ \overline{\widehat{x}}_2(k) \end{bmatrix}, \tag{2.92}$$

$$\overline{A} = U^{-1} A U = \begin{bmatrix} \overline{A}_{11}(k) & \overline{A}_{12}(k) \\ \overline{A}_{21}(k) & \overline{A}_{22}(k) \end{bmatrix}, \tag{2.93}$$

and

$$\overline{B} = U^{-1}B = \begin{bmatrix} \overline{B}_1(k) \\ \overline{B}_2(k) \end{bmatrix}. \tag{2.94}$$

Based on (2.91), the estimation of the actuator fault magnitude is defined as

$$\widehat{f}^0(k) = \widehat{f}_d(k) = VR^{-1}(\overline{\widehat{x}}_1(k+1) - \overline{A}_{11}\overline{\widehat{x}}_1(k) - \overline{A}_{12}\overline{\widehat{x}}_2(k) - \overline{B}_1 u(k)). \tag{2.95}$$

For a sensor fault, the fault estimation $\widehat{f}^0(k)$ is the last component of the estimated augmented state vector $\widehat{x}(k)$ as defined in (2.69).

### 2.6.2 Fault Estimation Based on Decoupled Filter

Based on the projected residual $p(k)$, an estimation of input vector $\eta(k)$ (which corresponds to the fault magnitude with a delay of one sample) should be directly exploited for fault detection. Indeed, a residual evaluation algorithm can be performed by the direct fault magnitude evaluation through a statistical test in order to monitor the process. It should be highlighted that the first component of projector vector (2.89), denoted $\gamma(k)$, can be considered as a quality indicator of the FDI module. If a fault is not equal to $f_d$ then the mean of the indicator will not equal zero. As previously, sensor fault estimation can be also provided by the last component of the augmented state-space .

### 2.6.3 Fault Estimation Using Singular Value Decomposition

Another method to estimate the actuator and sensor faults is based on SVD which will be described in this section.

#### Estimation of Actuator Faults

In the presence of an actuator fault and according to (2.23) and (2.31), the augmented state-space representation of the system is written as

$$\begin{cases} \begin{bmatrix} x(k+1) \\ z(k+1) \end{bmatrix} = \begin{bmatrix} A & 0_{n,p} \\ -T_s C_1 & I_p \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix} + \begin{bmatrix} B \\ 0_{p,m} \end{bmatrix} u(k) \\ \qquad\qquad + \begin{bmatrix} 0_{n,p} \\ T_s I_p \end{bmatrix} y_r(k) + \begin{bmatrix} F_a \\ 0 \end{bmatrix} f_a(k) \quad , \\ y(k) = \begin{bmatrix} C & 0_{q,p} \end{bmatrix} \begin{bmatrix} x(k) \\ z(k) \end{bmatrix} \end{cases} \tag{2.96}$$

where $F_a$ corresponds to the $i^{\text{th}}$ column of matrix $B$ in case the $i^{\text{th}}$ actuator is faulty.

The magnitude of the fault $f_a$ can be estimated if it is defined as a component of an augmented state vector $\overline{X}_a(k)$. In this case, the system (2.96) can be re-written under the following form:

$$\overline{E}_a\overline{X}_a(k+1) = \overline{A}_a\overline{X}_a(k) + \overline{B}_a\overline{U}(k) + \overline{G}_a y_r(k), \tag{2.97}$$

where

$$\overline{E}_a = \begin{bmatrix} I_n & 0 & -F_a \\ 0 & I_p & 0 \\ C & 0 & 0 \end{bmatrix}; \quad \overline{A}_a = \begin{bmatrix} A & 0 & 0 \\ -T_s C_1 & I_p & 0 \\ 0 & 0 & 0 \end{bmatrix}; \quad \overline{B}_a = \begin{bmatrix} B & 0 \\ 0 & 0 \\ 0 & I_q \end{bmatrix};$$

$$\overline{G}_a = \begin{bmatrix} 0 \\ T_s I_p \\ 0 \end{bmatrix}; \qquad \overline{X}_a(k) = \begin{bmatrix} x(k) \\ z(k) \\ f_a(k-1) \end{bmatrix}; \qquad \overline{U}(k) = \begin{bmatrix} u(k) \\ y(k+1) \end{bmatrix}.$$

The estimation of the fault magnitude $f_a$ can then be obtained using the SVD of matrix $\overline{E}_a$ if it is of full column rank [9].

Consider the SVD of matrix $\overline{E}_a$:

$$\overline{E}_a = T \begin{bmatrix} S \\ 0 \end{bmatrix} M^T, \qquad \text{with} \qquad T = \begin{bmatrix} T_1 & T_2 \end{bmatrix}.$$

$T$ and $M$ are orthonormal matrices such that: $TT^T = I$ , $MM^T = I$, and $S$ is a diagonal nonsingular matrix.

Substituting the SVD of $\overline{E}_a$ in (2.97) leads to

$$\begin{cases} \overline{X}_a(k+1) = \tilde{A}_a\overline{X}_a(k) + \tilde{B}_a\overline{U}(k) + \tilde{G}_a y_r(k) \\ 0 = \tilde{A}_0\overline{X}_a(k) + \tilde{B}_0\overline{U}(k) + \tilde{G}_0 y_r(k) \end{cases}, \tag{2.98}$$

where

$$\begin{aligned} \tilde{A}_a &= MS^{-1}T_1^T\ \overline{A}_a = \overline{E}_a^+\overline{A}_a; & \tilde{A}_0 &= T_2^T\overline{A}_a; \\ \tilde{B}_a &= MS^{-1}T_1^T\ \overline{B}_a = \overline{E}_a^+\overline{B}_a; & \tilde{B}_0 &= T_2^T\overline{B}_a; \\ \tilde{G}_a &= MS^{-1}T_1^T\ \overline{G}_a = \overline{E}_a^+\overline{G}_a; & \tilde{G}_0 &= T_2^T\overline{G}_a; \end{aligned} \tag{2.99}$$

and where $\overline{E}_a^+$ is the pseudo-inverse of matrix $\overline{E}_a$.

Therefore, the estimation $\hat{f}_a$ of the fault magnitude $f_a$ is the last component of the state vector $\overline{X}_a$, which is the solution of the first equation in (2.98). This solution $\overline{X}_a$ must satisfy the second equation of (2.98). It can be noted from (2.97) that the estimation of the fault magnitude $f_a$ at time instant $(k)$ depends on the system outputs $y$ at time instant $(k+1)$. To avoid this problem, the computation of the fault estimation is delayed by one sample.

**Estimation of Sensor Faults**

When a sensor fault affects the closed-loop system, the tracking error between the reference input and the measurement will no longer be equal to zero. In this case, the nominal control law tries to bring the steady-state error back to zero. Hence, in the presence of a sensor fault, the control law must be prevented from reacting, unlike the case of an actuator fault. This can be achieved by cancelling the fault effect on the control input.

For sensor faults, the output equation given in (2.25) is broken down according to (2.28), and can be written as

$$y(k) = Cx(k) + F_s f_s(k) = \begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} x(k) + \begin{bmatrix} F_{s1} \\ F_{s2} \end{bmatrix} f_s(k). \quad (2.100)$$

In this case, attention should be paid to the integral error vector $z$ which will be affected by the fault as well. The integral error vector can then be described as follows:

$$\begin{cases} z(k+1) = z(k) + T_s(y_r(k) - y_1(k)) \\ \qquad\quad = z(k) + T_s(y_r(k) - C_1 x(k) - F_{s1} f_s(k)) \end{cases}. \quad (2.101)$$

The sensor fault magnitude can be estimated in a similar way to that of the actuator fault estimation by describing the augmented system as follows:

$$\overline{E}_s \overline{X}_s(k+1) = \overline{A}_s \overline{X}_s(k) + \overline{B}_s \overline{U}(k) + \overline{G}_s y_r(k), \quad (2.102)$$

where

$$\overline{E}_s = \begin{bmatrix} I_n & 0 & 0 \\ 0 & I_p & 0 \\ C & 0 & F_s \end{bmatrix}; \quad \overline{A}_s = \begin{bmatrix} A & 0 & 0 \\ -T_s C_1 & I_p & -T_s F_{s1} \\ 0 & 0 & 0 \end{bmatrix}; \quad \overline{B}_s = \begin{bmatrix} B & 0 \\ 0 & 0 \\ 0 & I_q \end{bmatrix};$$

$$\overline{G}_s = \begin{bmatrix} 0 \\ T_s I_p \\ 0 \end{bmatrix}; \qquad \overline{X}_s(k) = \begin{bmatrix} x(k) \\ z(k) \\ f_s(k) \end{bmatrix}; \qquad \overline{U}(k) = \begin{bmatrix} u(k) \\ y(k+1) \end{bmatrix}.$$

The sensor fault magnitude $\hat{f}_s$ can then be estimated using the SVD of matrix $\overline{E}_s$ if this matrix is of full column rank.

## 2.7 Actuator and Sensor Fault-tolerance Principles

### 2.7.1 Compensation for Actuator Faults

The effect of the actuator fault on the closed-loop system is illustrated by substituting the feedback control law (2.33) in (2.23):

$$\begin{cases} x(k+1) = (A - BK_1)x(k) - BK_2 z(k) + F_a f_a(k) \\ y(k) = Cx(k) \end{cases}. \tag{2.103}$$

A new control law $u_{add}$ should be calculated and added to the nominal one in order to compensate for the fault effect on the system. Therefore, the total control law applied to the system is given by

$$u(k) = -K_1 x(k) - K_2 z(k) + u_{add}(k). \tag{2.104}$$

Considering this new control law given by (2.104), the closed-loop state equation becomes

$$x(k+1) = (A - BK_1)x(k) - BK_2 z(k) + F_a f_a(k) + Bu_{add}(k). \tag{2.105}$$

From this last equation, the additive control law $u_{add}$ must be computed such that the faulty system is as close as possible to the nominal one. In other words, $u_{add}$ must satisfy

$$Bu_{add}(k) + F_a f_a(k) = 0. \tag{2.106}$$

Using the estimation of the fault magnitude described in the previous section, the solution of (2.106) can be obtained by the following relation if matrix $B$ is of full row rank:

$$u_{add}(k) = -B^{-1} F_a \hat{f}_a(k). \tag{2.107}$$

The fault compensation principle presented under linear assumption can be directly extended to nonlinear affine systems but not to general ones. Indeed, according to (2.42) an additional control law can be applied to the decoupled linear subsystems. The three-tank system considered in Chap. 4 will provide an excellent example to illustrate this FTC design.

*Remark 2.1.* Matrix $B$ is of full row rank if the number of control inputs is equal to the number of state variables. In this case, $B$ is invertible.

### Case of Non Full Row Rank Matrix $B$

In the case when matrix $B$ is not of full row rank (*i.e.*, the number of system inputs is less than the number of system states), the designer chooses to maintain as many priority outputs as available control inputs to the detriment of other secondary outputs. To be as close as possible to the original system, these priority outputs are composed of the tracked outputs and of other remaining outputs. This is achieved at the control law design stage using, if necessary, a transformation matrix $P$ such that

$$\begin{cases} \begin{bmatrix} x_p(k+1) \\ x_s(k+1) \end{bmatrix} = \begin{bmatrix} A_{pp} & A_{ps} \\ A_{sp} & A_{ss} \end{bmatrix} \begin{bmatrix} x_p(k) \\ x_s(k) \end{bmatrix} + \begin{bmatrix} B_p \\ B_s \end{bmatrix} u(k) + \begin{bmatrix} F_{ap} \\ F_{as} \end{bmatrix} f_a(k) \\ y(k) = \begin{bmatrix} y_p(k) \\ y_s(k) \end{bmatrix} = C_T \begin{bmatrix} x_p(k) \\ x_s(k) \end{bmatrix} \end{cases}, \tag{2.108}$$

where index $p$ represents the priority variables and $s$ corresponds to the secondary variables. In this way, $B_p$ is a nonsingular square matrix. If the state-feedback gain matrix $K_1$ is broken down into $K_1 = \begin{bmatrix} K_p & K_s \end{bmatrix}$, the control law is then given by

$$u(k) = -\begin{bmatrix} K_p & K_s & K_2 \end{bmatrix} \begin{bmatrix} x_p(k) \\ x_s(k) \\ z(k) \end{bmatrix} + u_{add}(k). \tag{2.109}$$

Substituting (2.109) in (2.108) leads to

$$\begin{cases} x_p(k+1) = (A_{pp} - B_p K_p) x_p(k) - B_p K_2 z(k) \\ \qquad\qquad + (A_{ps} - B_p K_s) x_s(k) + F_{ap} f_a(k) + B_p u_{add}(k) \end{cases} \tag{2.110}$$

and

$$\begin{cases} x_s(k+1) = (A_{ss} - B_s K_s) x_s(k) - B_s K_2 z(k) \\ \qquad\qquad + (A_{sp} - B_s K_p) x_p(k) + F_{as} f_a(k) + B_s u_{add}(k) \end{cases}. \tag{2.111}$$

Here, the fault effect must be eliminated in the priority state variables $x_p$. Thus, from (2.110), this can be achieved by calculating the additive control law $u_{add}$ satisfying

$$(A_{ps} - B_p K_s) x_s(k) + F_{ap} f_a(k) + B_p u_{add}(k) = 0. \tag{2.112}$$

In this breakdown, if $x_s$ is not available for measurement, it can be computed from the output equation in (2.108), as $C_T$ is a full column rank matrix. Then, the solution $u_{add}$ of (2.112) is obtained using the fault estimation $\hat{f}_a$:

$$u_{add}(k) = -B_p^{-1}[(A_{ps} - B_p K_s) x_s(k) + F_{ap} \hat{f}_a(k)]. \tag{2.113}$$

The main goal is to eliminate the effect of the fault on the priority outputs. This is realized by choosing the transformation matrix $P$ such that

$$C_T = \begin{bmatrix} C_{T11} & 0 \\ C_{T21} & C_{T22} \end{bmatrix}.$$

Although that the secondary outputs are not compensated for, they must remain stable in the faulty case. Let us examine these secondary variables. Replacing (2.113) in (2.111) yields

$$x_s(k+1) = (A_{ss} - B_s B_p^{-1} A_{ps}) x_s(k) - B_s K_2 z(k) + (A_{sp} - B_s K_p) x_p(k) + (F_{as} - B_s B_p^{-1} F_{ap}) \hat{f}_a(k). \quad (2.114)$$

It is easy to see that the secondary variables are stable if and only if the eigenvalues of matrix $(A_{ss} - B_s B_p^{-1} A_{ps})$ belong to the unit circle.

### 2.7.2 Sensor Fault-tolerant Control Design

As for actuator faults, two main approaches have been proposed to eliminate the sensor fault effect which may occur on the system. One is based on the design of a software sensor where an estimated variable is used rather than the faulty measurement of this variable. The other method is based on adding a new control law to the nominal one.

#### Sensor Fault Masking

In the presence of sensor faults, the faulty measurements influence the closed-loop behavior and corrupt the state estimation. Sensor FTC can be obtained by computing a new control law using a fault-free estimation of the faulty element to prevent faults from developing into failures and to minimize the effects on the system performance and safety. From the control point of view, sensor FTC does not require any modification of the control law and is also called "sensor masking" as suggested in [131]. The only requirement is that the "estimator" provides an accurate estimation of the system output after a sensor fault occurs.

#### Compensation for Sensor Faults

The compensation for a sensor fault effect on the closed-loop system can be achieved by adding a new control law to the nominal one:

$$u(k) = -K_1 x(k) - K_2 z(k) + u_{add}(k). \quad (2.115)$$

It should be emphasized here that, in the presence of a sensor fault, both the output $y$ and the integral error $z$ are affected such that

$$\begin{cases} y(k) = Cx(k) = Cx_0(k) + F_s f_s(k) \\ z(k) = z_0(k) + \tilde{f}(k) \\ \tilde{f}(k) = \tilde{f}(k-1) - T_e F_{s1} f_s(k-1) \end{cases}, \quad (2.116)$$

where $x_0$ and $z_0$ are the fault-free values of $x$ and $z$ and $\tilde{f}$ is the integral of $-F_{s1} f_s$. Assuming that matrix $C = I$, these equations lead the control law to be written as follows:

$$u(k) = -K_1 x_0(k) - K_1 F_s f_s(k) - K_2 z_0(k) - K_2 \tilde{f}(k) + u_{add}(k). \quad (2.117)$$

The sensor fault effect on the control law and on the system can be cancelled by computing the additive control law $u_{add}$ such that

$$u_{add}(k) = K_1 F_s \hat{f}_s(k) + K_2 \tilde{f}(k). \tag{2.118}$$

*Remark 2.2.* In the case when matrix $C \neq I_n$, the control law can be calculated using the estimated state vector which is affected by the fault as well. The fault compensation will be achieved in a similar way to that given by (2.117) and (2.118).

It has been shown that the new control law added to the nominal one is not the same in the case of an actuator or sensor fault. Thus, the abilities of this FTC method to compensate for faults depend on the results given by the FDI module concerning the decision as to whether a sensor or an actuator fault has occurred.

### 2.7.3 Fault-tolerant Control Architecture

After having presented the different modules composing a general FTC architecture, the general concept of this approach is summarized in Fig. 2.9 in the linear framework, which is easily extended to the nonlinear case. The FDI module consists of residual generation, residual evaluation, and finally the decision as to which sensor or actuator is faulty. The fault estimation and compensation module starts the computation of the additive control law and is only able to reduce the fault effect on the system once the fault is detected and isolated. Obviously, the fault detection and isolation must be achieved as soon as possible to avoid huge losses in system performance or catastrophic consequences.

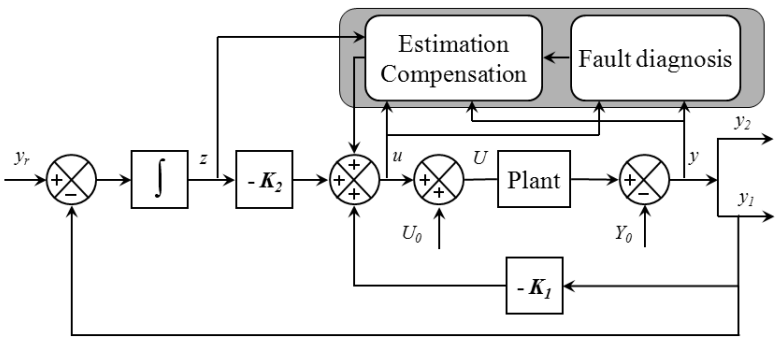


**Fig. 2.9.** FTC scheme

## 2.8 General Fault-tolerant Control Scheme

The general FTC method described here addresses actuator and sensor faults, which often affect highly automated systems. These faults correspond to a loss of actuator effectiveness or inaccurate sensor measurements.

The complete loss of a sensor can be overcome by using the compensation method presented previously, provided that the system is still observable. Actually, after the loss of a sensor, the observability property allows the estimation of the lost measurement using the other available measurements. However, the limits of this method are reached when there is a complete loss of an actuator; in this case, the controllability of the system should be checked. Very often, only a hardware duplication is effective to ensure performance reliability.

The possibility and the necessity of designing an FTC system in the presence of a major actuator failure such as a complete loss or a blocking of an actuator should be studied in a different way. For these kinds of failures, the use of multiple-model techniques is appropriate, since the number of failures is not too large. Some recent studies have used these techniques [104,137,139].

It is important to note that the strategy to implement and the level of achieved performance in the event of failures differ according to the type of process, the allocated degrees of freedom, and the severity of the failures . In this case, it is necessary to restructure the control objectives with a degraded performance. A complete active FTC scheme can be designed according to the previous classification illustrated in Fig. 1.1. This scheme is composed of the nominal control associated with the FDI module which aims to give information about the nature of the fault and its severity. According to this information, a reconfiguration or a restructuring strategy is activated. It is obvious that the success of the FTC system is strongly related to the reliability of the information issued from the FDI module. In the reconfiguration step, the fault magnitude is estimated. This estimation could be used as redundant information to that issued from the FDI module. The objective of this redundancy is to enhance the reliability of the diagnosis information. The complete FTC scheme discussed here is summarized in Fig. 2.10.

## 2.9 Conclusion

The FDI and the FTC problems are addressed in this chapter. The complete strategy to design an FTC system is presented. For this purpose, since many real systems are nonlinear, both nonlinear and linear techniques are shown. The linear techniques are used in case the system is linearized around an operating point.

The study presented here is based on the fault detection, the fault isolation, the fault estimation, and the compensation for the fault effect on the system. All these steps are taken into consideration. If this fault allows us to keep

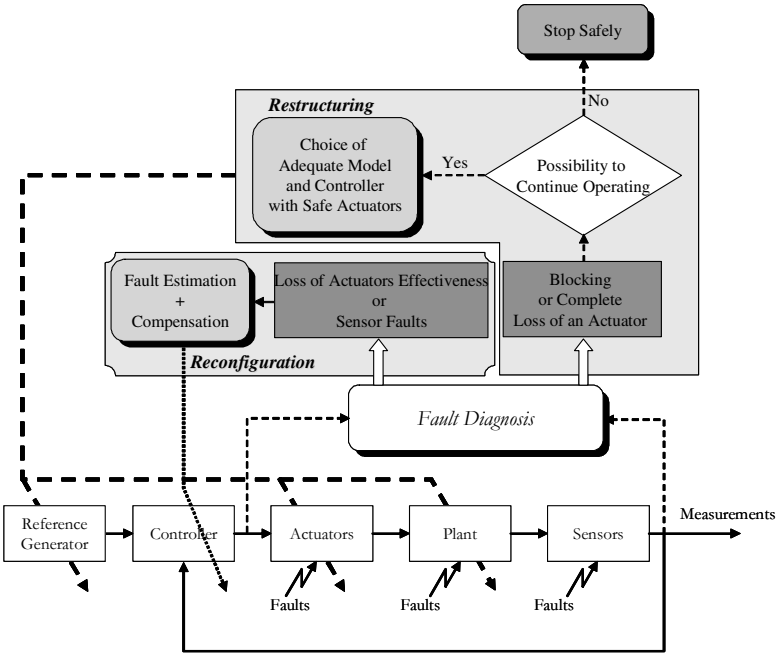


**Fig. 2.10.** General FTC scheme

using all the sensors and actuators, a method based on adding a new control law to the nominal one is described in order to compensate for the fault effect. For actuator faults, the objective of this new control law is to boost the control inputs in order to keep the performance of the faulty system close to the nominal system performance. Regarding sensor faults, the additive control law aims at preventing the total control inputs from reacting when these faults occur.

In case a major fault occurs on the system, such as the loss of an actuator, the consequences are more critical. This case is analyzed and the system should be restructured in order to use the healthy actuators and to redefine the objectives to reach. Therefore, the system will perform in degraded mode.

The following chapters are dedicated to the application of the linear and nonlinear methods described above to a laboratory-scale winding machine, a three-tank system, and finally in simulation of a full car active suspension system which is considered as a complex system. tured in order to use the healthy actuators and to redefine the objectives to reach. Therefore, the system will perform in degraded mode.

The following chapters are dedicated to the application of the linear and nonlinear methods described above to a laboratory-scale winding machine, a three-tank system, and finally in simulation to a full car active suspension system which is considered as a complex system.

http://www.springer.com/978-1-84882-652-6

Fault-tolerant Control Systems
Design and Practical Applications
Noura, H.; Theilliol, D.; Ponsart, J.-C.; Chamseddine, A.
2009, XXI, 233 p. With online files/update., Hardcover
ISBN: 978-1-84882-652-6