

Chapter 2

Send-on-Delta PI Control

Jesús Chacón, José Sánchez and Antonio Visioli

2.1 Introduction

The work described in this chapter is focused on *event-triggered* sampling, and in particular on the level crossing or *send-on-delta* (SOD) sampling [12]. Send-on-delta control and deadband control are similar schemes, and they are normally used as synonyms, since both consist in taking a new sample when a change greater than a predefined threshold δ is detected in the signal. However, in send-on-delta sampling, it is the signal itself and not the error function $\varepsilon(t)$ (defined as the difference between the last measurement taken and the current value of the signal) what is used to trigger an event. In practice, this scheme is equivalent to introduce a nonlinearity that can be characterized as a quantization of N_l levels with hysteresis.

The behavior of a control scheme based on level crossing sampling is studied considering two possible structures, the first one is when the sampler is located after the process output (Fig. 2.1a), and the second one after the controller output (Fig. 2.1b). Each case represents a configuration of a control scheme based on wireless transmissions, and has different properties. The first case corresponds to a plant with a wireless sensor which takes measures of the process variable and sends them to the controller, physically separated from the sensor but connected to the actuator. The second case is the opposite, where the controller is directly connected to the sensor and the actuator is in other place.

J. Chacón (✉) · J. Sánchez
Dpto. de Informática y Automática, Escuela Técnica Superior de Informática,
UNED, Madrid, Spain
e-mail: jchacon@bec.uned.es

J. Sánchez
e-mail: jsanchez@dia.uned.es

A. Visioli
Dipartimento di Ingegneria Meccanica e Industriale, Facoltà di Ingegneria,
Università degli Studi di Brescia, Brescia, Italy
e-mail: antonio.visioli@ing.unibs.it

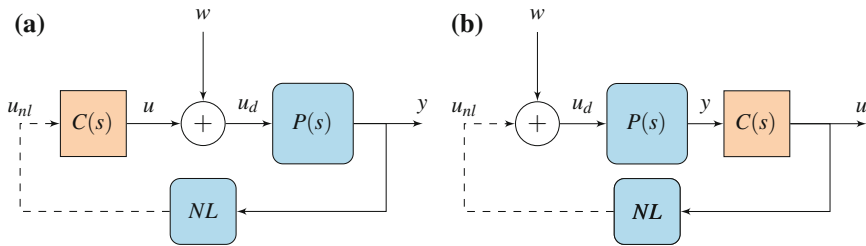


Fig. 2.1 Event-based control schemes. The block diagrams correspond to the two proposed configurations, **a** the event-based sampler at the process output and **b** at the controller output

The interest is to characterize with a systematic approach the behavior of the two event-based control structures with a set of processes such as integrator processes plus time delay process (IPTD), first-order processes plus time delay (FOPTD), and second-order processes plus time delay (SOPTD). The analysis is focused on the conditions for the existence of limit cycles, their period and amplitude, the effect of external disturbances, and the windup phenomena in the process due to the saturation of the actuators.

There are other works in the literature concerned to the study of limit cycles in event-based systems. In [216], authors analyze the existence, properties, and stability of limit cycles in relay systems. Related results can also be found in [33], where an event-based control scheme with a simple threshold detector is investigated, first in a double integrator and afterwards in the general case. Another approach can be found in [140], where the proposed structure is a model-based event-triggered PI in which the events are generated by the difference between the plant and the model implemented in the controller/sensor. The effect of actuator saturation is also investigated. In [81], a new method to perform the global stability analysis in piecewise linear systems (PLS) is proposed. The method is based on the use of quadratic Lyapunov functions in the switching surfaces. Finally, in [21, 22], authors analyze symmetric limit cycles in send-on-delta PI controllers, focusing on the stability of FOPTD processes and proposing tuning rules for this kind of controllers.

The main contributions of this chapter are the proposition of two general event-based schemes, and a new method for the analysis of the limit cycles that appear in the two presented schemes when they are applied to linear time-invariant (LTI) systems with time delay. The method has been applied to study a set of the most common industrial processes, obtaining results that have been confirmed in practice. In particular, a set of experiments carried out in the Distributed Collector Solar Field at the Solar Platform of Almería (PSA) showed the expected behavior.

2.2 The LTI and SOD Sampler Blocks

In the event-based control structure used in this work, three elements are present: the multilevel nonlinearity with hysteresis represented by the SOD sampler, the process (IPTD, FOPTD, or SOPTD), and the PI controller. In order to redefine the

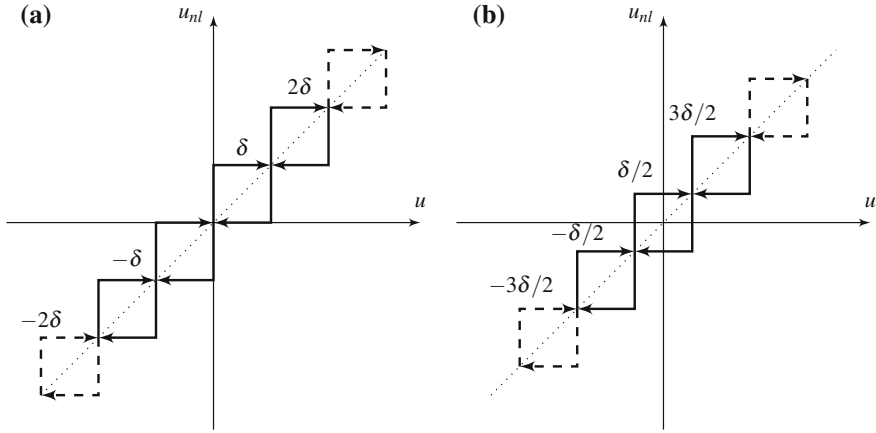


Fig. 2.2 Two cases of send-on-delta sampling with a different offset

event-based system as a PLS, first it is necessary to group the dynamics of the two linear elements in one block, that is, process and controller. Once this is done, the redefinition of the system as a PLS is simple since feedbacking the new linear block with the nonlinearity is equivalent to introduce a rule to switch between the N_l systems obtained by the combination of the dynamics of the linear block and the N_l output levels of the sampler.

Figure 2.2a shows the nonlinearity corresponding to the non-centered level crossing with saturation, and Fig. 2.2b represents a centered sampling with saturation. The dotted lines in both plots mean that the sampler has a saturation, but in general, the levels can be extended in both directions.

2.2.1 The Nonlinear Block: The SOD Sampler

The *event-triggered* control systems analyzed in this work use a level crossing sampling strategy, where, depending on the sampler location, the sensor sends information to the controller only when the sampled signal crosses certain predefined levels, or the controller sends the new values of the control action to the actuator when there is a significative change with respect to the previous value. The level crossing is considered the event that triggers the capture and the sending of a new sample.

Formally, a SOD sampler can be thought of as a block which has a continuous signal $u(t)$ as input and generates a sampled signal $u_{nl}(t)$ as output, which is a piecewise constant signal with $u_{nl}(t) = u(t_k)$, $\forall t \in [t_k, t_{k+1})$. Each t_k is denoted as *event time*, and it holds $t_{k+1} = \inf\{t \mid t > t_k \wedge |u(t) - u(t_k)| \geq \delta\}$, where δ is the sampling threshold, i.e., the minimum change that triggers the taking of a new sample. The previous condition holds for all t_k , except for t_0 , which is assumed to be the time instant when the block is initialized as $u_{nl}(t_0) = u(t_0)$.

In addition, this signal can be saturated due to the limitations in the sensors or in the actuators. As said before, the pattern resulting from sampling a signal with a SOD sampler can be described as a nonlinearity of N_l levels with hysteresis. It can be characterized as

$$u_{nl}(t^-) = (i + \alpha)\delta, \\ u_{nl}(t) = \begin{cases} (i + \alpha + 1)\delta & \text{if } u(t) \geq (i + \alpha + 1)\delta \\ (i + \alpha)\delta & \text{if } u(t) \in ((i + \alpha - 1)\delta, (i + \alpha + 1)\delta) \\ (i + \alpha - 1)\delta & \text{if } u(t) \leq (i + \alpha - 1)\delta \end{cases}, \quad (2.1)$$

where $\alpha \in [0, 1)$ is the offset with respect to the origin, $i \in \mathbb{Z}$ if the sampler is without saturation, and $i \in [i_{min}, i_{max}]$ when the sampler is with saturation.

Depending on the initial value, the nonlinearity introduced could have an offset with respect to the origin, $\alpha = u(t_0) - i\delta$, where $i = \lfloor u(t_0)/\delta \rfloor$. The level crossing sampling with offset is formally defined in the following paragraphs.

Definition 2.1 Let $T = \{t_0, \dots, t_{N_l}\}$, with $t_k \in \mathbb{R}$ and $t_{k-1} < t_k$, be a set of sampling times, and $u = \{u(t_0), \dots, u(t_{N_l})\}$ a set of samples. Thus, u is a level crossing sampling of $u(t)$ if, and only if, $|u(t_k) - u(t_{k-1})| = \delta$, $k = 0, 1, \dots, N_l$.

Note that every sample can be expressed as $y_s(t_k) = (i_k + \alpha)\delta$, where $i_k \in \mathbb{Z}$ is the crossed level by the input signal $y(t)$, and $\alpha \in [0, 1) \subset \mathbb{R}$ is the sampling offset which depends on the initial sample, $y(t_0)$.

Definition 2.2 The order of a nonlinearity with hysteresis is defined by subtracting 1 to the number of crossing levels.

For example, a two-level nonlinearity with hysteresis and zero offset owns three crossing levels, that is, $-\delta$, 0 , and δ . It must be noticed that if the nonlinearity had zero offset, the number of levels would be always even, and odd in the opposite case.

2.2.2 The Linear Blocks: The Process and the Controller

As Fig. 2.3 shows, the linear dynamics of the process and the PI controller can be joined in two different ways by placing the nonlinear block at the process output (Fig. 2.3a) or at the controller output (Fig. 2.3b). Depending on the combination, two different linear blocks G_{ps} and G_{cs} are obtained and, once the loop is closed with the sampler, two PLS are produced with different limit cycles features. The dynamics of the G_{ps} and G_{cs} blocks will be represented by the augmented state matrices obtained by combining process and controller.

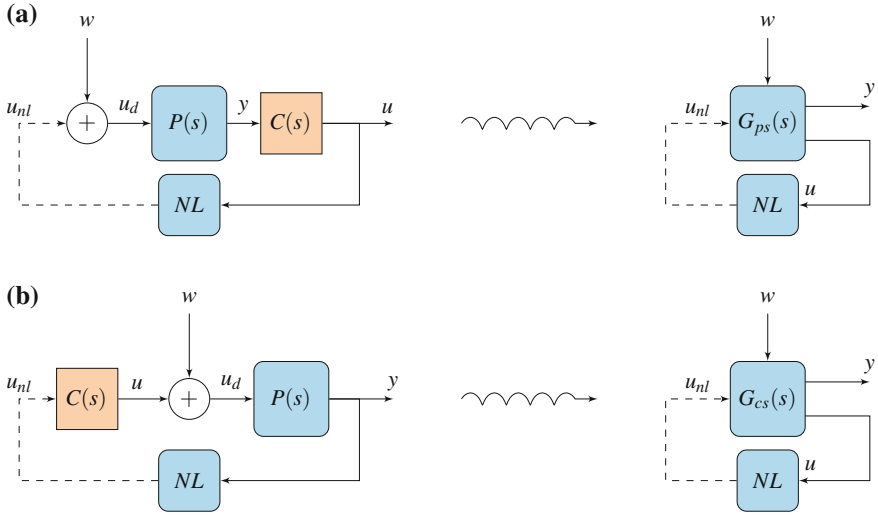


Fig. 2.3 Event-based control schemes. The block diagrams on the *left* correspond to the two proposed configurations, **a** the event-based sampler at the process output and **b** at the controller output. On the *right*, the continuous dynamics have been grouped in one block to simplify the analysis. *Solid lines* represent a continuous data flow, while *dotted lines* mean a discontinuous data flow

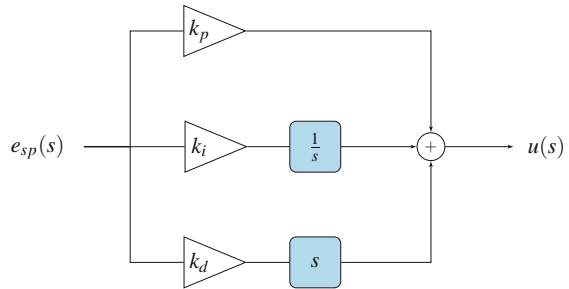
2.2.2.1 State-Space Representation of the Controller

Although there are different implementations of the PID algorithm in literature, all of them are essentially equivalent. In this work, the parallel form of the PID algorithm is considered (see Fig. 2.4). It can be represented by the transfer function:

$$u(s) = \left(k_p + \frac{k_i}{s} + k_d s \right) e_{sp}(s), \quad (2.2)$$

where $e_{sp} = y_{sp} - y$ is the control error.

Fig. 2.4 Block diagram of the PID controller in the parallel form



The PID transfer function has a high gain for high frequencies, due to the derivative term. To avoid problems with noisy signals, in practical implementations, it is common to filter the derivative with a first-order filter. With this consideration, the resulting transfer function is

$$u(s) = \left(k_p + \frac{k_i}{s} + \frac{k_d s}{1 + \frac{k_d}{k_N} s} \right) e_{sp}(s). \quad (2.3)$$

Let us start obtaining the matrices corresponding to the continuous case, that is, without considering the SOD sampler. No delays are considered now. Assume that the process $P(s)$ and the controller $C(s)$ are described by the time-invariant state-space systems:

$$\begin{aligned} P(s) &\sim \begin{cases} \dot{x}_p(t) = A_p x_p(t) + B_p(u(t) + w(t)) \\ y(t) = C_p x_p(t) \end{cases} \\ C(s) &\sim \begin{cases} \dot{x}_c(t) = A_c x_c(t) + B_c e_{sp}(t) \\ u(t) = C_c x_c(t) + D_c e_{sp}(t), \end{cases} \end{aligned} \quad (2.4)$$

where $x_p \in \mathbb{R}^n$ is the state of the process, $x_c \in \mathbb{R}^q$ is the state of the controller, and A_p is non-singular.

Combining the two parts of (2.4), the whole system is

$$\begin{aligned} \begin{pmatrix} \dot{x}_p(t) \\ \dot{x}_c(t) \end{pmatrix} &= \begin{pmatrix} A_p - B_p D_c C_p & B_p C_c \\ -B_c C_p & A_c \end{pmatrix} \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} B_p D_c & B_p \\ B_c & 0 \end{pmatrix} \begin{pmatrix} y_{sp} \\ w \end{pmatrix} \\ \begin{pmatrix} y(t) \\ u(t) \end{pmatrix} &= \begin{pmatrix} C_p & 0 \\ -D_c C_p & C_c \end{pmatrix} \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ D_c & 0 \end{pmatrix} \begin{pmatrix} y_{sp} \\ w \end{pmatrix}. \end{aligned} \quad (2.5)$$

2.2.2.2 Sampling the Process Variable: The G_{ps} Block

The introduction of the SOD sampler in the PID control loop modifies the expressions presented in the previous section. Depending on where the sampler is placed, either the controller or the process input only changes at certain times.

To include the effect of the SOD sampler in (2.4), a new variable is introduced: u_{nl} , the nonlinear output of the sampler. At the sampling instants t_k , $u_{nl} := y(t_k)$ if the sampler is at the process variable, and $u_{nl} := u(t_k)$ in the opposite case. The expressions corresponding to the control loop with the SOD sampler are obtained introducing u_{nl} in (2.4), which yields

$$\begin{pmatrix} \dot{x}_p \\ \dot{x}_c \end{pmatrix} = \begin{pmatrix} A_p & B_p C_c \\ 0 & A_c \end{pmatrix} \begin{pmatrix} x_p \\ x_c \end{pmatrix} + \begin{pmatrix} -B_p D_c & B_p D_c & B_p \\ -B_c & B_c & 0 \end{pmatrix} \begin{pmatrix} u_{nl} \\ y_{sp} \\ w \end{pmatrix}$$

$$\begin{pmatrix} y \\ u \end{pmatrix} = \begin{pmatrix} C_p & 0 \\ 0 & C_c \end{pmatrix} \begin{pmatrix} x_p \\ x_c \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ -D_c & D_c & 0 \end{pmatrix} \begin{pmatrix} u_{nl} \\ y_{sp} \\ w \end{pmatrix}. \quad (2.6)$$

2.2.2.3 Sampling the Control Variable: The G_{cs} Block

The procedure to obtain the augmented state matrices of the G_{cs} block is similar to the previous case but taking into account that now the input to the controller $u(t)$ is the process output $y(t)$, and the input to the process is the signal resulting of adding the disturbance $w(t)$ to an input named $u_s(t)$. The resulting system $G_{cs}(s)$ is

$$\begin{aligned} \begin{pmatrix} \dot{x}_p \\ \dot{x}_c \end{pmatrix} &= \begin{pmatrix} A_p & 0 \\ -B_c C_p & A_c \end{pmatrix} \begin{pmatrix} x_p \\ x_c \end{pmatrix} + \begin{pmatrix} B_p & 0 & B_p \\ 0 & B_c & 0 \end{pmatrix} \begin{pmatrix} u_{nl} \\ y_{sp} \\ w \end{pmatrix} \\ \begin{pmatrix} y \\ u \end{pmatrix} &= \begin{pmatrix} C_p & 0 \\ -D_c C_p & C_c \end{pmatrix} \begin{pmatrix} x_p \\ x_c \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & D_c & 0 \end{pmatrix} \begin{pmatrix} u_{nl} \\ y_{sp} \\ w \end{pmatrix}. \end{aligned} \quad (2.7)$$

To simplify the analysis of the previous expressions, from now on, the set-point is assumed to be null, i.e., $y_{sp} = 0$. This is done without loss of generality, as shown in the following Proposition.

Proposition 2.1 *Consider the systems*

$$G_{ps}(s) \sim \begin{cases} \dot{x}_{ps} = A_{ps}x_{ps} + B_{ps}u_{ps} \\ y_{ps} = C_{ps}x_{ps} + D_{ps}u_{ps}, \end{cases} \quad (2.8)$$

where

$$\begin{aligned} A_{ps} &= \begin{pmatrix} A_p & B_p C_c \\ 0 & A_c \end{pmatrix} & B_{ps} &= \begin{pmatrix} -B_p D_c & B_p \\ -B_c & 0 \end{pmatrix} \\ C_{ps} &= \begin{pmatrix} C_p & 0 \\ 0 & C_c \end{pmatrix} & D_{ps} &= \begin{pmatrix} 0 & 0 \\ -D_c & 0 \end{pmatrix}, \end{aligned} \quad (2.9)$$

and

$$G_{cs}(s) \sim \begin{cases} \dot{x}_{cs} = A_{cs}x_{cs} + B_{cs}u_{cs} \\ y_{cs} = C_{cs}x_{cs}, \end{cases} \quad (2.10)$$

where

$$\begin{aligned} A_{cs} &= \begin{pmatrix} A_p & 0 \\ -B_c C_p & A_c \end{pmatrix} & B_{cs} &= \begin{pmatrix} B_p & B_p \\ 0 & 0 \end{pmatrix} \\ C_{cs} &= \begin{pmatrix} C_p & 0 \\ -D_c C_p & C_c \end{pmatrix}. \end{aligned}$$

These systems are equivalent to (2.6) and (2.7). Therefore, the set-point can be assumed to be null without loss of generality.

Proof Consider that the sampler is placed at the process output. Note that defining $(\bar{x}_p, \bar{x}_c, \bar{u}, \bar{y}, \bar{u}_{nl}, \bar{w}) := (x_p, x_c, u, y, u_{nl} - y_{sp}, w)$, the system can be equivalently written as (2.6). Now, consider that the sampler is placed at the controller output. Defining $(\tilde{x}_p, \tilde{x}_c, \tilde{u}, \tilde{y}, \tilde{u}_{nl}, \tilde{w}) := (x_p - \frac{C_p^T}{\|C_p\|^2} y_{sp}, x_c, u, y - y_{sp}, u_{nl} - \frac{B_p^T A_p C_p^T}{\|B_p\|^2 \|C_p\|^2} y_{sp}, w)$, the system can be written equivalently as (2.7). In both cases, the introduced variables only differ from the original by a constant value. Therefore, it can be assumed without loss of generality that $y_{sp} = 0$. To make the notation more clear, from now on, the variables are written without the symbols \sim and $\bar{\cdot}$.

2.2.3 The P, I, PI, PD, and PID Controllers

In this section, the expressions (2.6) and (2.7) are particularized to the most frequently forms of the PID controller. Although the PI is considered in most of the examples, because it is the most extended controller, the propositions and algorithms presented in this chapter are in general form, so they can be applied to any of the enumerated cases by choosing the appropriate matrices.

2.2.3.1 Proportional Controller (P)

The proportional or P controller is a controller with a feedback based only in the current error of the process. Its associated state-space matrices are $A_c = B_c = C_c = 0$, and $D_c = k_p$. Thus,

$$\begin{aligned} A_{ps} &= \begin{pmatrix} A_p & 0 \\ 0 & 0 \end{pmatrix} & B_{ps} &= \begin{pmatrix} -k_p B_p & B_p \\ 0 & 0 \end{pmatrix} \\ C_{ps} &= \begin{pmatrix} C_p & 0 \\ 0 & 0 \end{pmatrix} & D_{ps} &= \begin{pmatrix} 0 & 0 \\ -k_p & 0 \end{pmatrix}, \end{aligned} \quad (2.11)$$

and

$$\begin{aligned} A_{cs} &= \begin{pmatrix} A_p & 0 \\ 0 & 0 \end{pmatrix} & B_{cs} &= \begin{pmatrix} B_p & B_p \\ 0 & 0 \end{pmatrix} \\ C_{cs} &= \begin{pmatrix} C_p & 0 \\ -k_p C_p & 0 \end{pmatrix}. \end{aligned} \quad (2.12)$$

2.2.3.2 Integral Controller (I)

The transfer function of the integrator or I controller is $C(s) = \frac{k_i}{s} e_{sp}(s)$, therefore one of its possible representations in the state space is given by $A_c = 0$, $B_c = 1$, $C_c = k_i$, and $D_c = 0$. Thus,

$$\begin{aligned} A_{ps} &= \begin{pmatrix} A_p & k_i B_p \\ 0 & 0 \end{pmatrix} & B_{ps} &= \begin{pmatrix} 0 & B_p \\ -1 & 0 \end{pmatrix} \\ C_{ps} &= \begin{pmatrix} C_p & 0 \\ 0 & k_i \end{pmatrix}, \end{aligned} \quad (2.13)$$

and

$$\begin{aligned} A_{cs} &= \begin{pmatrix} A_p & 0 \\ -C_p & 0 \end{pmatrix} & B_{cs} &= \begin{pmatrix} B_p & B_p \\ 0 & 0 \end{pmatrix} \\ C_{cs} &= \begin{pmatrix} C_p & 0 \\ 0 & k_i \end{pmatrix}. \end{aligned} \quad (2.14)$$

2.2.3.3 Proportional-Integral Controller (PI)

The PI controller state-space matrices are $A_c = 0$, $B_c = 1$, $C_c = k_i$, and $D_c = k_p$. Thus,

$$\begin{aligned} A_{ps} &= \begin{pmatrix} A_p & k_i B_p \\ 0 & 0 \end{pmatrix} & B_{ps} &= \begin{pmatrix} -k_p B_p & B_p \\ -1 & 0 \end{pmatrix} \\ C_{ps} &= \begin{pmatrix} C_p & 0 \\ 0 & k_i \end{pmatrix} & D_{ps} &= \begin{pmatrix} 0 & 0 \\ -k_p & 0 \end{pmatrix}, \end{aligned} \quad (2.15)$$

and, when the sampler is at the control variable,

$$\begin{aligned} A_{cs} &= \begin{pmatrix} A_p & 0 \\ -C_p & 0 \end{pmatrix} & B_{cs} &= \begin{pmatrix} B_p & B_p \\ 0 & 0 \end{pmatrix} \\ C_{cs} &= \begin{pmatrix} C_p & 0 \\ -k_p C_p & k_i \end{pmatrix}. \end{aligned} \quad (2.16)$$

2.2.3.4 Proportional-Derivative Controller (PD)

The PD controller is $A_c = -N \frac{k_p}{k_d}$, $B_c = N \frac{k_p}{k_d}$, $C_c = k_d$, and $D_c = (1 + N)k_p$. Thus,

$$\begin{aligned} A_{ps} &= \begin{pmatrix} A_p & k_d B_p \\ 0 & -N \frac{k_p}{k_d} \end{pmatrix} & B_{ps} &= \begin{pmatrix} -(1 + N)k_p B_p & B_p \\ -N \frac{k_p}{k_d} & 0 \end{pmatrix} \\ C_{ps} &= \begin{pmatrix} C_p & 0 \\ 0 & k_d \end{pmatrix} & D_{ps} &= \begin{pmatrix} 0 & 0 \\ -(1 + N)k_p & 0 \end{pmatrix}, \end{aligned} \quad (2.17)$$

and, when the sampler is at the control variable, the expressions are

$$\begin{aligned} A_{cs} &= \begin{pmatrix} A_p & 0 \\ -N \frac{k_p}{k_d} C_p & A_c \end{pmatrix} & B_{cs} &= \begin{pmatrix} B_p & B_p \\ 0 & 0 \end{pmatrix} \\ C_{cs} &= \begin{pmatrix} C_p & 0 \\ -(1+N)k_p C_p & k_d \end{pmatrix}. \end{aligned} \quad (2.18)$$

2.2.3.5 PID with Derivative Filter

The PID controller with derivative filter has $A_c = \begin{pmatrix} 0 & 0 \\ 0 & -N \frac{k_p}{k_d} \end{pmatrix}$, $B_c = \begin{pmatrix} 1 \\ N \frac{k_p}{k_d} \end{pmatrix}$, $C_c = (k_i \ k_d)$, and $D_c = (1+N)k_p$. Thus,

$$\begin{aligned} A_{ps} &= \begin{pmatrix} A_p & B_p k_i & B_p k_d \\ 0 & 0 & 0 \\ 0 & 0 & -N \frac{k_p}{k_d} \end{pmatrix} & B_{ps} &= \begin{pmatrix} -B_p(1+N)k_p & B_p \\ -1 & 0 \\ -N \frac{k_p}{k_d} & 0 \end{pmatrix} \\ C_{ps} &= \begin{pmatrix} C_p & 0 & 0 \\ 0 & k_i & k_d \end{pmatrix} & D_{ps} &= \begin{pmatrix} 0 & 0 \\ -(1+N)k_p & 0 \end{pmatrix}, \end{aligned} \quad (2.19)$$

where $A_p \in \mathbb{R}^{n \times n}$, $A_{ps} \in \mathbb{R}^{n_s \times n_s}$, $B_{ps} \in \mathbb{R}^{n_s \times 2}$, $C_{ps} \in \mathbb{R}^{2 \times n_s}$, and $B_{ps} \in \mathbb{R}^{2 \times 2}$, with $n_s = n + 2$.

When the sampler is at the control variable, the expressions are

$$\begin{aligned} A_{cs} &= \begin{pmatrix} A_p & 0 & 0 \\ -C_p & 0 & 0 \\ -N \frac{k_p}{k_d} C_p & 0 & -N \frac{k_p}{k_d} \end{pmatrix} & B_{cs} &= \begin{pmatrix} B_p & B_p \\ 0 & 0 \end{pmatrix} \\ C_{cs} &= \begin{pmatrix} C_p & 0 & 0 \\ -(1+N)k_p C_p & k_i & k_d \end{pmatrix}. \end{aligned} \quad (2.20)$$

where $A_{cs} \in \mathbb{R}^{n_s \times n_s}$, $B_{cs} \in \mathbb{R}^{n_s \times 2}$, and $C_{cs} \in \mathbb{R}^{2 \times n_s}$.

2.3 Defining an Event-Based System as a PLS

The results obtained in this section do not depend on where the sampler is placed. Therefore, the subindexes representing the position of the sampler, ps and cs , have been dropped to simplify the notation. For example, A is equivalent to A_{ps} , if the process variable is being sample, or to A_{cs} in the other case.

Let us consider that the input to $G(s)$ is delayed in time by τ and the loop is closed by adding the nonlinearity represented by the SOD sampler. Then,

$$G(s) \sim \begin{cases} \dot{x}(t) = Ax(t) + Bu(t - \tau) \\ y(t) = Cx(t) + Du(t - \tau). \end{cases} \quad (2.21)$$

Now, the resulting system is an infinite-dimensional system because of the time delay. However, it can be simplified because closing the loop with the nonlinearity makes the input signal piecewise constant. So, in the matrix u , the input $u_{nl}(t)$ is redefined as

$$u_{nl_k} = (j_k + \alpha)\delta,$$

where $j_k \in \mathbb{Z}$ is the crossed level by the input signal $u_{nl}(t)$, and $\alpha \in [0, 1) \in \mathbb{R}$ is the sampling offset which depends on the initial sample, $u_{nl}(t_0)$. Let us define the switching times as follows:

Definition 2.3 Consider a limit cycle composed of N_l switchings, and assume $T = \{t_0, \dots, t_{N_l}\}$ are the sampling times, as in Definition 2.1. Then the switching times are defined as $t_i^* = t_i - t_{i-1}$ (see Fig. 2.5).

And the order of the limit cycle is defined as follows:

Definition 2.4 Consider a limit cycle in a symmetric nonlinearity of order N_l ($N_l + 1$ crossing levels, as in Definition 2.2). Then, the number of switchings of the limit cycle is $2N_l$.

Then, if limit cycles of period T with switching times, $t_i^* > \tau$ where $T = t_1^* + t_2^* + \dots + t_{2N_l}^*$, are only considered, the input $\{u_{nl}(t), t_i^* - \tau < t \leq t_i^*\}$ is

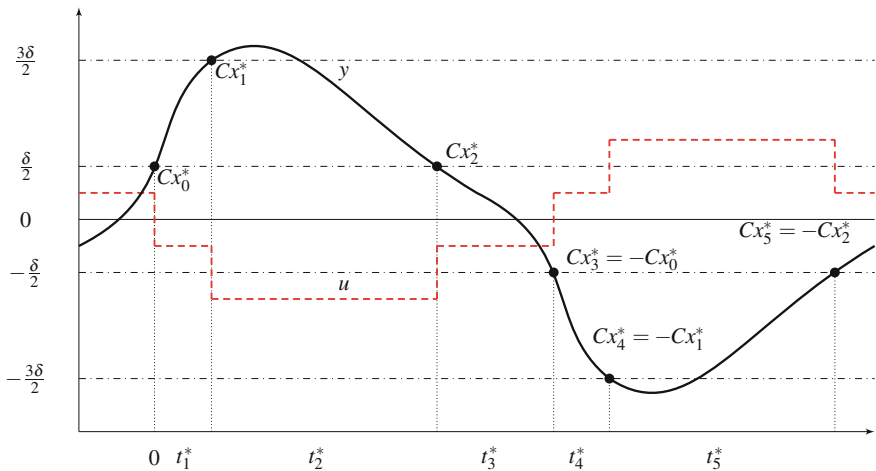


Fig. 2.5 Trajectory of a solution of a PLS system corresponding to a limit cycle in a system with a symmetric send-on-delta sampling scheme, with sampling threshold δ , and sampling offset $\alpha = 0.5$. The solid line represents the process output, and the dashed line the control input. The process output y crosses the sampling levels at times t_i , generating new updates. Each time interval is denoted by $t_i^* = t_i - t_{i-1}$. In this particular case, the control input u is constant between consecutive crossings, $|Cx_{i+1}^* - Cx_i^*| = \delta$, and each x_i^* is the system state at t_i

constant and its value is given by the feedback. In this case, the state of the system at the sampling times $t_1, t_2, \dots, t_{2N_l}$ is uniquely given by $x(t_i)$. Taking into account this consideration, the system can be considered as a PLS defined by

$$\begin{aligned}\dot{x}(t) &= Ax(t) + B_i \\ y(t) &= Cx(t) + D_i,\end{aligned}\tag{2.22}$$

where $B_i = B(u_{nl_i} w)^T$, $D_i = D(u_{nl_i} w)^T$, and $u_{nl_i} = (j_i + \alpha)\delta$, for $i = \{0, \pm 1, \dots, \pm N_l\}$.

In this PLS, the rule to switch between the linear systems is included in the definitions of the nonlinearities. It must be noted that the switching rules have memory and the decision of which LTI to use may not depend only on the actual values of the state, but also on their past values. In the state-space system, the points in which a rule provokes the switching from the system i to the system j define a surface which is known as *switching surface* (see Fig. 2.6). These surfaces consist of hyperplanes of dimension $n - 1$, being n the order of the system, i.e., $x \in \mathbb{R}^n$,

$$S_i = \{x \mid Cx - u_{nl_i} = 0\} \text{ for } i = \{0, \pm 1, \dots, \pm N_l\}.$$

It is interesting to characterize the limit cycles that can appear in the system due to the effect of the nonlinearities introduced with the event-based sampling scheme.

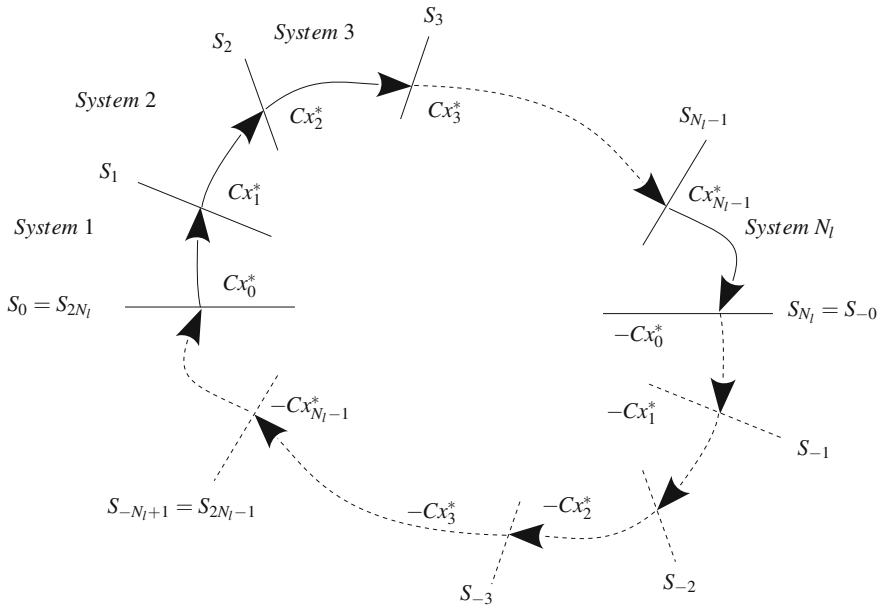


Fig. 2.6 Trajectory of a solution of a PLS system. In each region, the dynamics of the system is determined by a different LTI system defined by (2.22). The lines represent the switching surfaces

The study of these limit cycles can be interesting for several reasons. There is a wide range of processes that will almost surely present limit cycles with the studied control schemes, while for other processes, they can be prevented by carefully choosing the controller parameters. In any case, limit cycles mean oscillations, which, depending on the process, may be more or less problematic. For example, a high frequency of oscillation may wear out the actuators. Also, the study of limit cycles can be used for identification purposes. For example, the relay autotuning method [1] is based on the properties of the limit cycle that appears in a process subject to relay feedback (which can be considered as a particular case of the studied event-based scheme). In addition, for the cases when the limit cycles cannot be prevented, it may be important to know about the stability of these limit cycles.

In order to calculate the period and amplitude, first it is assumed that the limit cycle contains $N_l + 1$ levels, and thus it is composed of $2N_l$ switchings, where the first N_l correspond to the positive level crossings, and the rest N_l are the negative ones. Next, it is presented the set of equations that allows us to find the switching times and the values of the states at these switching times. The result stated in the following proposition is a generalization to N_l levels of the approaches in [33, 81, 216].

Proposition 2.2 *Consider the PLS in (2.22), with a nonlinearity defined by the switching surfaces $S_i = \{x \mid Cx - (j_i + \alpha)\delta = 0\}$, where $\alpha \in [0, 1)$, $j_i \in \mathbb{Z}$, $i \in \{0, \pm 1, \pm N_l\}$, and $0 < \delta \in \mathbb{R}$. Assume that there exists a symmetric periodic solution γ with $2n$ switching surfaces per period $T = t_1^* + t_2^* + \dots + t_{2N_l}^*$, where $t_1^*, t_2^*, \dots, t_{2N_l}^*$ are the switching times when the switching surfaces $S_1, \dots, S_{N_l}, S_{-1}, \dots, S_{-N_l}$ are crossed, respectively (Fig. 2.6). Define*

$$f_k(t_1^*, \dots, t_{2N_l}^*) = C(I + e^{AT})^{-1} \left[\sum_{i=1}^{2N_l-1} \Phi_{2N_l-1} \dots \Phi_{i+1} (\Phi_i - I) \Lambda_i \right] - E_k, \quad (2.23)$$

where $\Phi_i = \Phi(t_i) = e^{At_i}$, and $\Lambda_i = A^{-1}B_i$. Then the following conditions hold

$$\begin{cases} f_1(t_1^*, t_2^*, \dots, t_{2N_l}^*) = 0 \\ f_2(t_1^*, t_2^*, \dots, t_{2N_l}^*) = 0 \\ \vdots \\ f_{2N_l}(t_1^*, t_2^*, \dots, t_{2N_l}^*) = 0, \end{cases} \quad (2.24)$$

and

$$\begin{cases} E_i \leq y(t) = Cx_i(t) < E_{i+1} \text{ for } 0 \leq t < t_i^* & i = 1 \dots N_l - 1 \\ y(t) = Cx_{N_l}(t) \geq E_{N_l+1} & \text{for } 0 \leq t < t_{N_l}^* \\ E_i \geq y(t) = Cx_i(t) > E_{i+1} \text{ for } 0 \leq t < t_i^* & i = N_l + 1 \dots 2N_l, \end{cases} \quad (2.25)$$

where

$$E_i = (j_i + \alpha)\delta, \text{ and } x_i(t) = e^{At}x_{i-1}^* - A^{-1}(e^{At} - I)B_i.$$

Furthermore, the limit cycle can be obtained with the initial condition

$$x_0^* = (I + e^{AT})^{-1} \left[\sum_{i=1}^{2N_l-1} \Phi_{2N_l-1} \cdots \Phi_{i+1} (\Phi_i - I) \Lambda_i \right]. \quad (2.26)$$

Proof Assuming that $t_i > \tau$, where t_i is the time elapsed between the crossing of two consecutive switching surfaces, for example i and $i + 1$, then the state is obtained by integrating (2.22) from $t = 0$ to $t = t_i$. It gives

$$x_{i+1} = \Phi(t_i)x_i + \Gamma_1(t_i)u_{i-1} + \Gamma_0(t_i)u_i, \quad (2.27)$$

where $\Phi(t) = e^{At}$ is the state transition matrix, $\Gamma_0(t) = \int_0^{t-\tau} \Phi(s)ds$ accounts for the effect of the input, and $\Gamma_1(t) = \int_{t-\tau}^t \Phi(s)ds$ is a term which represents the effect of the input because of the time delay of the system.

Then, for a limit cycle involving N_l switching times, we have a system of equations described by

$$\begin{aligned} \Phi(t_1)x_1 + \Gamma_1(t_1)u_{N_l} + \Gamma_0(t_1)u_1 - x_2 &= 0 \\ &\vdots \\ \Phi(t_{N_l-1})x_{N_l-1} + \Gamma_1(t_{N_l-1})u_{N_l-2} + \Gamma_0(t_{N_l-1})u_{N_l-1} - x_{N_l} &= 0 \\ \Phi(t_{N_l})x_{N_l} + \Gamma_1(t_{N_l})u_{N_l-1} + \Gamma_0(t_{N_l})u_{N_l} - x_1 &= 0. \end{aligned} \quad (2.28)$$

Substituting recursively, we get the following expression:

$$\begin{aligned} [I - \Phi(t_{N_l}) \cdots \Phi(t_1)]x_n &= \Phi(t_{N_l}) \cdots \Phi(t_2)(\Gamma_1(t_1)u_{N_l} + \Gamma_0(t_1)u_1) \\ &\quad + \Phi(t_{N_l}) \cdots \Phi(t_3)(\Gamma_1(t_2)u_1 + \Gamma_0(t_2)u_2) + \cdots \\ &\quad + \Phi(t_{N_l})(\Gamma_1(t_{N_l-1})u_{N_l-2} + \Gamma_0(t_{N_l-1})u_{N_l-1}) \\ &\quad + \Gamma_1(t_{N_l})u_{N_l-1} + \Gamma_0(t_{N_l})u_{N_l}. \end{aligned} \quad (2.29)$$

The previous expression can be written in compact form as

$$[I - \Phi_{N_l} \cdots \Phi_1]x_{N_l} = \sum_{i=1}^{2N_l-1} \Phi_{2N_l-1} \cdots \Phi_{i+1} [\Gamma_1(t_i)u_{i-1} + \Gamma_0(t_i)u_i], \quad (2.30)$$

and

$$x_{N_l} = [I - \Phi_{N_l} \cdots \Phi_1]^{-1} \sum_{i=1}^{2N_l-1} \left(\prod_{j=1}^{2N_l-1-i} \Phi_{2N_l-j} \right) [\Gamma_1(t_i)u_{i-1} + \Gamma_0(t_i)u_i]. \quad (2.31)$$

If we assume that the system matrix A is non-singular, then the integrals Γ_1 and Γ_0 can be explicitly computed and the state x_{N_l} can be solved, yielding the expression

of the initial state (2.26). Since we know that at the switching times the state must be at the switching surface, we can combine the previous expression with the switching conditions to get the set of equations given by (2.24). Finally, the conditions (2.25) hold because the state does not cross the switching surface in the interval between two switchings.

However, if the system matrix is assumed to be singular, neither the state nor the integrals Γ_0 and Γ_1 can be computed explicitly. In this case, the system of equations is obtained in the same way, but the computations are more involved. First the functions in (2.24) are redefined as

$$f_i(x_i^*, t_1^*, \dots, t_{2N_l}^*) = [I - \Phi_{2N_l} \dots \Phi_1]x_i^* - \sum_{i=1}^{2N_l-1} \Phi_{2N_l-1} \dots \Phi_{i+1}[\Gamma_1(t_i^*)u_{i-1} + \Gamma_0(t_i^*)u_i], \quad (2.32)$$

where, in contrast to the previous case, x_i^* appear as unknowns.

Note that the system of equations given by the functions f_i is composed of N_l^2 scalar equations and $N_l^2 + N_l$ unknowns. Thus, in order to solve it, the system must be completed with N_l additional equations. These equations are obtained from the switching conditions in the sampler, which fix the values of either the process output or the control input at the event times, depending on where the sampling is placed. Then, the complete set of equations that must be solved to obtain the features of the limit cycle is

$$\left\{ \begin{array}{l} f_1(x_1^*, t_1^*, \dots, t_{2N_l}^*) = 0 \\ \vdots \\ f_{2N_l}(x_{2N_l}^*, t_1^*, \dots, t_{2N_l}^*) = 0 \\ Cx_1^* - d_1 = 0 \\ \vdots \\ Cx_{2N_l}^* - d_{2N_l} = 0. \end{array} \right. \quad (2.33)$$

How to use this result to analyze the limit cycles is demonstrated with examples in Sects. 2.4 and 2.5.

2.3.1 Local Stability

The local stability of the limit cycles described in the previous paragraphs can be analyzed by observing the system at the switching times. The following result, which is a generalization of the approaches in [33, 81, 216], can be applied to study the behavior of the trajectories of the system in the proximities of the limit cycles.

Proposition 2.3 *Assume that there exists a limit cycle γ with k states in the system (2.22). The limit cycle is locally stable if and only if $W = W_k W_{k-1} \dots W_1$ has all*

its eigenvalues inside the unit circle, where $W_i = (I - \frac{V_i C_i}{C_i V_i})e^{A t_i^*}$, $V_i = A x_i^* + B_i$, t_i^* are the switching times, and x_i^* the state at the switching times.

Proof Consider a trajectory with initial condition $x(0) = x_0^*$. In the time interval before the first switching, this trajectory is defined as $x(t) = \Phi(t)x_0^* + \Gamma_1(t)u_{nl_{N_l-1}} + \Gamma_0(t)u_{nl_{N_l}}$. When x reaches the switching surface at time $t_1^* + \delta_1 t_1^*$, we have

$$x(t_1^* + \delta_1 t_1^*) = \Phi(t_1^* + \delta_1 t_1^*)(x_0^* + \delta_1 x_0^*) + \Gamma_1(t_1^* + \delta_1 t_1^*)u_{nl_{N_l-1}} + \Gamma_0(t_1^* + \delta_1 t_1^*)u_{nl_{N_l}}.$$

The series expansion in $\delta_1 t_1^*$ and $\delta_1 x_0^*$ is

$$x(t_1^* + \delta_1 t_1^*) = x_1^* + v_1 \delta_1 t_1^* + \Phi(t_1^*)\delta_1 x_0^* + O(\delta_1^2),$$

where $v_1 = A x_1^* + B_1 = A[\Phi(t_1^*)x_0^* + \Gamma_1(t_1^*)u_{nl_{N_l-1}} + \Gamma_0(t_1^*)u_{nl_{N_l}}] + B u_{nl_{N_l}}$.

Since the solution is on the switching surface at $t_1^* + \delta_1 t_1^*$, we have

$$C_1 x(t_1^* + \delta_1 t_1^*) + d_1 = C_1 x_1^* + C_1 v_1 \delta_1 t_1^* + C_1 \Phi(t_1^*)\delta_1 x_0^* + d_1 = 0,$$

and, therefore, the following equality holds:

$$\delta_1 t_1^* = -\frac{C_1 \Phi(t_1^*)}{C_1 v_1} \delta_1 x_0^*.$$

The rest of the proof follows as in [81]. The state after the first switch is

$$x(t_1^* + \delta_1 t_1^*) = x_1^* + \left(I - \frac{v_1 C_1}{C_1 v_1}\right) \Phi(t_1^*)\delta_1 x_0^* + O(\delta_1^2) = x_1^* + W_1 \delta_1 x_0^* + O(\delta_1^2). \quad (2.34)$$

Taking as initial condition $x_1^* + \delta_2 x_1^*$ and neglecting the $O(\delta^2)$ term, we have $x(t_2^* + \delta_2 t_2^*) = x_2^* + W_2 \delta_2 x_1^*$. Combining with (2.34) yields $\delta_2 x_1^* = W_1 \delta_1 x_0^*$. Replacing in the previous expression and applying successively for k eventually lead to

$$x(t_k^* + \delta_k t_k^*) = x_0^* + W_k W_{k-1} \dots W_1 \delta_1 x_0^* + O(\delta_1^2). \quad (2.35)$$

Neglecting the $O(\delta_1^2)$ term, the dynamics of Eq. (2.35), is stable if and only if the eigenvalues of $W = W_k W_{k-1} \dots W_1$ are inside the unit circle. This proves the proposition.

2.4 Analysis of the Limit Cycles

In the following sections, the expressions which take into account the effects of the event-based sampling at the output of the process and controller are presented. The method used is based on the grouping of all the continuous dynamics into one block

to obtain the state-space matrix (as shown in Fig. 2.3), and then to study the effect of the nonlinear feedback introduced by the sampling block.

To easily distinguish between the different types of controller and sampling, the following naming convention is used: *controller-SOD_n-process*, when the sampler is after the controller output, and *controller-process-SOD_n* if the sampler is placed after the process output, where *process* corresponds to the type of process considered (IPTD, SOPDT,...) and *controller* refers to the type of controller (PI, PD, PID,...). The index n refers to the number of hysteresis bands presented in the sampler. For example, PI-SOD₁-IPTD denotes the system composed by an IPTD process controlled by a PI controller with the sampler placed after the controller output, and a limit cycle with one hysteresis band (i.e., a system with relay feedback).

There are two directions in which the complexity of the analysis can be increased. The first one is to consider that the order of the process is increasing, i.e., a simple integrator, a double integrator, etc., and the second one is to consider an increase in the number of hysteresis bands of the sampler.

To simplify the analysis, it is worth noting that the solutions of (2.33) are linear on δ , thus the state and control signal can be normalized dividing by δ (note that $\delta > 0$).

2.4.1 Equilibrium Points

Consider the system (2.22). The set of equilibrium points is defined as $\mathcal{X} = \{x^* | \dot{x}^* = 0\}$. An equilibrium point is one in which the derivatives of the states are null, i.e., $x^* | \dot{x}^* = 0$. Since the derivatives are null, all the trajectories which enter into it at t_0 will stay for $t > t_0$. An immediate necessary condition to have an equilibrium point is that the linear system $Ax + B_i = 0$ has at least one solution. Note that, except for the P and PD controller, it is easy to see that $\det(A) = 0$, due to the integrator added by the controller, thus being possible to have a system of equations which is either indetermined or incompatible. The system does not have any solution if $rg(A) < rg(A|B_i)$, so an equilibrium point can exist only if $\exists i \in \mathbb{R} \mid rg(A) = rg(A|B_i)$.

Now assume that the output is within the k band of hysteresis, i.e., $x \in \Omega_k = \{x | \delta_k \leq y(t) = Cx(t) < \delta_{k+1}\}$ for some time interval $t_k \leq t < t_{k+1}$.

Proposition 2.4 *A necessary condition for the system to be ultimately bounded to Ω_k is that either $C[Ax + B_k] = 0$ or $C[Ax + B_{k+1}] = 0$.*

Proof Assume that $Cx(t)$ enters into Ω_k at t_0 . After a time $t > \tau$, the derivative of the system is $C\dot{x}(t) = C[Ax(t) + B_i]$, for $i \in \{k, k+1\}$. Thus, if $C[Ax(t) + B_i] \neq 0$ for both $i = k$ and $i = k+1$, the process output will eventually cross the boundaries of Ω_k for some t .

Computing the equilibrium points of the PI control with SOD sampling at the process output (the set-point is assumed to be null) yields

$$\begin{pmatrix} \dot{x}_p \\ \dot{x}_c \end{pmatrix} = \begin{pmatrix} A_p & k_i B_p \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_p \\ x_c \end{pmatrix} + \begin{pmatrix} -k_p B_p & B_p \\ -1 & 0 \end{pmatrix} \begin{pmatrix} \delta_i \\ w \end{pmatrix} = 0. \quad (2.36)$$

Thus, a necessary condition for the existence of an equilibrium point, from (2.36), is $\exists i \in \mathbb{Z} | \delta_i = 0$, because otherwise the integrator derivative is a non-null constant. Note that, if the set-point is not assumed to be null, the condition still holds with a slight modification: $\exists i \in \mathbb{Z} | \delta_i = y_{sp}$, i.e., the set-point must be an exact multiple of δ . Furthermore, since A_p is a non-singular matrix, the computation of the equilibrium point is straightforward: $x_c = \frac{-w}{k_i}$, $x_p = 0 \in \mathbb{R}^n$.

If the sampler affects to the controller output, then the equilibrium equation is

$$\begin{pmatrix} \dot{x}_p \\ \dot{x}_c \end{pmatrix} = \begin{pmatrix} A_p & 0 \\ -C_p & 0 \end{pmatrix} \begin{pmatrix} x_p \\ x_c \end{pmatrix} + \begin{pmatrix} B_p & B_p \\ -1 & 0 \end{pmatrix} \begin{pmatrix} \delta_i \\ w \end{pmatrix} = 0, \quad (2.37)$$

and the necessary condition to have an equilibrium point is $\exists i \in \mathbb{Z} | \delta_i = (1 - C_p A_p^{-1} B_p)^{-1} w$. If this expression holds, then the equilibrium point can be computed as $x_p = \frac{C_p^T}{\|C_p\|} \delta_i$, $x_c = (1 + k_p) \delta_i$.

The rest of this section presents an algorithm to compute the period and amplitude of a limit cycle in a generic process, and then shows examples of application to several common processes.

2.4.2 Algorithm

In this section, an algorithm to obtain computationally the period of a limit cycle and the intermediate switching times is outlined. Assume that $\alpha = 0.5$ and that the system presents a limit cycle in which the condition $y(t) = Cx(t) \in ((\alpha - N_l)\delta, (\alpha + N_l - 1)\delta)$ holds. The limit cycle is assumed to have only two changes in the sign of the derivative: one at the beginning of the first semiperiod and the other at the beginning of the second semiperiod. Thus, the limit cycle must have $4N_l - 2$ switchings. Because of the symmetry, the behavior of the limit cycle can be inferred by studying only the first semiperiod, thus reducing the complexity to $2N_l - 1$ levels. The algorithm, which can be implemented either in a symbolic or in a numerical computation tool, is as follows:

- Initialization:
 1. Set N_l as the number of levels crossed within the limit cycle.
 2. Fix the values of k_p , k_i , α , δ , w , τ , and the matrices A and B .
 3. Calculate $\Phi(t) = e^{At}$, $\Gamma_0(t) = \int_0^{t-\tau} e^{As} ds$, and $\Gamma_1(t) = \int_{t-\tau}^t e^{As} ds$.
- To calculate the period:
 1. For i from 1 to $2N_l$ repeat steps 2–3.
 2. If $i \in (1, N_l)$, set $j_i = i - \lfloor \frac{N_l}{2} \rfloor$, else $j_i = \lfloor \frac{N_l}{2} \rfloor + N_l - i$.

3. Set $u_{nl_i} := (j_i + \alpha)\delta$, and,
 - $x_{c_i} = -\frac{u_{nl_i} + k_p x_{p_i}}{k_i}$, if sampling the controller output, or,
 - $x_{p_i} = u_{nl_i}$, if sampling the process output.
 4. Set $eq_i := -x_{i+1} + \Phi(t_i)x_i + \Gamma_1(t_i)u_j + \Gamma_0(t_i)u_i = 0$.
 5. Solve the system of equations given by eq_i , with the unknowns t_i , and x_{p_i} or x_{c_i} .
 6. $T = \sum_{i=1}^{2N_l} t_i$.
- To calculate the amplitude:
 1. Set $j_{max} = j|(x_j > x_i, \forall i \neq j)$ and $j_{min} = j|(x_j < x_i, \forall i \neq j)$.
 2. Find $t_{min} = \min(\tau, t|C\dot{x}_{j_{min}}(t) = 0)$, and $t_{max} = \min(\tau, t|C\dot{x}_{j_{max}}(t) = 0)$, corresponding to the minimum and maximum values of the output.
 3. Compute the amplitude of the process output, $\Delta_p = C_p[x(t_{max}) - x(t_{min})]$, and the control input, $\Delta_c = C_c[x(t_{max}) - x(t_{min})]$.

2.4.3 Examples of Analysis

To illustrate the application of Proposition 2.2 and the use of the algorithm of Sect. 2.4.2, the analysis of several systems (IPTD, SOPTD, FOPTD, and SOPTD) is detailed in the following lines. The results are summarized in Table 2.1, at the end of the section.

2.4.3.1 IPTD Process

Let us consider an IPTD process $P(s) = \frac{k}{s}e^{-\tau s}$, which can be described by its state-space model in the canonical observable form as

$$\begin{aligned}\dot{x}(t) &= -ku(t - \tau) + kw(t) \\ y(t) &= x(t) = x_p(t),\end{aligned}\tag{2.38}$$

where k is the process gain, x is the state, u is the process input, and y is the process output. First it is presented the approach for the PI-IPTD-SOD_n structure, and then for the PI-SOD_n-IPTD.

Process variable sampling (PI-IPTD-SOD₁) According to (2.6), the state feedback matrix corresponding to the system controlled by a PI with SOD sampling at the process output is

$$\begin{aligned}\begin{pmatrix} \dot{x}_p(t) \\ \dot{x}_c(t) \end{pmatrix} &= \begin{pmatrix} 0 & kk_i \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} kk_p & k \\ 1 & 0 \end{pmatrix} \begin{pmatrix} u_{nl_k} \\ w \end{pmatrix} \\ y(t) &= (1 \ 0) \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix}.\end{aligned}\tag{2.39}$$

Assuming there exists a stable limit cycle with two states, then the equations that allow us to obtain the amplitude and period of the oscillations are

$$\begin{aligned}\Phi(t_1)x_1 + \Gamma_1(t_1)u_2 + \Gamma_0(t_1)u_1 - x_2 &= 0 \\ \Phi(t_2)x_2 + \Gamma_1(t_2)u_1 + \Gamma_0(t_2)u_2 - x_1 &= 0 \\ x_{p1} &= -u_{nl1} = \alpha\delta \\ x_{p2} &= -u_{nl2} = (\alpha - 1)\delta,\end{aligned}\tag{2.40}$$

where $x_i = [x_{p_i} \ x_{c_i}]^T$, and $u_i = [u_{nl_i} \ w]^T$, are the state and input, respectively.

The matrices Φ , Γ_0 , and Γ_1 can be calculated as

$$\Phi(t) = \begin{pmatrix} 1 & kk_i t \\ 0 & 1 \end{pmatrix}\tag{2.41}$$

$$\begin{aligned}\Gamma_0(t) &= \begin{pmatrix} kk_p t - kk_p \tau + \frac{1}{2}kk_i t^2 - kk_i t \tau + \frac{1}{2}kk_i \tau^2 & k(t - \tau) \\ t - \tau & 0 \end{pmatrix} \\ \Gamma_1(t) &= \begin{pmatrix} kk_p \tau + kk_i t \tau - \frac{1}{2}kk_i \tau^2 & k\tau \\ \tau & 0 \end{pmatrix}.\end{aligned}\tag{2.42}$$

Introducing (2.41) in (2.40), and solving the resulting equations, the period of the limit cycle can be obtained (see Table 2.1). Looking at the expression of the period, it can be seen that the symmetry of the limit cycle, i.e., the difference between the two semiperiods t_1 and t_2 , depends on the offset of the sampler α . In particular, for $\alpha = 0.5$, the two semiperiods have the same value. When $\alpha = 0$, t_2 vanishes, which can be interpreted as this limit cycle cannot exist. In this case, either the system will reach a steady-state or enter into a limit cycle with higher number of levels. With respect to the disturbance rejection, it can be seen that w does not affect the period. This is because it is rejected by the integrator, which changes its mean value to absorb the disturbance.

With the switching times t_1 and t_2 , the amplitudes of the oscillations can be computed. It is necessary to find the maximum and minimum of the output, which correspond to the times when its first derivative is null, i.e., $\dot{x}_p(t) = kk_i x_c(t) + kk_p u_{nl_k} + kw = 0$. By solving the previous expression, the value of x_c and the times when the peaks are reached can be obtained, and so for this case it can be found an analytic expression for the amplitude of the process output, A_{po} , and of the control input, A_{ci} (see Table 2.1).

Controller variable sampling (PI-SOD₁-IPTD) When the sampler is placed at the controller output, the description of the system in the state-space model is given by expressions (2.7). Thus, for this process, it is obtained

$$\begin{aligned}\begin{pmatrix} \dot{x}_p(t) \\ \dot{x}_c(t) \end{pmatrix} &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ k & k \end{pmatrix} \begin{pmatrix} u_{nl_k} \\ w \end{pmatrix} \\ y(t) &= (k_p \ k_i) \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix}.\end{aligned}\tag{2.43}$$

Assuming there exists a stable limit cycle with two states, then the equations that allow to obtain the amplitude and period of the oscillations are

$$\begin{aligned}
 \Phi(t_1)x_1 + \Gamma_1(t_1)u_2 + \Gamma_0(t_1)u_1 - x_2 &= 0 \\
 \Phi(t_2)x_2 + \Gamma_1(t_2)u_1 + \Gamma_0(t_2)u_2 - x_1 &= 0 \\
 k_px_{p1} + k_ix_{c1} = u_{nl1} &= \alpha\delta \\
 k_px_{p2} + k_ix_{c2} = u_{nl2} &= (\alpha - 1)\delta.
 \end{aligned} \tag{2.44}$$

Here, the amplitude of the limit cycle can be computed directly, since the control input is piecewise constant and the switching times and values are known. The period of the limit cycle can be obtained by solving the system of Eqs. (2.44) (see Table 2.1).

As opposed to the process sampling, here the disturbance appears in the expression of the semiperiods, thus affecting to the symmetry of the limit cycle. It is possible to have oscillations where the process is changing slowly nearly all the time and then to have an abrupt change. Since in one semiperiod the control action is more aggressive, this decreases the margin of delay that can be added to the system without reaching the next sampling level.

To obtain the expression corresponding to the amplitude, and since the maximum and minimum values of the process output are reached at times $t_1 + \tau$ and $t_2 + \tau$, integrating (2.43) yields the desired result (see Table 2.1).

2.4.3.2 DIPTD Process

It is well known in classic control theory that the double integrator process cannot be stabilized with a continuous PI controller, due to the 90° phase lag of each integrator. It becomes then necessary to introduce the derivative action (PD controller) to compensate this lag. In the same way, either with the PI-SOD_n-DIPTD or PI-SOD_n-DIPTD, the system will oscillate with unbounded growing amplitudes. Although it is not in the scope of this work, it should be possible to stabilize this kind of process by a SOD-PD controller. In practice, the implementation of the derivative action must be carefully studied, because it can be problematic specially when the sampler is placed after the process output, since the estimation of the derivative may be poor.

2.4.3.3 FOPTD Process

The process considered in this section is a FOPTD process $P(s) = \frac{k}{Ts+1}e^{-\tau s}$, which is described in the state-space system by the following expressions:

$$\begin{aligned}
 \dot{x}(t) &= -\frac{1}{T}x(t) + \frac{k}{T}u(t - \tau) + w(t) \\
 y(t) &= x(t) = x_p(t),
 \end{aligned} \tag{2.45}$$

where k is the process gain, x is the state, u is the process input, y is the process output, and w is an external disturbance.

Process variable sampling (PI-FOPTD-SOD₁) The expressions corresponding to the PI-FOPTD-SOD₁ are as follows:

$$\begin{aligned} \begin{pmatrix} \dot{x}_p(t) \\ \dot{x}_c(t) \end{pmatrix} &= \begin{pmatrix} \frac{1}{T} & \frac{kk_i}{T} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} \frac{kk_p}{T} & \frac{k}{T} \\ 1 & 0 \end{pmatrix} \begin{pmatrix} u_{nlk} \\ w \end{pmatrix} \\ y(t) &= (k_p \ k_i) \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix}. \end{aligned} \quad (2.46)$$

From (2.46), the matrices Φ , Γ_0 , and Γ_1 can be obtained as

$$\begin{aligned} \Phi(t) &= \begin{pmatrix} e^{\frac{t}{T}} & kk_i(e^{\frac{t}{T}} - 1) \\ 0 & 1 \end{pmatrix} \\ \Gamma_0(t) &= \begin{pmatrix} -k(k_p + k_i T)(e^{\frac{t-\tau}{T}} - 1) + kk_i(t - \tau) T(e^{\frac{t-\tau}{T}} - 1) & 0 \\ t - \tau & 0 \end{pmatrix} \\ \Gamma_1(t) &= \begin{pmatrix} k(k_p + k_i T)(e^{\frac{t-\tau}{T}} - e^{\frac{t}{T}}) - kk_i \tau - T(e^{\frac{t-\tau}{T}} - e^{\frac{t}{T}}) & 0 \\ \tau & 0 \end{pmatrix}. \end{aligned} \quad (2.47)$$

As it can be seen in the previous expressions, since the system of equations obtained for the FOPTD contains terms involving exponentials, it is not possible to find analytical solutions, as opposed to the case of IPTD processes. Instead of this, the solutions have to be found numerically. However, the algorithm proposed in Sect. 2.4.2 can be applied.

Controller variable sampling (PI-SOD₁-FOPTD) The expressions corresponding to the PI-SOD₁-FOPTD are the following:

$$\begin{aligned} \begin{pmatrix} \dot{x}_p(t) \\ \dot{x}_c(t) \end{pmatrix} &= \begin{pmatrix} -\frac{1}{T} & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} \frac{k}{T} & \frac{k}{T} \\ 1 & 0 \end{pmatrix} \begin{pmatrix} u_{nlk} \\ w \end{pmatrix} \\ u(t) &= (k_p \ k_i) \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix} \end{aligned} \quad (2.48)$$

$$y(t) = (1 \ 0) \begin{pmatrix} x_p(t) \\ x_c(t) \end{pmatrix}. \quad (2.49)$$

As in the previous case, the solutions of the equations must be found by means of numerical tools.

2.4.3.4 SOPTD Process

The process considered in this section is a SOPTD $P(s) = \frac{k}{(\tau_1 s + 1)(\tau_2 s + 1)} e^{-\tau s}$, which is described in the state-space system by the following expressions:

$$\begin{aligned}\ddot{x}(t) &= -\frac{1}{\tau_1 \tau_2} x - \frac{\tau_1 + \tau_2}{\tau_1 \tau_2} \dot{x} + \frac{k}{\tau_1 \tau_2} u(t - \tau) + w(t) \\ y(t) &= x(t) = x_p(t),\end{aligned}\quad (2.50)$$

where k is the process gain, τ_1 and τ_2 are the time constants, x is the state, u is the process input, y is the process output, and w is an external disturbance.

Process variable sampling (PI-SOPTD-SOD_n) The expressions corresponding to the PI-SOPTD-SOD_n are the following ones:

$$\begin{aligned}\begin{pmatrix} \dot{x}_p(t) \\ \ddot{x}_p(t) \\ \dot{x}_c(t) \end{pmatrix} &= \begin{pmatrix} 0 & 1 & 0 \\ -\frac{1}{\tau_1 \tau_2} & -\frac{\tau_1 + \tau_2}{\tau_1 \tau_2} & \frac{k k_i}{\tau_1 \tau_2} \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_p(t) \\ \dot{x}_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ \frac{k k_p}{\tau_1 \tau_2} & \frac{k}{\tau_1 \tau_2} \\ 1 & 0 \end{pmatrix} \begin{pmatrix} u_{nlk} \\ w \end{pmatrix} \\ y(t) &= \begin{pmatrix} 1 & 0 & 0 \\ k_p & 0 & k_i \end{pmatrix} \begin{pmatrix} x_p(t) \\ \dot{x}_p(t) \\ x_c(t) \end{pmatrix}.\end{aligned}\quad (2.51)$$

As in the FOPTD case, for this system, it is not possible to find analytical solutions due to the exponentials that appear in the equations. Therefore, numerical methods must be used to find the solutions.

Controller variable sampling (PI-SOD_n-SOPTD) The expressions corresponding to the PI-SOD_n-SOPTD are the following:

$$\begin{aligned}\begin{pmatrix} \dot{x}_p(t) \\ \ddot{x}_p(t) \\ \dot{x}_c(t) \end{pmatrix} &= \begin{pmatrix} 0 & 1 & 0 \\ -\frac{1}{\tau_1 \tau_2} & -\frac{\tau_1 + \tau_2}{\tau_1 \tau_2} & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_p(t) \\ \dot{x}_p(t) \\ x_c(t) \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ \frac{k}{\tau_1 \tau_2} & \frac{k}{\tau_1 \tau_2} \\ 1 & 0 \end{pmatrix} \begin{pmatrix} u_{nlk} \\ w \end{pmatrix} \\ y(t) &= \begin{pmatrix} 1 & 0 & 0 \\ k_p & 0 & k_i \end{pmatrix} \begin{pmatrix} x_p(t) \\ \dot{x}_p(t) \\ x_c(t) \end{pmatrix}.\end{aligned}\quad (2.52)$$

As in the previous case, the solutions of the equations must be found by means of numerical tools.

2.4.4 Implementation in MATLAB[®]

The algorithm to obtain the periods and amplitudes of the simulation examples and the models identified from experimental data has been implemented in MATLAB. The code is shown in Listing 2.1. First, the system matrices are defined, corresponding to the PI controller with sampling at the process output (lines 7–11), and with the sampling at the controller output (lines 12–16). Then, the type, order, and parameters of the sampler are stored in the variables `sampling`, `delta`, and `alpha` (lines 17–22). Finally, the set of equations is defined and solved in lines 33–38.

Table 2.1 Summary table with the limit cycle periods and amplitudes

$P(s)$	PI-process-SOD _I	PI-SOD _I -process
$\frac{k}{s}e^{-\tau s}$	$T = \frac{1}{2} \frac{kk_i \tau^2 - 2kk_p \tau - 2}{k(k_p - k_i \tau)(\alpha - 1)\alpha}$ $t_1 = (1 - \alpha)T, t_2 = \alpha T$ $\Delta_{po} = \frac{\delta}{2} \frac{kk_i \tau^2 - 2kk_p \tau - 2}{k_p - k_i \tau}$ $\Delta_{co} = \frac{k_p^2 + k_i}{k} \Delta_{po}$	$T = \frac{1}{2} \frac{kk_i \tau^2 - 2kk_p \tau - 2}{k(k_p - k_i \tau)(\alpha - 1 + \frac{w}{\delta})(\alpha + \frac{w}{\delta})}$ $t_1 = (1 - \alpha - \frac{w}{\delta})T,$ $t_2 = (\alpha + \frac{w}{\delta})T$ $\Delta_{po} = \frac{\delta}{2} \frac{kk_i \tau^2 - 2kk_p \tau - 2}{k_p - k_i \tau}$ $\Delta_{co} = \delta$
$\frac{k}{s^2}e^{-\tau s}$	Limit cycle does not exist	
$\frac{e^{-\tau s}}{\tau_1 s + 1}$	$T, \Delta_{po},$ and Δ_{co} can be obtained using numerical methods	
$\frac{e^{-\tau s}}{(\tau_1 s + 1)(\tau_2 s + 1)}$		

Δ_{po} denotes the amplitude of the process output, and Δ_{co} is the amplitude of the controller output

Listing 2.1 Limit Cycle Finder algorithm

```

1 %% Implementation of the Limit Cycle Finder Algorithm
2 clear all; clc;
3 numberOfProcessStates = size(Ap, 1);
4 numberOfProcessInputs = size(Bp, 1);
5 numberOfProcessOutputs = size(Cp, 1);
6 I = eye(numberOfProcessStates+1);
7 % Process Sampling
8 Aps = [Ap ki*Bp; zeros(1, numberOfProcessStates+1)];
9 Bps = [kp*Bp Bp; 1 0];
10 Cps = [Cp zeros(numberOfProcessOutputs, 1)];
11 Dps = [zeros(numberOfProcessOutputs, numberOfProcessInputs+1);
        -kp zeros(1, numberOfProcessInputs)];
12 % Controller Sampling
13 Acs = [Ap ki*Bp; zeros(1, numberOfProcessStates+1)];
14 Bcs = [kp*Bp Bp; 1 0];
15 Ccs = [Cp zeros(numberOfProcessOutputs, 1)];
16 Dcs = [zeros(numberOfProcessOutputs, numberOfProcessInputs+1);
        -kp zeros(1, numberOfProcessInputs)];
17 % Type of sampling ('process', 'controller')
18 sampling = 'process';
19 delta = 1.0;
20 alpha = 0.5;
21 % Hysteresis bands
22 n = 1;
23 % Switchings
24 if (alpha == 0.0)
25     m = 2*n-2;
26     u = [(alpha-n+1):(alpha+n-2); ones(1,m)*disturbance];
27 elseif (alpha == 0.5)
28     m = 2*n-1;
29     u = [(alpha-n+1):(alpha+n-1); ones(1,m)*disturbance];
30 end
31
32 tic,
33 for tau = 0:0.1:1
34     f = @(t) slc(t, u, Aps, Bps, Cps, tau);
35     Tguess = 5;
36     k = Tguess*rand(1, m);
37     [sol, val] = fsolve(f, k);
38 end
39 elapsedTime = toc;

```

2.5 Simulation Results

This section shows simulations which illustrate the behavior of the different combinations of control schemes and processes commented in the previous sections.

2.5.1 PI-IPTD-SOD_n and PI-SOD_n-IPTD

Let us consider an IPTD process controlled by a PI controller with event-based sampling, and let the values of the plant parameters be $k = 1.0$, $\tau = 0.2$. The controller gains $k_p = 1.2$, $k_i = 1.0$ have been chosen to produce a limit cycle with two states, and the sampler $\alpha = 0.5$, $\delta = 0.1$. The system is described by the following expressions:

$$\begin{pmatrix} \dot{x}_p(t_k) \\ \dot{x}_c(t_k) \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1.0 & 0 \end{pmatrix} \begin{pmatrix} x_p(t_k) \\ x_c(t_k) \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 1.2 & 1 \end{pmatrix} \begin{pmatrix} u_{nl_k} \\ w \end{pmatrix}. \quad (2.53)$$

There exists a symmetric limit cycle with two states (see Fig. 2.7a), with period $T = 5.6093$ and amplitude $\Delta_{po} = 0.2095$, which have been computed with the algorithm presented in Sect. 2.4.2. With the chosen gains, the system converges to the limit cycle after introducing a step change in the set-point.

When the sampler is placed at the controller output, the limit cycle that appears (see Fig. 2.7c) has the same period $T = 5.6093$ but a different amplitude $\Delta_{po} = 0.1402$. While in the first case an external disturbance does not vary the properties of the limit cycle, in the second case it does as it is shown in Fig. 2.8.

Varying the parameters of the controller, it is possible to obtain limit cycles with higher number of levels. For example, increasing the integral gain of the controller to $k_i = 2.0$, and also the order of the sampler to $N_l = 2$, the system with the sampler at the process output presents the response shown in Fig. 2.7b and with the sampler at the controller output, it has the response of Fig. 2.7d. The period and amplitude of the oscillation computed are $T = 4.9745$ and $\Delta_{po} = 0.3584$, which correspond to the results obtained in the simulation.

The simulations show that the system, with the chosen parameters, converges to a limit cycle even in the presence of constant disturbances. The local stability of the limit cycles can also be proven by applying Proposition 2.3. As an example, for the two-level limit cycle, looking at the eigenvalues of the matrix $W = W_2 W_1$, where

$$W_i = \begin{pmatrix} 1 - u_{nl_i} & 0 \\ -x_{p_i} + 1.2(u_{nl_i} + w) + t_i^* & 1 \end{pmatrix}.$$

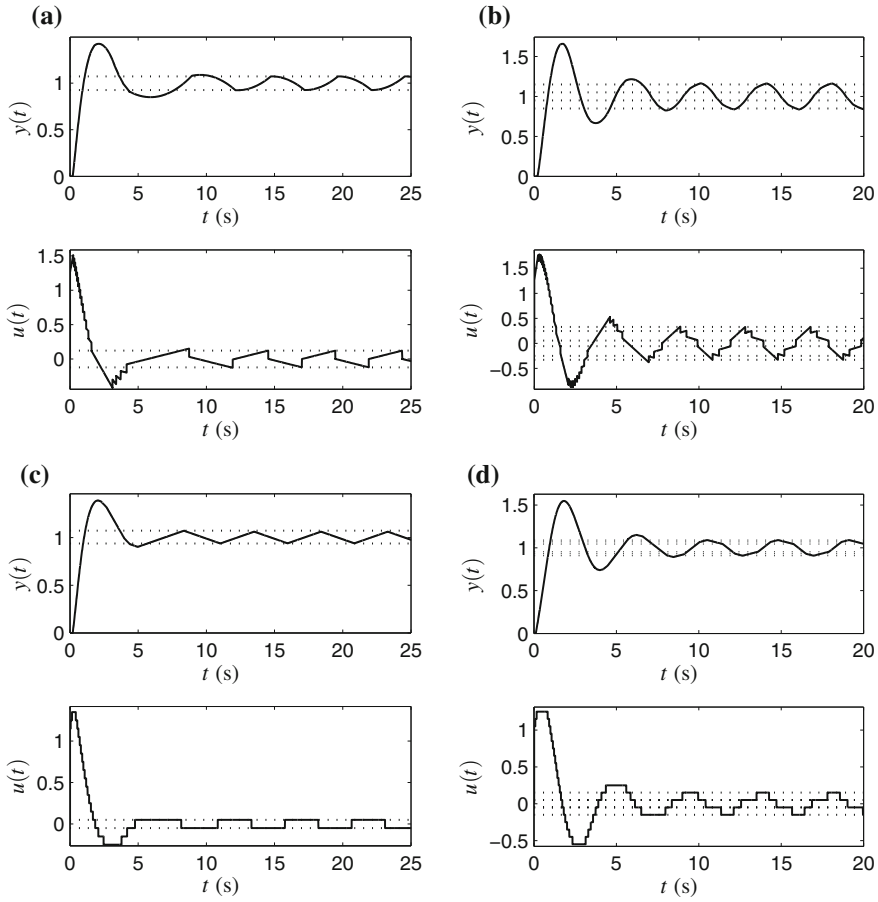


Fig. 2.7 Limit cycles in an IPTD process controlled by a PI with event-based sampling at the process output with one hysteresis band (a) and two (b), and with sampling at the controller output with one hysteresis band (c) and two (d). The dotted lines show the sampling levels of the process variable in (a), (c) and of the control variable in (b), (d), and the value at the switching times of the control variable in (a), (c) and of the process variable in (b), (d)

after straightforward computations, it can be shown that the eigenvalues of W are $\lambda_1 = 1$, $\lambda_2 = (1 - \alpha)\alpha < 1$, and therefore the limit cycle is locally stable. If limit cycles with $2n$ switchings are considered, then the eigenvalues of W are $\lambda_1 = 1$, $\lambda_2 = \prod_{i=0}^{2N_l-1} (1 - \alpha - N_l + i)$. It is easy to verify that $|\lambda_2| \leq 1$ for every α when $N_l = 1, 2$, i.e., the corresponding limit cycles are locally stable. However, for $N_l > 2$, the local stability of the limit cycles depends on the particular value of α .

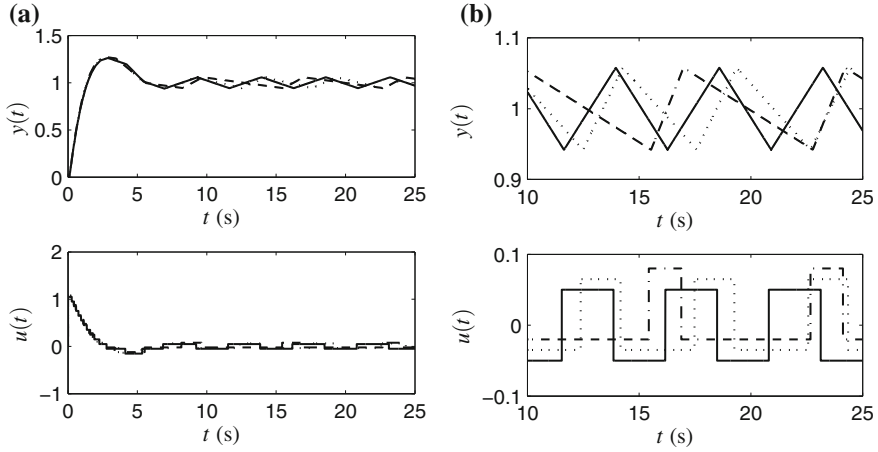


Fig. 2.8 Limit cycles in an IPTD process controlled by a PI with event-based sampling at the controller output **a** with an external disturbance $w = 0.0$ (solid line), $w = 0.015$ (dotted line), and $w = 0.03$ (dashed-dotted line). **b** Detail of the limit cycles

2.5.2 PI-SOPTD-SOD_n

Now, consider a SOPTD with parameters $k = 1.0$ (gain), $\tau_1 = 1.0$, $\tau_2 = 0.5$ (time constants), and $\tau = 0.2$ (time delay), which is controlled by a PI with event-based sampling placed at the process output, with $\alpha = 0.5$ and $\delta = 0.1$. Setting the controller gains to $k_p = 1.2$ and $k_i = 1.0$, the matrices A and B of the system are

$$A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 2 & -2 & -3 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 2.4 & 2 \end{pmatrix}. \quad (2.54)$$

Solving the system of equations corresponding to the limit cycle with one band of hysteresis, the values of the switching times and levels are

$$\begin{aligned} t_1 &\approx 3.0772, \quad t_2 \approx 3.0772, \\ \dot{x}_{p1} &\approx -0.0517, \quad \dot{x}_{p2} \approx 0.0517, \\ \dot{x}_{c1} &\approx -0.0669, \quad \dot{x}_{c2} \approx 0.0669. \end{aligned}$$

Finally, the period is $T = t_1 + t_2 \approx 6.1545$ and the amplitude of the process output is $A \approx 0.1338$. This limit cycle, and another composed of two hysteresis bands, is plotted in Fig. 2.9.

It is interesting to note that, when the sampler is placed at the process output, the limit cycle does not vary when a constant disturbance affects the system, because it is rejected by the controller.

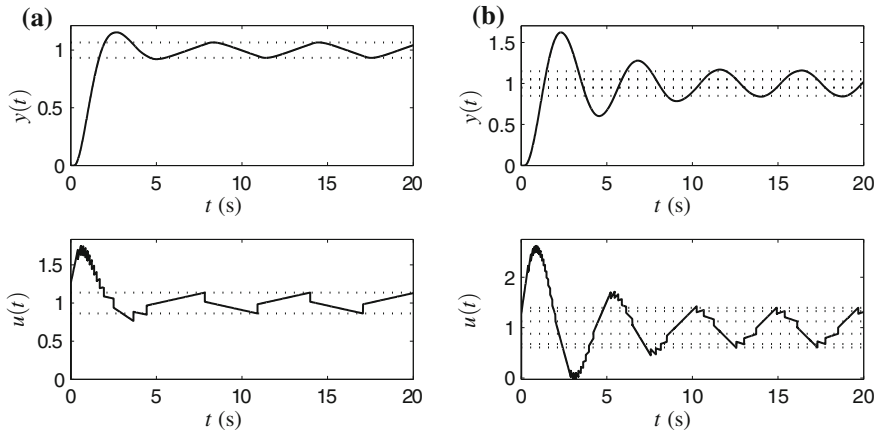


Fig. 2.9 Limit cycles in an SOPTD process controlled by a PI with event-based sampling at the controller output, involving **a** one, and **b** two hysteresis bands

The local stability of the limit cycle can be studied by looking at the eigenvalues of the matrix W :

$$W = W_2 W_1 = \begin{pmatrix} 0 & 0 & 0 \\ -0.0787 - 2.5953w & 0.0042 & 0.0021 \\ 0.0699 + 2.4253w & -0.0042 & -0.0021 \end{pmatrix}. \quad (2.55)$$

Since the eigenvalues of W are inside the unit circle ($\lambda_1 \approx 0.0021$, $\lambda_2 \approx 0.0000$, $\lambda_3 \approx 0.0000$), the limit cycle is locally stable.

2.6 Experimental Results

This section shows experimental results which clearly evidence the existence of the results derived in the previous sections in a real system. Therefore, the experiences carried out were focused on finding limit cycles in the Acurex system to compare them with that predicted by the theory and simulations. As shown in the following paragraphs, it is worth noting that even when the model used is a simplification of the process which ignores many of the complex dynamics existent in the system, the results are very close to those predicted in theory. Below, the Acurex system and the model identified from experimental data are presented, commenting some implementation issues of the controller and, finally, the obtained results are shown and interpreted.



Fig. 2.10 Acurex distributed collector system at the PSA, Spain

2.6.1 The Acurex System

The experiments have been done with an equipment, known as Acurex, built in 1981 at the PSA, Spain [228]. In this plant (Fig. 2.10), two types of collecting systems were considered, a central receiver system (CRS) and a distributed collector system (DCS) using parabolic troughs. Parabolic trough systems concentrate sunlight onto a receiver pipe which contains a heat transfer fluid (HTF) that is heated as it flows along the receiver pipe. Then, the HTF is used to produce steam that may be used for example to feed an industrial process. A survey of basic and advanced control approaches for distributed solar collector fields can be found in [29, 30]. For more information on control of solar plants see [31].

2.6.2 The Model

The plant was identified as a FOTPD, where the process input is the oil flow (l/s) in the pipes and the process output is the temperature of the oil at the collector field outlet ($^{\circ}\text{C}$). There are unmodeled dynamics that are considered as external disturbances, such as the oil temperature at the input or the solar irradiance. However, because of the time scale of the tests performed, which is smaller than the rate of variation of these variables (under clear day conditions without clouds), the model obtained seems to be a valid representation of the process for our purposes.

The transfer function was identified from experimental data obtained from the plant in a set of step tests. The procedure followed in each test is to drive the temperature manually to the working point, and when the process has reached it to introduce a step change in the input, registering the data measured from the sensors until the process stabilizes again. The FOPTD model, obtained by applying a least-squares procedure, is

$$P(s) = -\frac{6.0715}{103.2723s + 1}e^{-67.5021s}, \quad (2.56)$$

where the time constant τ_1 and time delay τ are given in seconds, and the gain k in $^{\circ}\text{C}/\text{l}$. The tests were carried out with an input around 8.5 l/s, being the range of the pump from 2 to 12 l/s. Also, the time constants and delays obtained make sense from the previous published works in this field. Notice the minus sign in (2.56), which represents the inverse response of the plant, i.e., a positive change in the flow produces a negative change in the temperature.

2.6.3 Implementation

The Acurex system has a SCADA software developed in LabVIEWTM. This software provides the user with an interface to execute its own controller implementation in MATLAB code. Thus, one must write a MATLAB callback function which is invoked with a configurable sampling period (it was fixed to $T_s = 15$ s). This function receives the measures from the sensors, updates the controller state and, finally, sends the new control action to the actuators.

The controller implementation can be configured to work in three modes, namely,

- **manual** the control input is set manually,
- **SOD-PI** the sampler is at the process output, and
- **PI-SOD** the sampler is at the controller output.

An excerpt of the code of the controller is shown in Listings 2.2 and 2.3.

Listing 2.2 Code of the controller with the sampler at the process output

```

1 % Sampling at the process output
2 e = setpoint - output;
3 if ~( (u_prev >= umax && e > 0) || (u_prev <= umin && e < 0) )
4     I = I + e_prev*Ts;
5 end
6
7 % event detection
8 if (abs(e - e_prev) > delta)
9     levels = floor(abs(e - e_prev) / delta);
10    e_prev = e_prev + sign(e - e_prev)*delta*levels;
11    u_prev = sat(Kp*e_prev + Ki*I, umin, umax);
12    I = (u_prev - Kp *e_prev) / Ki;
13 end

```

Listing 2.3 Code of the controller with the sampler at the controller output

```

1 % Sampling at the controller output
2   e = - output;
3
4   % anti-windup
5   if (~((u_prev >= umax && e > 0) || (u_prev <= umin && e <
6       0)))
7       I = I + e*Ts;
7       u = Kp*e + Ki*I;
8   else
9       u = u_prev;
10  end
11
12  % event detection
13  if (abs(u - u_prev) > delta)
14      levels = floor(abs(u - u_prev) / delta);
15      u_prev = sat(u_prev + sign(u - u_prev)*delta, umin,
16                  umax);
16  end

```

2.6.3.1 Controller Sampling

The first set of tests presented was carried out with the controller in *PI-SOD* mode, with the purpose of reproducing the two-level limit cycles obtained in simulation (with $\alpha = 0.5$) and the three-level limit cycles (with $\alpha = 0.0$). The procedure followed is the same for each test, first the system temperature is moved to an operating point, and next when the process reaches a steady state, a set-point step change is introduced into the controller.

The response is shown in Fig. 2.11, where the oil temperature, the flow, and the solar irradiance during the test are plotted. It can be seen that the system goes into a limit cycle composed of two levels, which is similar to the results obtained in simulation with the FOPTD model.

Figure 2.12a, b shows the comparison of the limit cycles obtained in simulation with the model and the results obtained with the Acurex system. The results are similar both qualitatively (limit cycles with two states) and quantitatively (the period and amplitude are approximately equal).

2.6.3.2 Process Sampling

The second set of tests was carried out with the sampler placed at the process output. After verifying that, as in the previous section, the system enters into a limit cycle of two levels (Fig. 2.13), the existence of more complex limit cycles was investigated. Increasing the proportional gain, a limit cycle with eight different levels was found, which is shown in Fig. 2.14. It is remarkable that even in this case, the comparison between the experimental data and the simulated process shows that there are no significative differences in the behavior.

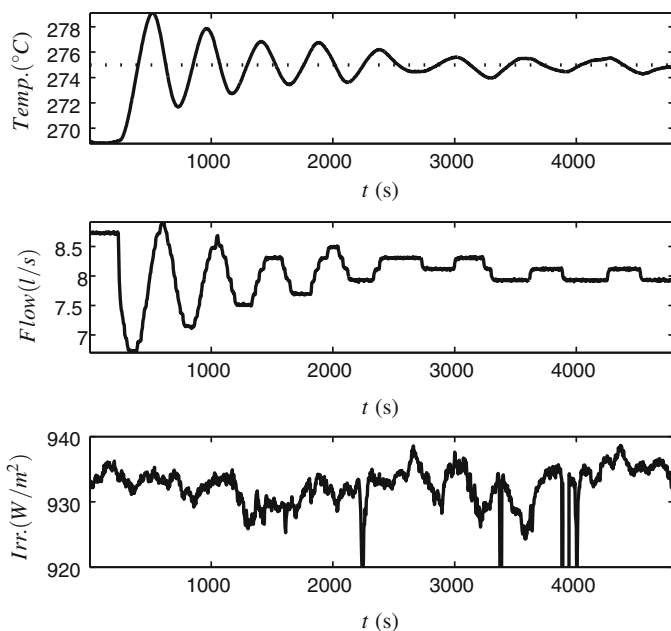


Fig. 2.11 Response of the Acurex system (*solid line*) to a step change in the set-point, with the event-based sampler placed at the controller output

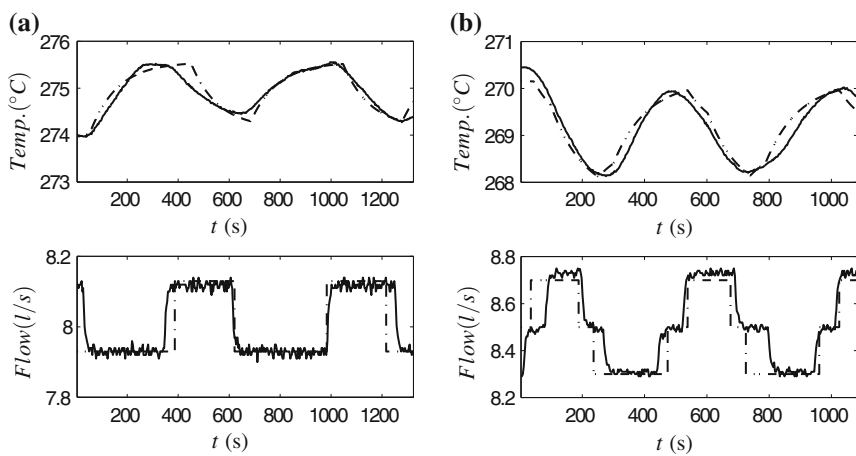


Fig. 2.12 Limit cycles in the Acurex system (*solid line*) and in the simulated model (*dashed line*) controlled by a PI with SOD sampling placed after the controller output. **a** With two levels and **b** with three levels

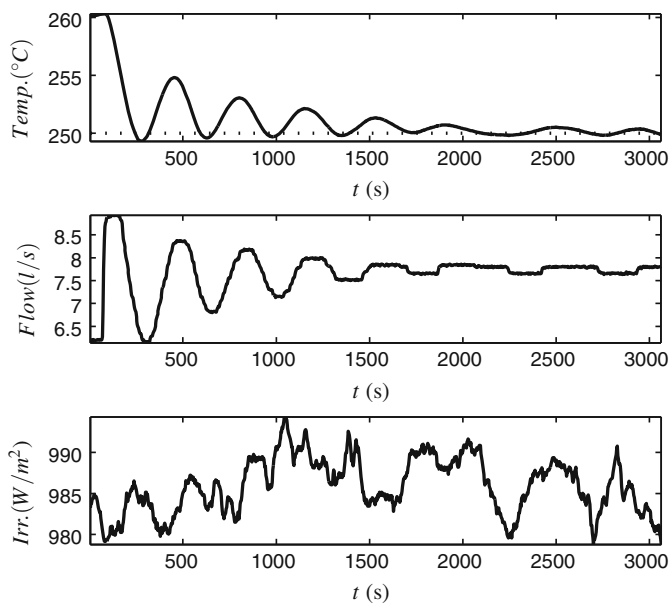


Fig. 2.13 Response of the Acurex system (*solid line*) to a step change in the set-point, with the event-based sampler placed at the process output

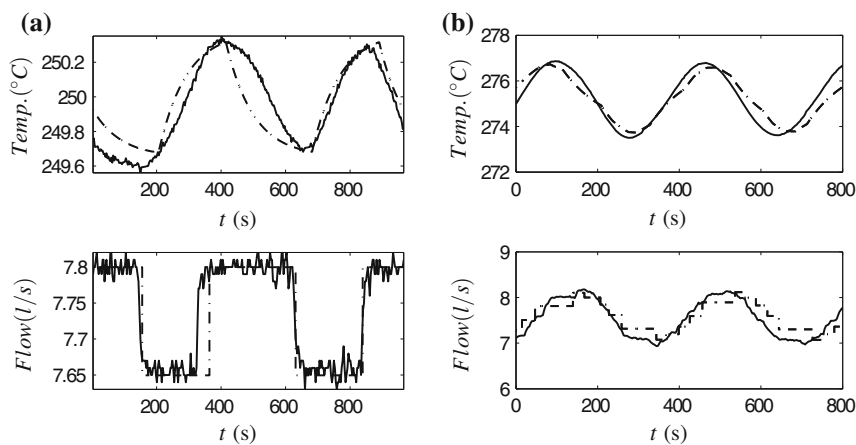


Fig. 2.14 Limit cycles in the Acurex system (*solid line*) and in the simulated model (*dashed line*) controlled by a PI with SOD sampling placed after the process output. **a** With two levels and **b** with eight levels

2.7 Conclusions

The behavior of a control system based on the use of a level crossing sampling either in the process output or in the control output has been studied. Limit cycles are of particular interest since they are associated to oscillations in processes, and therefore it is worth gaining knowledge about them in order to avoid them when possible or to assure that they are not problematic.

When trying to find properties about the limit cycles, it is common to have systems of equations involving transcendent functions and thus it is not possible, in general, to find closed-form solutions. Moreover, due to the combinatorial explosion, it can be computationally expensive to find these solutions, and it becomes harder when higher order process models are considered.

Therefore, an algorithm to analyze the properties of the limit cycles has been proposed; it allows us to introduce some knowledge in the hypothesis of the problem, so that the complexity can be reduced.

A set of simulation results illustrates the behavior of the controllers with some models frequently used in industrial context, which are the IPTD, the FOPTD, and the SOPTD. Also, this behavior has been tested and verified in the Acurex Field of the Solar Platform of Almería, Spain. The experiments performed confirm that the simulation results can be extrapolated to real cases, obviously with the divergences due to unmodeled dynamics of the process, disturbances, etc.

<http://www.springer.com/978-3-319-21298-2>

Asynchronous Control for Networked Systems

Guinaldo Losada, M.; Rodríguez Rubio, F.; Dormido, S.

(Eds.)

2015, XXIV, 339 p. 127 illus., 102 illus. in color.,

Hardcover

ISBN: 978-3-319-21298-2