A PLC-based System for Advanced Control

Samo Gerkšič[1], Gregor Dolanc[1], Damir Vrančić[1], Juš Kocijan[1,6], Stanko Strmčnik[1], Sašo Blažič[2], Igor Škrjanc[2], Zoran Marinšek[3], Miha Božiček[3], Anna Stathaki[4], Robert King[4], Mincho Hadjiski[5], Kosta Boshnakov[5]

[1] Department of Systems and Control, Jožef Stefan Institute, Ljubljana, Slovenia

[2]University of Ljubljana, Faculty of Electrical Engineering, Ljubljana, Slovenia

[3]INEA d.o.o., Ljubljana, Slovenia

[4]Computer Technology Institute, Athens, Greece

[5]University of Chemical Technology and Metallurgy – Sofia, Sofia, Bulgaria

[6]Nova Gorica Polytechnic, Nova Gorica, Slovenia

R. King: School of Engineering, University of Patras

Introduction

Industrial control systems are normally based on conventional linear control methods, although the processes are mostly nonlinear by nature. Modern control theory offers many alternatives for achieving more efficient control of nonlinear processes. Advanced control methods are based on more comprehensive and accurate process models [bibBeq91; bibHS97; bibMSJ97]. Surveys of industrial control technology [bibTIA98; bibSeb99; bibSA11] show that there is a sizeable and growing market for advanced controllers; yet relatively few vendors offer useful turn-key products.

Certain advanced control concepts, for example, fuzzy parameter scheduling [bibTHC97; bibBOOB02], multiple-model control [bibGHP02; bibDC03], adaptive control [bibHA00], and nonlinear model-predictive control [bibQB99] offer considerable improvements in the control of nonlinear or time-varying processes. There are, however, several constraints when it comes to applying these methods in industrial applications:

1. Narrow field of application. Real-life problems are diverse, and nonlinear control methods have restrictions regarding their scope of applications. Flexible methods or toolboxes of methods are needed in industry.
2. Availability. Advanced methods are required in the form of ready-to-use building blocks for industrial process control development software environ-

ments. Custom design requires considerable effort, time, and money and is often not favoured.
3. Hardware requirements. Due to the complexity of implementation and the computational demands, advanced methods may require resources not normally available in industrial control hardware.
4. Tuning complexity. Non-specialised field engineers are intimidated by the complexity of tuning and maintenance.
5. Model reliability. With methods heavily dependent on accurate process models, the long-term reliability of nonlinear modelling is often an issue.
6. High profitability demand. In many cases, nonlinear processes can be controlled using the ubiquitous PID controller. In order to replace a conventional control system with an advanced one, a considerable performance increase (financial gain) must be ensured. The maintenance costs of an inadequate conventional control solution may be less obvious.

Our aim was to design an advanced controller named *ASPECT* that addresses some of the aforementioned issues, with the emphasis on reducing tuning complexity, using the concepts of agent-based systems (ABS) [bibWJ95; bibDR09]. The controller configuration procedure is simplified by the partial automation of the commissioning procedure, which is typically carried out by a control engineer. The idea behind ABS is that difficult problems may be solved by assigning tasks to networked *software agents*. These software agents are characterised by properties such as autonomy (i.e., they operate without the direct intervention of humans), social ability (i.e., they interact with other agents), reactivity (i.e., they perceive their environment and respond to it), pro-activeness (i.e., they exhibit goal-directed behaviour, take the initiative), etc. This work does not address the issues of ABS theory, but rather the application of basic ABS concepts to the rather conservative field of process systems engineering, where a number of restrictions have to be considered. For example: initiative is restricted, a high degree of reliability and predictability is demanded, insight into the problem domain is limited to the sensor readings, specific hardware platforms are used where object-oriented programming is not supported, etc.

The outline of this chapter is as follows. Firstly, Section 2 presents an overview of the ASPECT controller's implementation structure. Section 3 describes the Run-Time Module's structure and its most important sub-modules (agents), including some simulation results and an overview of the implementation on a PLC (Programmable Logic Controller). Section 4 provides a brief description of the Configuration Tool. Section 5 describes the experimental results of the application of the controller to a pilot plant where it is used to control the pressure difference on a hydraulic valve in a valve test apparatus. Finally, the lessons learned are discussed and the conclusions are drawn.

ASPECT Controller Concept

The ASPECT controller was designed to be an efficient and user-friendly engineering tool for the implementation of parameter-scheduling nonlinear control in the process industry. User-friendliness was addressed by simplifying the commissioning of the controller using automatic experimentation and tuning. Both the control and self-tuning algorithms were adapted for implementation on the PLC or open-controller industrial hardware platforms.

The key to the concept is the self-tuning mechanism. The controller parameters are automatically tuned from a nonlinear process model. This model is determined on the basis of operating process signals by experimental modelling, where a novel online learning procedure is used. This procedure is based on model identification using the local learning approach [bibMSJ97, p. 188]. Compared to adaptive methods that use recursive identification continuously [bibHA00], a specific approach is used, where the measurement data are processed batch-wise, with additional steps performed before and after the model identification. These additional steps check the validity of the data prior to the self-tuning actions, and thus the reliability of the modelling and self-tuning is improved. Recently, similar approaches have been studied in the framework of evolving systems [bibAF04; bibDS11].

The controller is intended for single-input, single-output processes; a measured disturbance may be included and used for the feed-forward. The application range of the controller depends on the selected control algorithm. The controller has a modular structure that permits the use of a range of control algorithms that are suitable for different processes. The controller also monitors the resulting control performance and reacts to detected irregularities.

The two main components of the ASPECT system are the Run-Time Module (RTM) and the Configuration Tool (CT). The RTM runs on a PLC or an embedded controller, performing all the main functionality of real-time control, online learning and control performance monitoring. The RTM includes a human-machine interface (HMI) in the form of a hierarchical set of dialogue windows on the PLC operator panel, which allows the direct configuration of all the RTM parameters and assists in the execution of the plant experiments. The CT, which is used on a personal computer (PC) only during the initial configuration phase, simplifies the commissioning procedure by providing guidance and default parameter values.

Run-Time Module

RTM Structure

The RTM is made up of a set of interconnected modules, linked in the form of a multi-agent system. Fig. 1 shows an overview of the RTM and its main modules: the Signal Pre-processing Agent (SPA), the Online Learning Agent (OLA), the Model Information Agent (MIA), the Control Algorithm Agent (CAA), the Control Performance Monitor (CPM), and the Operation Supervisor (OS).

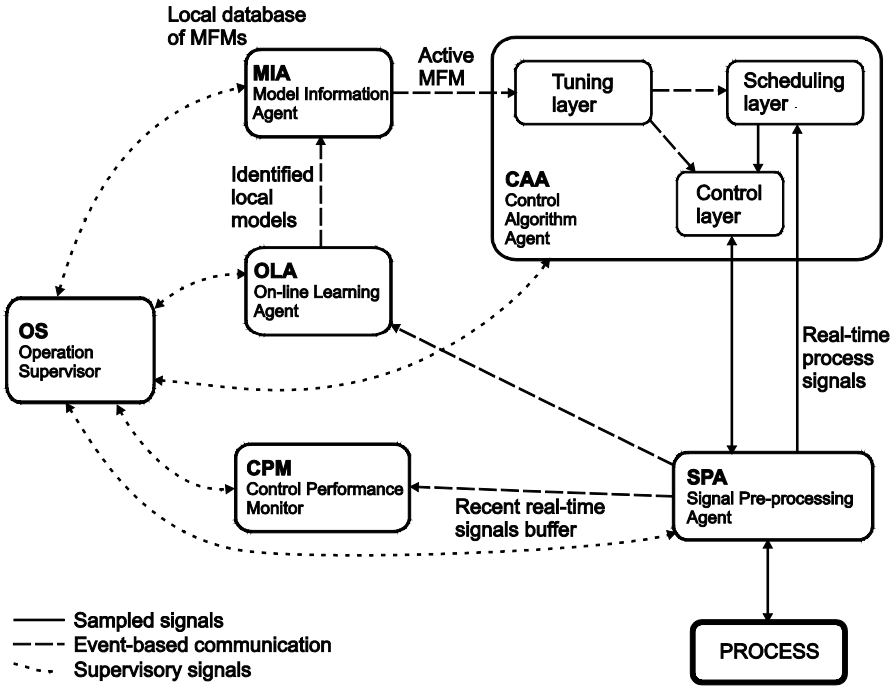


Fig. 1. Run-Time Module Structure

Multi-faceted Model (MFM)

In order to accommodate the diverse model requirements of the OLA and the CAAs, the ASPECT controller is based on the multi-faceted model (MFM) concept. The concept was proposed by Zeigler [bibZei79] and later on elaborated by other authors (e.g., [bibSHL90], [bibFra95]). In general, multi-faceted modelling represents the modelling of one system from different aspects and at different levels of complexity, thus providing a means to solve different (engineering) tasks or one task in different ways. The original idea was to have a multi-faceted model that would represent a database of various models. The appropriate models from this database should be selected according to the type of process, the working point of the process, the specifics of the CAAs, etc. With various types of models the different situations (states) in which the process is likely to be should be covered, thus enabling a high degree of autonomy of the system controlling the process.

During development, the original idea was reduced to a less ambitious MFM that includes a set of local first- and second-order discrete-time linear models with time delay and offset, and also allows a fuzzy model interpretation. Each local model corresponds to one value of the scheduling variable $s(k)$. The model equation for first-order local models is

$$y(k+1) = -a_{1,j}\,y(k) + b_{1,j}\,u(k-du_j) + c_{1,j}\,v(k-dv_j) + r_j \tag{1}$$

and for second-order models it is

$$\begin{aligned} y(k+1) = {} & -a_{1,j}\,y(k) - a_{2,j}\,y(k-1) + b_{1,j}\,u(k-du_j) + b_{2,j}\,u(k-1-du_j) + \\ & + c_{1,j}\,v(k-dv_j) + c_{2,j}\,v(k-1-dv_j) + r_j \end{aligned} \tag{2}$$

where k is the discrete time index, j is the number of the local model, $y(k)$ is the process output signal, $u(k)$ is the process input signal, $v(k)$ is the optional measured disturbance signal (MD), du is the delay in the model branch from u to y, dv is the delay in the model branch from v to y, and $a_{i,j}$, $b_{i,j}$, $c_{i,j}$, and r_j are the parameters of the j-th local model.

For the OLA, the set of local models is interpreted as a Takagi-Sugeno fuzzy model, which in the case of a second-order model can be written in the following form:

$$
\begin{aligned}
y(k+1) = &-\sum_{j=1}^{m} \beta_j(k) a_{1,j} y(k) - \sum_{j=1}^{m} \beta_j(k) a_{2,j} y(k-1) + \\
&+ \sum_{j=1}^{m} \beta_j(k) b_{1,j} u(k-du_j) + \sum_{j=1}^{m} \beta_j(k) b_{2,j} u(k-1-du_j) + \\
&+ \sum_{j=1}^{m} \beta_j(k) c_{1,j} v(k-dv_j) + \sum_{j=1}^{m} \beta_j(k) c_{2,j} v(k-1-dv_j) + \sum_{j=1}^{m} \beta_j(k) r_j
\end{aligned}
\tag{3}
$$

where $\beta_j(k)$ is the fuzzy membership function value of the j-th local model at the current position of the scheduling variable $s(k)$. Normalised triangular membership functions are used, as illustrated in Fig. 2.

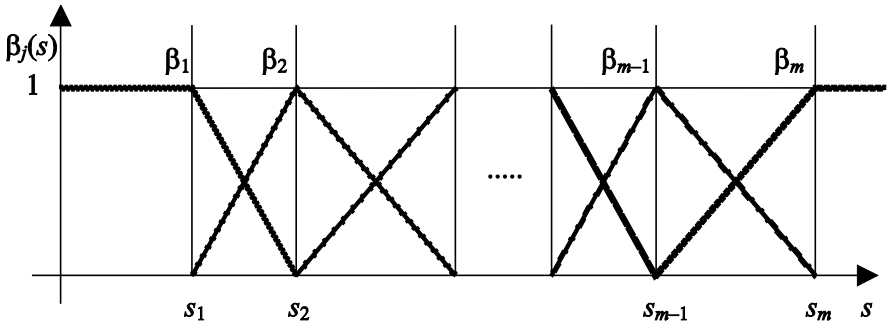


Fig. 2. Fuzzy membership functions of local models

The scheduling variable $s(k)$ is calculated as a weighted sum of the process signals with the coefficients k_r, k_y, k_u, and k_v as follows:

$$
s(k) = k_r r(k) + k_y y(k) + k_u u(k-1) + k_v v(k) \tag{4}
$$

The coefficients are configured by the engineer according to the nature of the nonlinearity of the process.

Note that the applied modelling approach is very similar to the concept of "local model networks" [bibMSJ97] but still preserves the idea of different models for different purposes present in multifaceted modelling. In this sense, the process can be represented by a wide variety of models, starting from one simple, linear, first-order model without dead-time and ending with a Takagi-Sugeno model that includes 10 second-order sub-models with dead-time.

Online Learning Agent (OLA)

The task of the OLA is to estimate the local models from the process signals using an experimental modelling procedure. The OLA scans the buffer of recent real-time signals, prepared by the SPA, and performs a model identification (estimation of the parameters) of those local models that are excited by the signals. Then, the identified model is verified by comparing the simulated model output with the process output measurements. If the new sets of estimated parameters pass the verification test and are found to be better than the existing sets, they are submitted to the MIA.

The OLA is invoked either upon demand from the OS or autonomously, when an interval of the process signals with sufficient excitation is available for processing. Once activated, it processes the signals batch-wise and uses the local learning approach. An advantage of the batch-wise concept is that the decision on whether to adapt the model is not performed in real-time but following a delay that allows for an inspection of the identification result before it is applied. Thus, better means for control over the data selection is provided. Fig. 3 illustrates in more detail the procedure that is executed when the OLA is invoked.

The problem of distributing the computation time required for model identification appears with the batch-wise processing of the data (as opposed to the online recursive processing that is typically used in model-based adaptive controllers). This problem is resolved by using a multi-tasking operation system, so that the OLA is executed as a low-priority task. The OLA typically requires considerably more computation than the real-time control algorithm; however, the computational load is acceptable for microprocessors in modern industrial control equipment, as long as the processing of the batch does not have to be carried out within a single sampling period of the control algorithm.

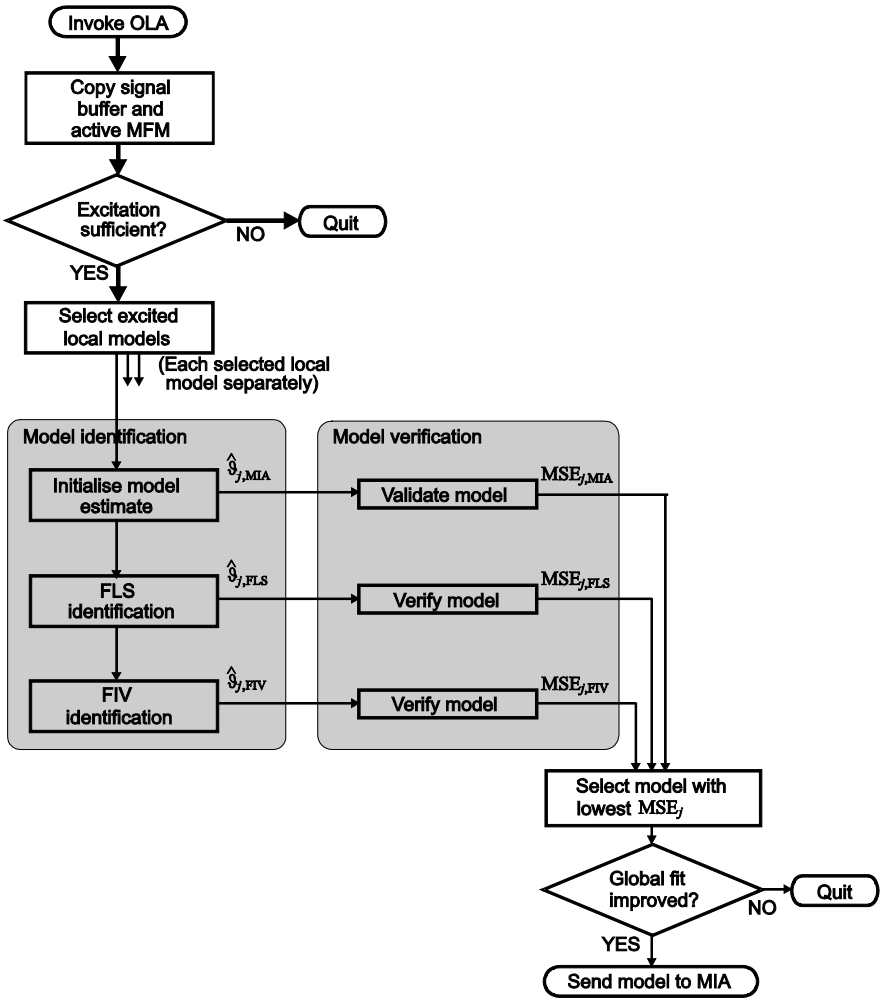


Fig. 3. Online learning procedure

Copy signal buffer and active MFM

When the OLA is invoked, the relevant section of the signal buffer is acquired from the SPA that maintains it. This is required because the signal buffer is being updated during the OLA computations.

The online learning procedure always compares the newly identified local models with the previous set of parameters. Therefore, the active MFM is obtained

from the MIA. A default set of model parameters is used for the local models that have not yet been identified.

Excitation check

A quick excitation check is performed at the start. If the standard deviations of the signals $r(k)$, $y(k)$, $u(k)$, and $v(k)$ in the active buffer indicate excitation above their threshold levels, further processing is initiated, otherwise the OLA terminates.

Select local models

A local model is selected if the sum of its membership functions $\beta_j(k)$ over the active buffer normalised by the active buffer length exceeds a given threshold. Only the selected local models are included in further processing.

Model identification

The local model parameters are estimated using the Fuzzy Instrumental Variables (FIV) identification method developed by Blažič et al. [bibBSGD03, bibBSGD09]. This is an extension of the linear instrumental variables identification procedure [bibLju87] for the Takagi-Sugeno fuzzy model. The method uses the local learning approach [bibMSJ97]. This approach is based on the assumption that the parameters of all the local models will not be estimated in a single regression operation. Compared to the global approach, it is less prone to the problems of ill-conditioning and local minima, and involves a less complex computation. The FIV identification method is well suited to the needs of industrial operation (intuitiveness, gradual building of the nonlinear model, modest computational demands). It allows an inventory of the local models that are not estimated properly due to insufficient excitation. It is efficient and reliable in the early stages of controller configuration, when all the local models have not been estimated yet. On the other hand, the convergence in the vicinity of the optimum is slow. Therefore, it is likely to yield a worse model fit than methods employing nonlinear optimisation using the global approach. An alternative approach based on recursive clustering and recursive least-squares has been investigated recently [bibDS11].

The main procedure of the FIV method is outlined as follows. Model identification is performed for each selected local model (denoted by the index j) separately. $\hat{\boldsymbol{\theta}}_{j,\mathrm{MIA}}$, the initial estimate of the parameter vector, is copied from the active MFM, and the covariance matrix $\mathbf{P}_{j,\mathrm{MIA}}$ is initialised to $10^5 \cdot \mathbf{I}$ (identity matrix). In

the first step, the FLS (fuzzy least-squares) estimates, $\hat{\boldsymbol{\theta}}_{j,\mathrm{FLS}}$ and $\mathbf{P}_{j,\mathrm{FLS}}$, are obtained using weighted least-squares identification, with $\beta_j(k)$ used for the weighting. The calculation is performed recursively to avoid matrix inversion:

$$\begin{aligned} e(k+1) &= D_Z\left(\beta_j y(k+1) - \boldsymbol{\psi}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{\theta}}_j(k)\right) \\ \mathbf{P}_j(k+1) &= \mathbf{P}_j(k) - \frac{\mathbf{P}_j(k)\boldsymbol{\psi}_j(k+1)\boldsymbol{\psi}_j^{\mathrm{T}}(k+1)\mathbf{P}_j(k)}{1+\boldsymbol{\psi}_j^{\mathrm{T}}(k+1)\mathbf{P}_j(k)\boldsymbol{\psi}_j(k+1)} \\ \hat{\boldsymbol{\theta}}_j(k+1) &= \hat{\boldsymbol{\theta}}_j(k) + \mathbf{P}_j(k+1)\boldsymbol{\psi}_j(k+1)e(k+1) \end{aligned} \tag{5}$$

where $\boldsymbol{\psi}_j(k+1) = \beta_j\left[-y(k), -y(k-1), u(k-du_j), u(k-1-du_j), v(k-dv_j)\right]^{\mathrm{T}}$ is the vector of the measurements, and $D_Z(.)$ is the dead-zone operator with the parameter x_{dead}

$$D_Z(x) = \begin{cases} 0, & |x| \le x_{\mathrm{dead}} \\ x, & |x| > x_{\mathrm{dead}} \end{cases} \tag{6}$$

In the second step, the FIV (fuzzy instrumental variables) estimates, $\hat{\boldsymbol{\theta}}_{j,\mathrm{FIV}}$ and $\mathbf{P}_{j,\mathrm{FIV}}$, are calculated using weighted instrumental variables identification as follows:

$$\begin{aligned} e(k+1) &= D_Z\left(\beta_j y(k+1) - \boldsymbol{\psi}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{\theta}}_j(k)\right) \\ \mathbf{P}_j(k+1) &= \mathbf{P}_j(k) - \frac{\mathbf{P}_j(k)\boldsymbol{\chi}_j(k+1)\boldsymbol{\psi}_j^{\mathrm{T}}(k+1)\mathbf{P}_j(k)}{1+\boldsymbol{\psi}_j^{\mathrm{T}}(k+1)\mathbf{P}_j(k)\boldsymbol{\psi}_j(k+1)} \\ \hat{\boldsymbol{\theta}}_j(k+1) &= \hat{\boldsymbol{\theta}}_j(k) + \mathbf{P}_j(k+1)\boldsymbol{\chi}_j(k+1)e(k+1) \end{aligned} \tag{7}$$

where $\boldsymbol{\chi}_j(k+1) = \beta_j\left[-\hat{y}(k), -\hat{y}(k-1), u(k-du_j), u(k-1-du_j), v(k-dv_j)\right]^{\mathrm{T}}$ is the instrumental variables vector, and $\hat{y}(k) = \sum_{j=1}^{m}\boldsymbol{\psi}_j^{\mathrm{T}}(k)\hat{\boldsymbol{\theta}}_j(k-1)$ is the simulated output. The dead zone is used in the FLS and FIV recursive estimation in order to prevent result degradation as a result of noise. The vector of the parameters and the covariance matrix are updated only if the absolute weighted difference between the process output and its prediction is above the configured noise threshold.

In the case of a lack of excitation in the model branch from *u* to *y* or in the model branch from *v* to *y* (or when the measured disturbance is not present at all), simplified variants of the method with reduced parameter estimate vectors are used.

Model verification/validation

This step involves local and global verification/validation, where the recent batch of measurements is compared with the model simulations. The same procedure is referred to as "verification" with the newly estimated parameter sets $\hat{\boldsymbol{\theta}}_{j,\mathrm{FLS}}$ and $\hat{\boldsymbol{\theta}}_{j,\mathrm{FIV}}$ (over the same data set), or as "cross-validation" with the initial parameter set $\hat{\boldsymbol{\theta}}_{j,\mathrm{MIA}}$.

Local verification is performed by comparing the simulation output $\hat{y}$ of each selected local model with the actual process output, recorded in the signal buffer, in the proximity of the selected local model position. The normalised sum of mean square errors $\mathrm{MSE}_j = \frac{1}{N}\sum_{k=0}^{N-1}\left(\hat{y}(k) - y(k)\right)$ is calculated. The proximity to the selected local model position is defined by the membership functions β_j. For each of the selected local models, this step is carried out with three sets of model parameters: $\hat{\boldsymbol{\theta}}_{j,\mathrm{MIA}}$, $\hat{\boldsymbol{\theta}}_{j,\mathrm{FLS}}$, and $\hat{\boldsymbol{\theta}}_{j,\mathrm{FIV}}$. From among $\hat{\boldsymbol{\theta}}_{j,\mathrm{MIA}}$ and $\hat{\boldsymbol{\theta}}_{j,\mathrm{FLS}}$, the set with the lower MSE_j is selected.

Global verification is performed by comparing the simulation output of the fuzzy model, including the selected set with the actual process output. The normalised sum of mean square errors (MSE_G) is calculated. If the global verification result is sufficiently improved compared to the initial fuzzy model, the selected set is sent to the MIA as a result of the online learning, otherwise the original set $\hat{\boldsymbol{\theta}}_{j,\mathrm{MIA}}$ remains in use.

For each processed local model, the MIA receives the MSE_j, which serves as a confidence index, and a flag indicating whether the model is new or not. Even if no new model is obtained, this may serve for model validity checking.

Model structure estimation

Two model structure estimation units are also included in the OLA. The Dead-Time Unit (DTU) estimates the process time delay by comparing the estimation results with different dead-time values. The Membership Function Unit (MFU) suggests whether a new local model should be inserted. It estimates an additional local model in the middle of the interval between the two neighbouring local models that are the most excited. The model is submitted to the MIA if the global validation of the resulting fuzzy model is sufficiently improved, compared to the original fuzzy model. The operation of both structure estimation units is only reliable when suitable excitation is present in the process signals, and therefore higher excitation thresholds apply.

Model Information Agent (MIA)

The MIA maintains the active MFM and its status information.

Its primary function is to process the online learning results. When the OLA sends a new local model, it is accepted if it passes the stability test and its confidence index is sufficient. If it is accepted, a "ready for tuning" notification is sent to the CAA. Each local model contains a flag indicating whether its parameters have been estimated since start-up or not, facilitating a quick overview of the progress of self-tuning, and a model confidence index. If the confidence index is below the threshold, the Automatic Mode may be disabled.

The MIA contains a mechanism for adding additional local models (at new positions of the scheduling variable) into the MFM. This may occur either by external request or automatically, using the MFU of the OLA. The MIA may also store the active MFM to a local database or recall a previously stored one, which is useful if the process switches between different operating modes.

The process model in the MIA is built gradually. During the initial configuration, the MIA is filled with default local models based on the initial estimation of the process dynamics. They are not exact but may provide reliable (although sluggish) control performance, similar to the Safe Mode. Using online learning through experiments or normal process operation (when the conditions are appropriate for closed-loop identification), an accurate model of the process is estimated by receiving identified local models from the OLA.

Control Algorithm Agent (CAA)

A CAA is composed of a non-linear control algorithm and a procedure for automatic tuning of its parameters. Several different CAAs may be used in the controller and may be interchanged in the initial configuration phase.

The controller may operate in the following modes:

- *Manual Mode*: open-loop operation (actuator constraints are enforced)
- *Safe Mode*: a fixed PI controller with conservatively tuned parameters
- *Auto Mode* (or several auto modes with different tuning parameters): a non-linear controller

The CAAs share a common interface of interaction with the OS and a common modular internal structure, consisting of three layers:

1. The *control layer* contains the functionality of a local linear controller (or several local linear controllers simultaneously), including everything required for reliable operation in industry, such as the handling of constraints with anti-windup protection, bump-less mode, parameter switching, etc.

2. The *scheduling layer* performs real-time blending (switching or scheduling) of the tuned local linear controllers, so that in conjunction with the control layer, a fixed-parameter, non-linear controller is formed.
3. The *tuning layer* is the automatic tuning procedure of the controller parameters from the MFM when the MIA reports that a new local model is generated, if auto-tuning is enabled. The replacement of the parameters of the control and scheduling layers must be carried out in such a manner that real-time control is not disturbed.

Three CAAs have been developed and each has proved effective in specific applications: the Fuzzy Parameter-Scheduling Controller (FPSC), the Dead-Time Compensation Controller (DTCC), and the Rule-Based Neural Controller (RBSC). In the following subsections, an overview of the three CAAs using the above-mentioned layers is presented.

Fuzzy Parameter-Scheduling Controller

A graphical overview of the FPSC is shown in Fig. 4.

The control layer includes a single PID controller in a form suitable for controller blending using velocity-based linearization. It is equipped with anti-windup protection and bump-less transfer during mode changes.

The scheduling layer performs fuzzy blending of the PID controller parameters according to the scheduling variable $s(k)$ and the fuzzy membership functions $\beta_j(k)$ of the local models. The concept of velocity-based linearization enables the dynamics of the blended global controller to be a linear combination of the local controller dynamics across the entire operating region, not just around the equilibrium operating points. This provides the potential to improve the performance with a few local models and more transparent behaviour at the off-equilibrium operating points [bibLL98; bibKZSV02]. In order to facilitate the velocity-based linearization approach, the *Kp* and *Td* parameters are blended directly, while in the case of *Ti*, its inverse value is blended.

The tuning layer is based on the magnitude optimum (MO) criterion implemented using the multiple integration (MI) method [bibVSJ01]. By applying the MO criterion, the magnitude (amplitude) of the system's closed-loop, set-point response is made as flat and close to unity as possible for a large bandwidth [bibWhi46]. This results in a relatively fast and non-oscillatory response of the closed-loop system. The expressions for calculating the PID controller's parameters using the MO criterion are quite complex; however, the MI method significantly simplifies the equations and enables the calculation of the PID controller's parameters directly from the open-loop response of the process.

Some additional steps are required for using MOMI tuning in the ASPECT controller. At the start of the auto-tuning procedure, a discrete-time local model is received from the MIA. This model is converted into a continuous-time form.

Then, the so-called areas are calculated using the MI method. Finally, the PI and PID controllers' parameters are calculated from the areas. Thanks to the transparent concept of the FPSC, an experienced engineer may choose to configure the control algorithm manually by specifying the local PID controller's positions and parameters directly, without using the model-based tuning procedure.

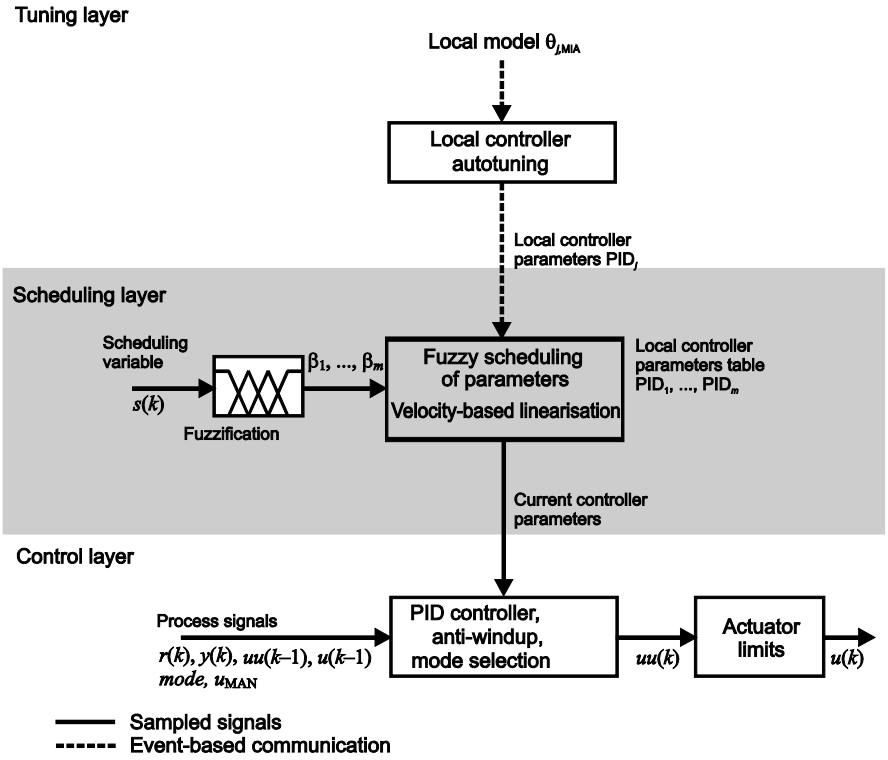


Fig. 4. FPSC overview

Dead-Time Compensation Controller

The DTCC is a nonlinear scheduling control algorithm based on Predictive Functional Control, a relatively simple predictive control method that uses an independent internal model, polynomial control signal parameterisation, and performance criterion reduced to a few coincidence points [bibRic93; bibMac02]; the algorithm is also closely related to the Smith predictor [bibSmi59].

The control layer of the DTCC is composed of a set of local linear PFC controllers, one for each local model. The local PFC controller implementation supports first- or second-order local models with time delay and feedforward compensation

for the measured disturbance. The storage of the signal buffers for the controller input, output, and four internal signals is required by the algorithm. Fuzzy blending of the local controllers (the scheduling layer) is performed at the controller outputs. A schematic diagram of the DTCC structure is displayed in Fig. 5.

The auto-tuning algorithm of the DTCC's tuning layer starts by the conversion of the local models (1)-(2) to continuous time; with second-order models (2), the factorisation to two serial first-order transfer functions is applied. Then it computes the tuning parameters of the local controllers, which are the desired settling time

$$T_{\mathrm{DST},j} = 0.6231 T_{du,j} + 0.0914 T_{P,j} \tag{8}$$

and the coincidence-point location (horizon)

$$H_j = \mathrm{ceil}(1.1155 \frac{T_{P,j}}{T_{\mathrm{samp}}}) \tag{9}$$

where T_{samp} is the sampling time, T_{du} is the time delay in the model branch from u to y, and T_P is the time constant (or the sum of time constants) in the model branch from u to y. Notice that T_{DST} is a more user-friendly alternative to the exponential filter coefficient for the reference trajectory λ, which is calculated as follows:

$$\lambda = \mathrm{e}^{-\frac{3T_{\mathrm{samp}}}{T_{\mathrm{DST}}}} \tag{10}$$

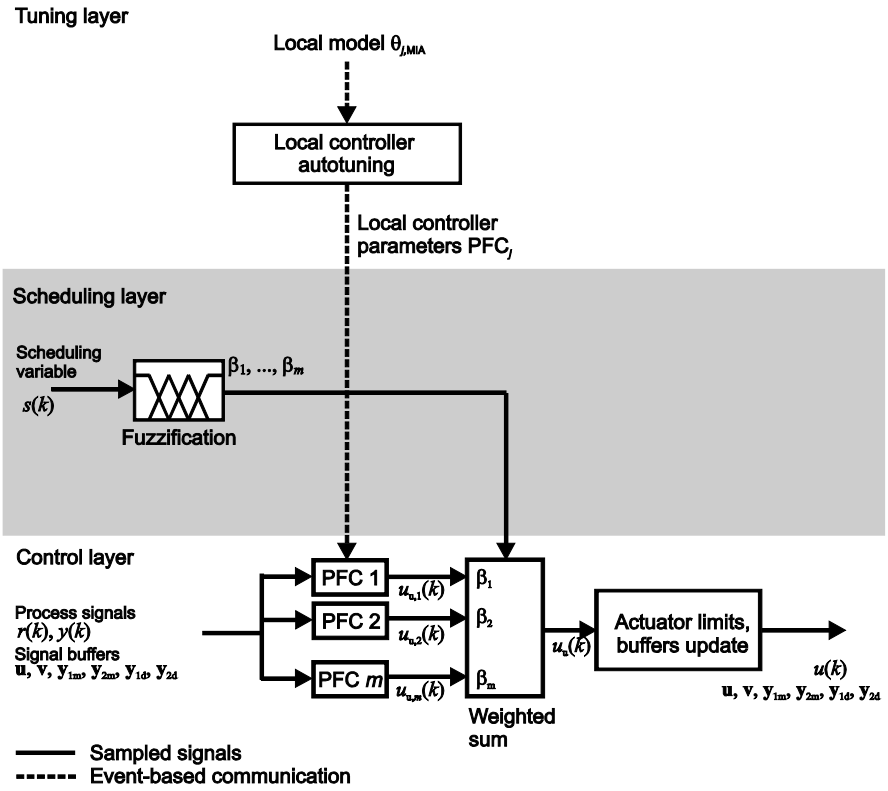


Fig. 5. DTCC overview

Rule-Based Switching Controller

The RBSC is based on safe-switching theory, a relatively recent approach to controlling a large class of nonlinear processes whose behaviour varies considerably over its operating region [bibMor95; bibABLM01; bibKKS02; bibKKS07]. The design objectives of the RBSC are to attain optimum performance in the neighbourhood of any anticipated equilibrium operating point and generation of the best strategy that will ensure safe transition from any operating point to another without jeopardizing the stability of the closed-loop system.

A very practical simplification of the general safe-switching theory used in the RBSC is to make *all* the candidate local controllers for the different operating points have the *same* architecture, in which case only the set of *parameters* of the controller needs be changed. It is paramount that safe and bump-less transfer be

ensured following every controller parameter change so that the plant is not subjected to switching transients that can lead to disruption in the operation of the plant. Specifically, the industrial-standard PID structure of the incremental form is used in the control layer, and a look-up table of local controller parameters is stored in the switching layer of the RBSC. A schematic diagram of the RBSC structure is displayed in Fig. 6.

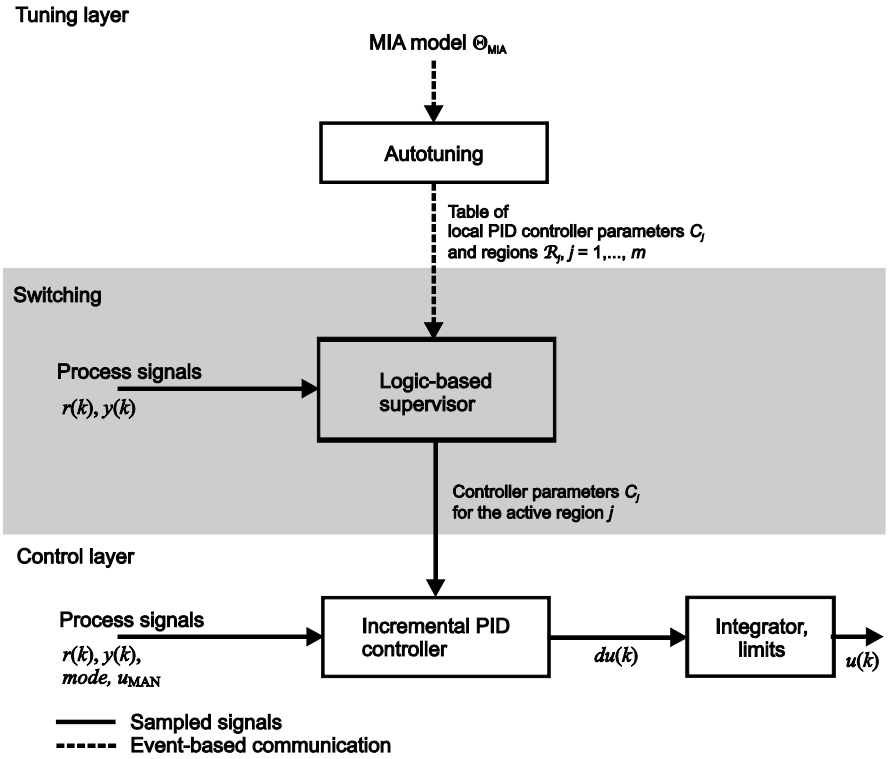


Fig. 6. RBSC overview

For a brief illustration of the safe-switching strategy, we consider a simplified case with a second-order local linearized discrete-time model about every equilibrium operating point (*approximant* $\hat{\boldsymbol{\theta}}_j$)

$$(1+a_1q^{-1}+a_2q^{-2})y(k)=(b_1+b_2q^{-1})u(k) \tag{11}$$

where the coefficients are real and $0 < b_1 < b_2$, and q^{-1} is the delay operator, and a PI controller that allows the discrete-time representation

$$(1-q^{-1})u(k)=(c_0+c_1q^{-1})(r(k)-y(k)) \tag{12}$$

The stability of the closed loop can now be examined from the closed system polynomial

$$\begin{aligned} A_c(q^{-1}) &= 1 + a_{c1}q^{-1} + a_{c2}q^{-2} + a_{c3}q^{-3} = \\ &= (1 + a_1 q^{-1} + a_2 q^{-2})(1 - q^{-1}) + (b_1 + b_2 q^{-1})(c_1 + c_2 q^{-1}) \end{aligned} \tag{13}$$

Given the coefficients of the plant approximant, the closed system is *asymptotically* stable if the following inequalities are satisfied:
a) $b_1 \neq 0$, $b_2 \neq 0$

$$|c_0| \leq \frac{3 + |a_1 + a_2 - 1| + |a_2|}{|b_1|} \quad \text{and} \quad |c_1| \leq \frac{1 + |a_2|}{|b_2|} \tag{14}$$

b) $b_1 = 0$, $b_2 \neq 0$

$$|c_0| \leq \frac{5 + 4a_2^2 + 2|a_1 a_2| + 7|a_2| + 2|a_1|}{|b_2|} \quad \text{and} \quad |c_1| \leq \frac{1 + |a_2|}{|b_2|} \tag{15}$$

c) $b_1 \neq 0$, $b_2 = 0$

$$|c_0| \leq \frac{2 + |a_1 + a_2 - 1|}{|b_1|} \quad \text{and} \quad |c_1| \leq \frac{3 + 2a_2^2 + 2|a_1 a_2| + 5|a_2| + |a_1|}{|b_1|} \tag{16}$$

The proposed rule-based switching algorithm requires information on the current operating conditions and the intended target set-point of the process. The iterative design procedure consists of the following rules:
Step 1: Assume that at a countable set of nominal equilibrium points Y_1, Y_2,... Y_m the parameters of the corresponding plant approximants $\hat{\boldsymbol{\theta}}_1$, $\hat{\boldsymbol{\theta}}_2$,... $\hat{\boldsymbol{\theta}}_m$ have been identified and are available.
Step 2: Satisfy the design requirements for all process approximants. This implies that the closed system is optimum in every sector and stable during transition.
Step 3: Consider the controller C_j applied to the plant model $\hat{\boldsymbol{\theta}}_j$. Determine the regions $\mathcal{R}_j$, $j = 1, ..., m$ so that the controllers satisfy the requirements set in Step 2.
Step 4: If the intersection $\mathcal{R}_j \cap \mathcal{R}_{j+1} \neq \varnothing$, go to Step 5, otherwise go to Step 6.
Step 5 (Switch): For $Y \in [Y_j, Y_{j+1}]$ use a controller $C_j \in \mathcal{R}_j \cap \mathcal{R}_{j+1}$. Repeat until $j - 1 = m$.
Step 6: Select one of the following and repeat the algorithm:
a) Relax the requirements in Step 2.

b) Identify the process at points lying between existing operating points, thereby increasing the available knowledge regarding the process.

As long as the output value is close to a nominal operating point at which an approximant has been identified, the candidate controller in region $\mathcal{R}_j$ will result in optimum performance. Depending on whether the target set-point is set higher or lower than the current operating point, the transition strategy can be simply stated as follows:

1. If a new target set-point lies in the same region as the current one, then do nothing.
2. If a new target set-point is in a region *above* the current one, then switch to the target region controller.
3. If the new target set-point is in a region *below* the current one, then remain with the current controller.
4. If quiescence is reached in any target region, then revert to the target region controller.

The foregoing logic-based switching strategy is suitable for processes whose steady-state input-output relationship $y_{ss} = f(u_{ss})$ is increscent, i.e., $f'(u_{ss}) > 0$, in which case the stability regions satisfy the condition $\mathcal{R}_{j-1} \subset \mathcal{R}_j \subset \mathcal{R}_{j+1}$. The switching strategy can be generalized. For illustration, a diagram of the regions of stability and the optimum controller parameters of a pH controller with three steady-state operating points (pH 6, 7, and 8) is displayed in Fig. 7.

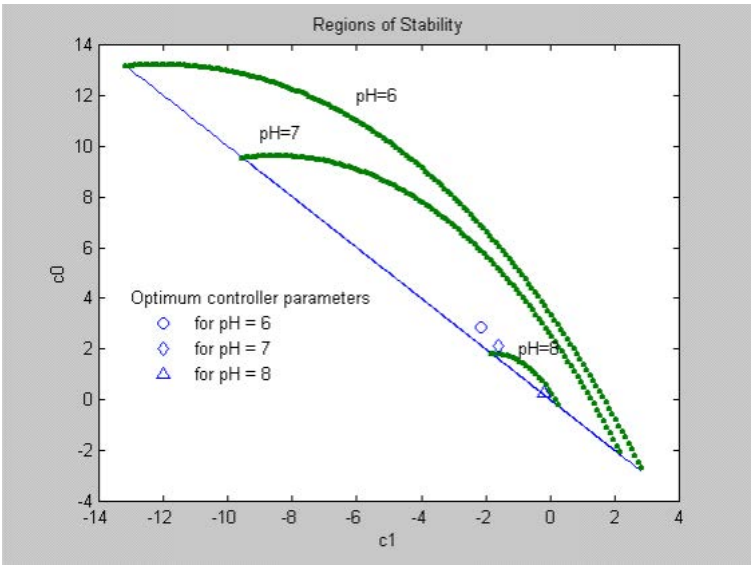


Fig. 7. Regions of stability for the three closed subsystems and the optimum controller parameters for each case

Given the coefficients of each approximant $\hat{\boldsymbol{\theta}}_j$, it is necessary to determine the optimum controller parameter set (c_0, c_1) with respect to a certain performance criterion. Here, the ITAE criterion was used because it leads to a compromise between rise time and overshoot. An initial estimate of the set of local controller parameters by using the Åström and Persson technique [bibAH95] is used as a starting point for the stochastic search using the Metropolis-Hastings simulated annealing algorithm. The search region is determined by the chord defined by the stability bounds. However, a rectangular search space whose boundaries are defined by the algebraic criterion was found to reduce the computational effort significantly. To increase computational effort further, a modified Jury stability test was carried out for each trial parameter set, immediately discarding over 90% of the candidate sets of parameters.

Control Performance Monitor (CPM)

The CPM is a software agent that assesses the behaviour of the control loop. It consists of three modules: the Buffer Pre-processor (BP), the Situation Classifier (SC), and the Performance Estimator (PE). Just like the OLA, it is invoked autonomously or upon demand from the OS and runs as a low-priority task. A block diagram of the CPM's operation is shown in Fig. 8.

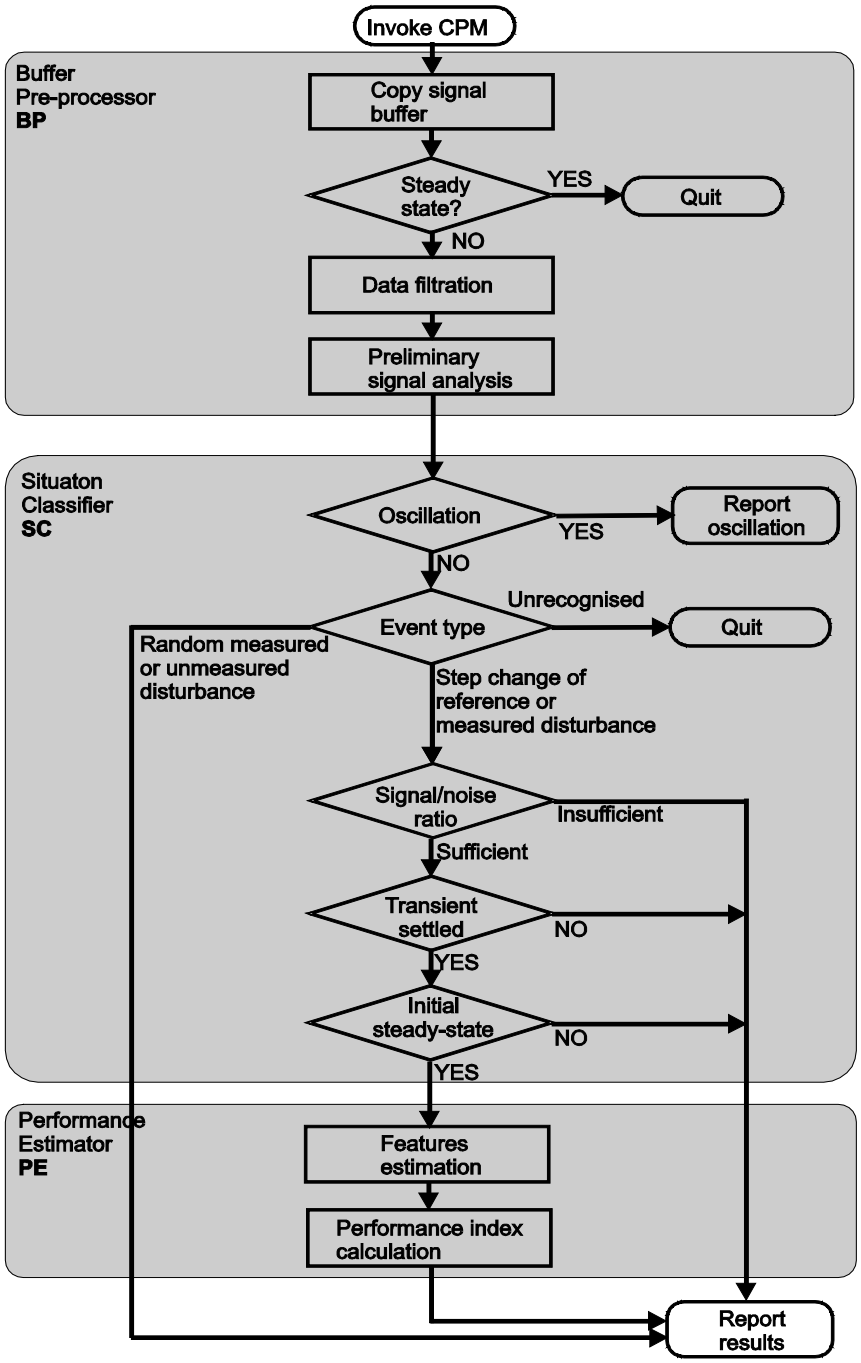


Fig. 8. CPM overview

When the CPM is invoked, the BP scans the buffer of the recent real-time signals, which is maintained by the SPA. It starts by making a copy of the buffer. Then, it checks if the process is in a steady state; if there is no external excitation and the standard deviations of the signals are within the limits, it terminates the processing. Otherwise, it filters the signals and performs a low-level analysis.

The SC searches the pre-processed buffer for the last recognisable event that may be evaluated or is otherwise important. Firstly, if oscillation is detected, a warning is issued to the OS and the processing is aborted. Then, the event classification takes place, where a step change in the reference signal or the measured disturbance signal, or an unmeasured disturbance transient may be identified. In the case of an unrecognised event, the processing is terminated. The final task of the SC is to assess whether the event is eligible for an estimation of features by the PE. Unmeasured disturbance transients do not qualify because their causes are generally not known. In the case of step changes, the SC verifies whether the signal/noise ratio is sufficient, whether the transient after the event has settled, and whether there was a steady-state period before the event.

When an eligible event is detected, the PE estimates the appropriate features, depending on the event type. The following features may be estimated: overshoot, settling time, rise time, oscillation decay rate, and tracking-error measure or regulation-error measure. Using a fuzzy evaluation procedure, an overall performance index (PI) is also calculated from the features.

The CPM results are sent to the human-machine interface for display and to the OS for further automatic actions. If poor performance is detected, an automatic switchover to the Safe Mode may be triggered. Other automatic actions include, for example, blocking the OLA if an oscillation is detected (oscillatory signals may appear to contain rich excitation; however, their frequency spectrum is not suitable for model identification by the OLA). Generally, the OS does not perform a "direct" adaptation of the CAA parameters based on the CPM results because the primary concept of the ASPECT controller is "indirect" model-based, self-tuning and the role of the CPM is supervisory; however, direct adaptation could also be implemented for specific applications.

Operation Supervisor (OS)

The OS coordinates the control, modelling, and tuning activities of the agents and user interaction through the hierarchical set of dialogue windows of the human-machine interface (HMI). The OS and the HMI include the functionality required for automatic, user-friendly experimentation, which is usually required for controller commissioning. The controller-commissioning procedure includes the phases of the basic settings, an approximate estimation of the process dynamics for safe controller tuning, nonlinear modelling and tuning of the CAA, and configuration of the regime for regular operation.

Typically, the commissioning of the controller is performed by conducting a set of experiments that are intended to explore the process dynamics over the whole operating range of the process. The OS supports the control engineer by automatically executing the experiments for the identification of local models. These experiments consist of a series of step changes about the operating point of the model, in either an open or closed loop. In addition, the OS coordinates the OLA, the MIA, and the CAA to automatically process the signals, build the model, and tune the local controllers. This is the fastest and most reliable way to tune the controller when experimentation with the plant is allowed. The automatic conducting of experiments for a closed-loop performance evaluation using the CPM is also supported.

Alternatively, if experimentation is not allowed, it may be possible to perform controller commissioning without the scheduling of experiments by gathering model information during normal plant operation. In this case, the controller is initialised in the Safe Mode, and processing of the signals for modelling and tuning is triggered by the OLA autonomously. However, it is required that sufficient excitation over the whole operating region is available during regular operation. The progress of the modelling is indicated by the status flags of the local models in the MIA. The flags show which models have been tuned and their respective confidence indices.

A range of operating regimes may be configured by enabling or disabling the agents and changing their configuration parameters. This results in a flexible control system that covers the requirements of a wide range of applications, and may help diagnose problems. Thus, although designed for the control of nonlinear processes, the ASPECT controller may also be used for adaptive control using a single linear model or as a tool for PID controller tuning. Some specific operating-regime options include the following:

- The OLA and/or the CPM may be invoked autonomously (during regular operation) or upon OS demand (following scheduled experiments), or both.
- The OLA may estimate the process dead-time continuously or not.
- The OLA may attempt to insert additional local models when appropriate, or estimate the local models at the fixed pre-selected positions only.
- Controller retuning may be triggered automatically immediately after each change of the model in MIA ("adaptive" operation), or following the confirmation by the engineer ("self-tuning" operation).
- The OLA may also be used for monitoring the process dynamics by cross-validation of the model, without the intention of controller tuning.

While the initiative and suggestions of the agents are helpful during system configuration, this may not be desirable during regular operation. Therefore, at the end of the commissioning procedure, the system may be reconfigured to deactivate self-tuning activity.

Simulation results

In order to test the concept of the control system and the performance of particular algorithms, the prototype of the RTM was first developed in the MATLAB/SIMULINK environment.

In the initial simulation tests [bibBSGD03], the performance of the controller was evaluated on the pH control benchmark of Henson and Seborg [bibHS94], shown in Fig. 9. In the benchmark process, an acid stream Q_1, a buffer stream Q_2, and a base stream Q_3 are mixed in a tank T1. The pH of the mixture is measured with a sensor located downstream. The effluent pH is the controlled variable y, and the manipulated variable u is the flow of the base stream Q_3. The static characteristic of the process is highly nonlinear and its open-loop gain changes by a factor of 8, and therefore it is very difficult to control with a conventional PID controller.

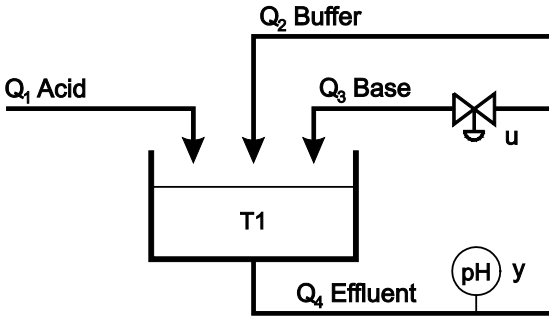


Fig. 9. A simulated neutralisation benchmark process

The nonlinear model of the benchmark simulated by using Matlab/Simulink was used in the role of the real process, while the controller used a model in the form of the MFM obtained by online learning. The operating range between pH values 6 and 8 was covered with five local models placed at positions 6, 7, 7.15, 7.4, and 8. These positions were determined based on the known shape of the titration curve. $s(k) = 0.3\, y(k) + 0.7\, r(k)$ was used as the scheduling variable. Each local model was trained with online learning using an open-loop experiment consisting of three step-changes of u of small amplitude about the operating point, and the local controllers were tuned automatically from the local models. Fig. 10 shows the performance comparison between PI control in the Safe Mode (top), FPSC control, DTCC control, and RBSC control (bottom). The process output follows the reference signal reasonably well under FGSC control, while the performance of a fixed PI controller is sluggish in some areas. The PI controller is tuned

for stable performance in the high-gain operating area around pH 8; however, its operation in the low-gain region below pH 7 is sluggish.

The presented tests as well as some other tests performed on the pH model demonstrated that the proposed concept of the system for advanced control is viable and that further work towards implementation on PLC platforms is reasonable.

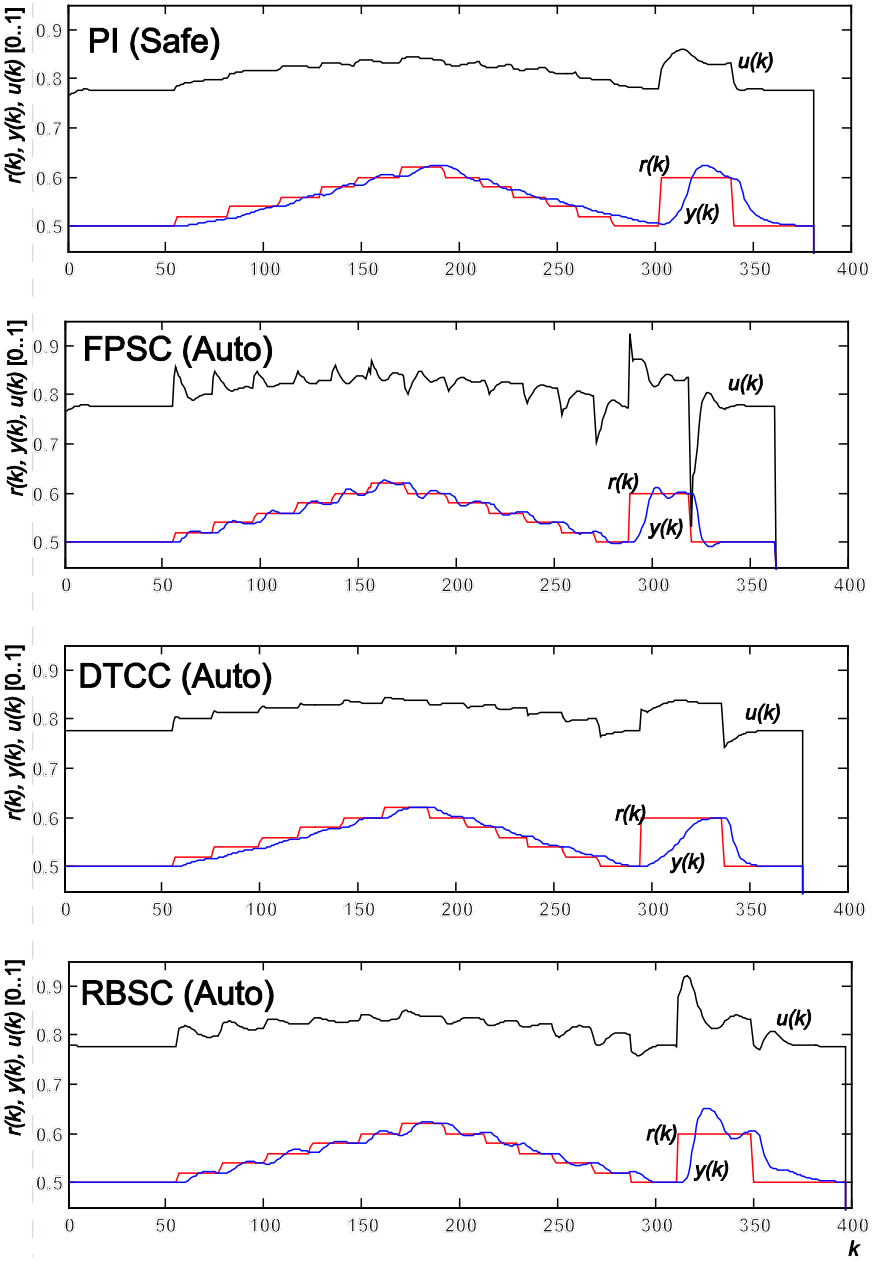
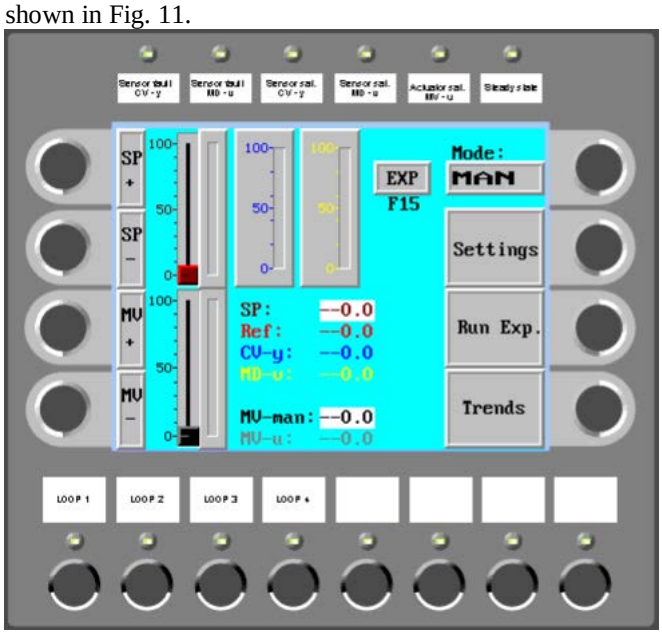


Fig. 10. Simulated pH control performance comparison:
PI (top), FPSC, DTCC, and RBSC control (bottom)

PLC Implementation

Despite the careful selection and modification of the algorithms to reduce the computational demand, the OLA and CPM modules are not suitable for implementation in typical PLCs. A DSP or an open controller add-on module tends to be a more cost-effective solution than an upper-market PLC. The platform for running the RTM of the ASPECT controller in the pilot application presented in this paper consists of a Mitsubishi A1S series PLC with an INEA IDR SPAC20 coprocessor, based on the Texas Instruments DSP TMS320C32 at 40 MHz with 2MB of RAM, and a Mitsubishi MAC E700 HMI unit.

The RTM is implemented as an extension for INEA IDR BLOK[1] v. 4.22, a graphical development tool for closed-loop control applications in the process industry using Mitsubishi Electric MELSEC AnSH PLC controllers [bibIne01]. The FPSC algorithm is included as an additional controller block "PID/FPSC". Other RTM components are implemented as separate PLC tasks, coded in C and downloaded to the PLC in the compiled form. They are supervised using the HMI unit through a hierarchical set of menus (such as: Operator Display, Trends Display, Settings Overview, Experiment Parameters, Online Learning Settings, Model Parameters, Model Status, FPSC Settings, FPSC Parameters, etc.). Two sample HMI menus are shown in Fig. 11.



[1] The concept of IDR BLOK is closely related to the more recent "Function Block Diagram" of the IEC 61131-3 standard. An IEC 61131-3 compliant version of IDR BLOK has been developed recently.

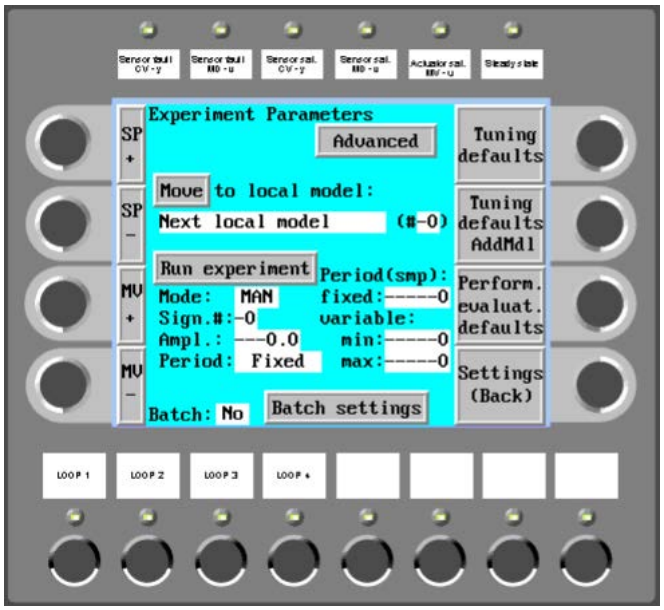


Fig. 11. Sample HMI menus: Operator Display (top), Experiment Parameters (bottom)

Configuration Tool

The CT is intended to assist the control-system designer in the commissioning of the nonlinear controller during the initial configuration phase. It simplifies the commissioning procedure by providing guidance and default parameter values. It runs on a PC which is connected to the PLC running the RTM, and takes advantage of the better graphical-user-interface capabilities of the PC platform.

Two implementations of the CT were developed:

– The original CT facilitates self-configuration of the RTM, closely following the above-described project concept, and mostly relies on the functionality of the RTM.
– The alternative CT is an extension of the single-loop PID tuning tool "Lek Tuner" [bibVH07] for the tuning of non-adaptive, stand-alone FPSC controllers for applications on less-capable PLC platforms that cannot host the whole RTM.

Original CT

The original CT contains a configuration "wizard" that guides the engineer through the typical scheduling-controller commissioning procedure. It was designed for plant engineers who may not have an in-depth knowledge of nonlinear modelling and control. The procedure is broken down into small steps (25 dialogue windows). In each step, instructions are displayed and default values are suggested by using rules of thumb, based on the information already available. Inconsistency warnings may be displayed. Notice that an experienced engineer may conduct a similar procedure directly with the RTM via the HMI.

The main phases of the commissioning procedure are:

A. *Basic settings*: the selection of the control signals, the signal constraints, the sampling time, the CAA, the scheduling variable, and the model order.
B. *Safe-mode configuration*: the estimation of the process dynamics, where experimentation and identification using the RTM may be used; self-tuning of the "safe" controller parameters; optional performance verification.
C. *MFM initialisation:* initialisation of the local model positions and their parameters; display of the local model parameters and step responses.
D. *CAA settings:* the initialisation of the default values and the advanced autotuning parameters.
E. *OLA settings:* the initialisation of the default values and the advanced OLA settings.
F. *CPM settings:* the initialisation of the default values and the advanced CPM settings.
G. *Experiment settings:* the initialisation of the default experimentation parameters and advanced automatic experimentation settings.
H. *Local controller tuning:* conducts the sequence of automatic (open- or closed-loop) experimentation, online learning, and tuning using the RTM around each local model position.
I. *Performance verification:* conducts the sequence of automatic experimentation and performance evaluation using the RTM around each local model position.

Sample dialogue windows for OLA settings (stage E) are shown in Fig. 12.

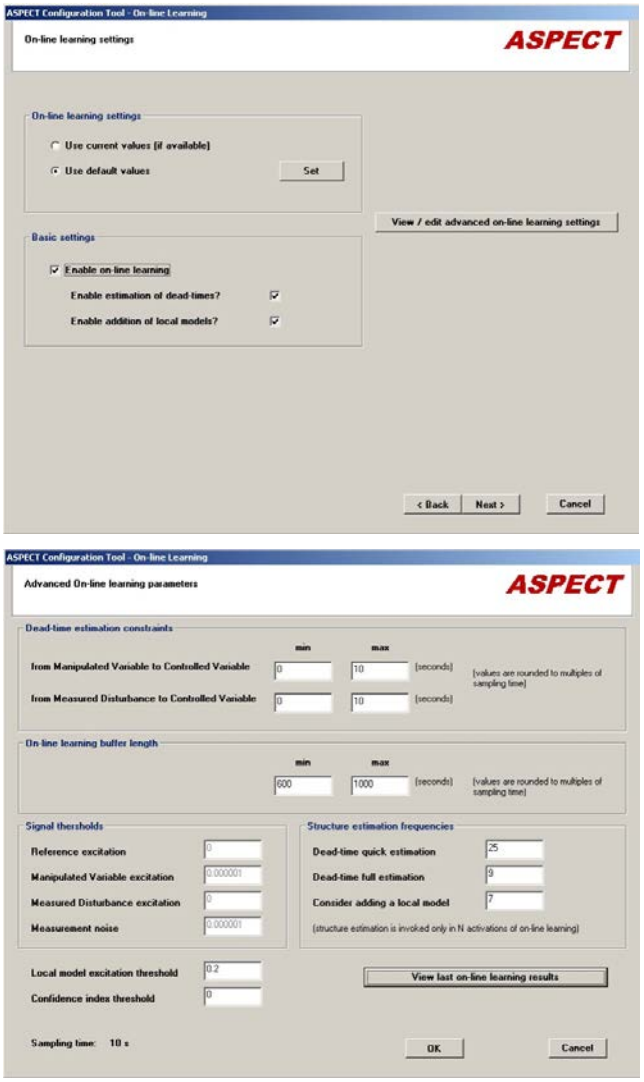


Fig. 12. OLA Configuration in original CT. Basic OLA settings (top) and advanced OLA parameters (bottom).

Alternative CT

The alternative CT is built on a tab-based user interface and designed to allow experienced engineers closer interaction with the stand-alone, non-adaptive FPSC controller configuration. It assumes the self-tuning functionality of the RTM, and allows detailed supervision or overriding of all the self-tuning steps.

The user interface is composed of the following tabs:

1. *Overview:* a condensed overview of the basic configuration parameters, an overview of the status of the local model estimation and local controller tuning, with buttons that trigger actions for the experimental tuning of local controllers.
2. *Connection:* OPC communication settings with the corresponding FPSC controller block on a PLC.
3. *Experiment:* contains a list of experiments for local controller tuning, scheduled in the Overview tab. Untypical plant experiments may also be triggered. When an experiment is active, its progress is displayed in a separate window.
4. *Measurements processing:* displays the experimental results, estimates the excitation of local models, enables import, export, filtering, and trimming of measurements for model identification.
5. *Local model:* displays the estimated local model parameters (MIA vs. OLA) and the model simulation results.
6. *Local controller:* displays the tuned local-controller parameters and a simulated response of the closed-loops system to a step change in the reference or disturbance signal.

Tabs 4-6 refer to each configured local controller (scheduling variable position, shown in the upper right quadrant) individually. For illustration, tabs 1 and 5 are shown in Fig. 13.

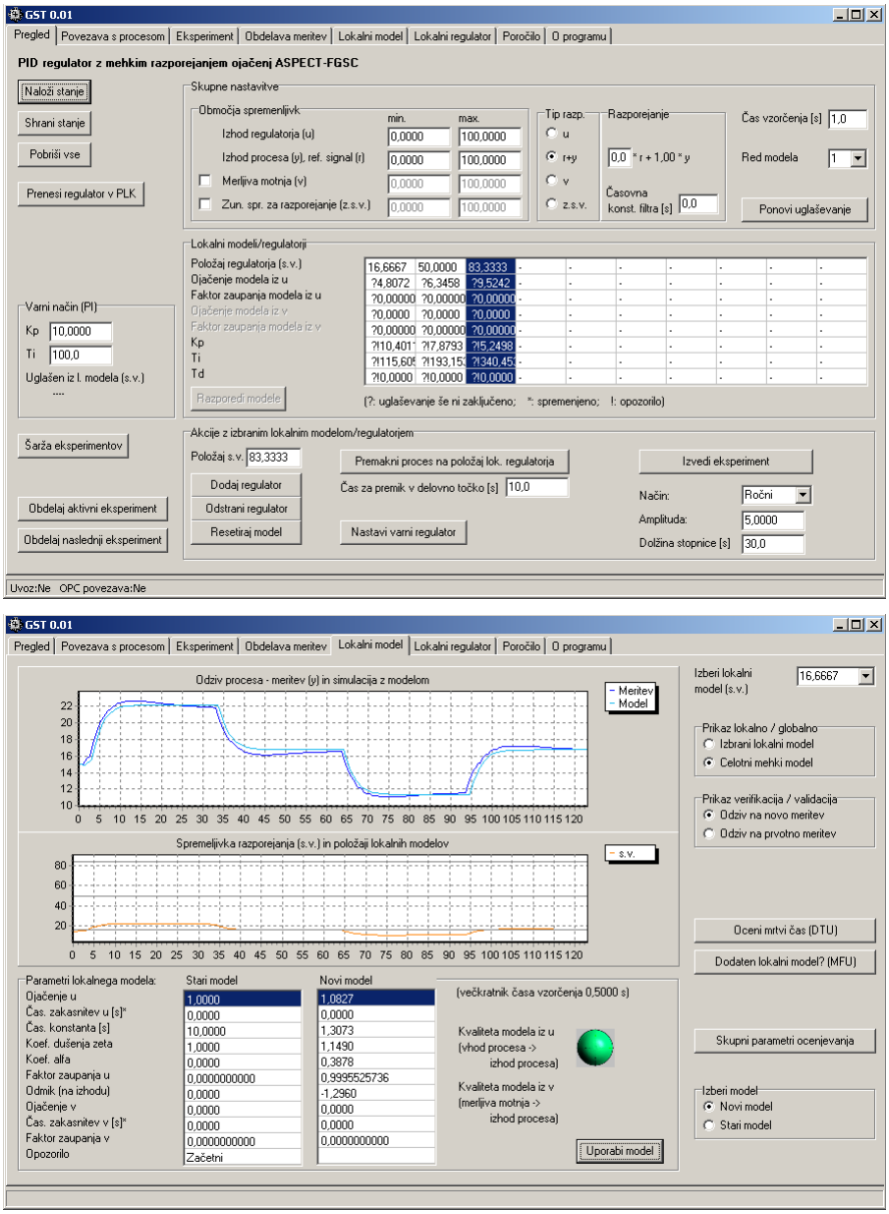


Fig. 13. Alternative CT sample tabs (in Slovene). Overview tab (top) and Local Model tab (bottom).

Experimental application to a valve-testing apparatus

The ASPECT controller was tested in several pilot applications, for example, on a gas-liquid separator [bibKVDG03] and a hydraulic pilot plant [bibBSGD09]; in this section we present a pilot application on an apparatus for testing hydraulic valves, located in a hydraulic-equipment production plant. A simplified scheme of the apparatus is shown in Fig. 14. The apparatus is composed of a boiler with local temperature control, three pumps P1-P3, a pressure sensor PT, a valve test stand with a pressure-difference sensor ΔPT, three flow meters QT1-QT3 that may be connected alternately for different measurement ranges, and an expansion vessel. The pumps are connected in parallel and may be activated in different combinations so that different flow ranges may be achieved. They are equipped with frequency converters; when switched on, all of them receive the same control signal *u*.

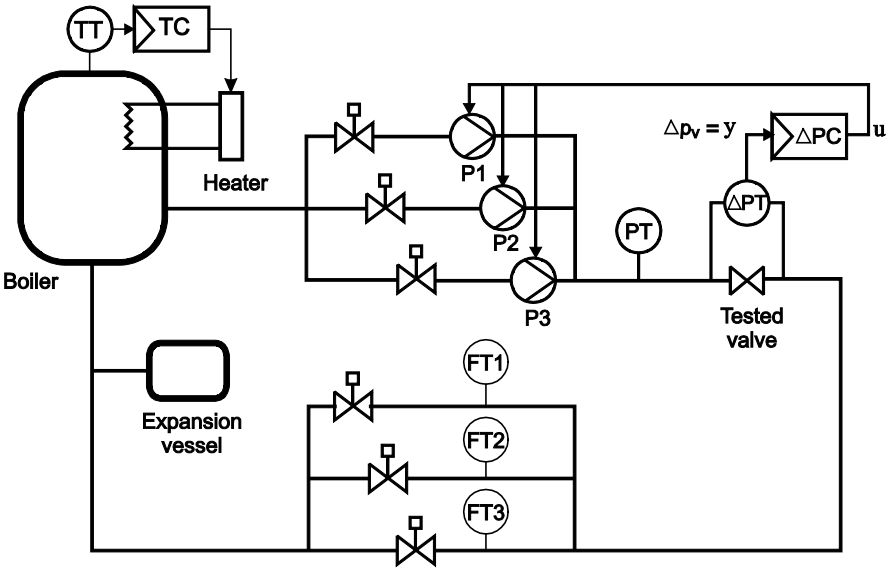


Fig. 14. Apparatus for testing a hydraulic valve (simplified)

The apparatus is used for testing various types of valves in a range of controlled operating conditions. The most important control task is to control the pressure difference on the tested valve Δp_v (also denoted as the process output *y*), by adjusting the control signal *u* that is connected to the active pumps. The process is nonlinear and time-varying because:

1. The steady-state relation between the pressure difference on the valve Δp_v and the mass flow through the valve Q_m (related to the pump rotation speed ω) is quadratic;
2. The openness of the valve S_v is sometimes changed during a test, but the signal S_v is generally not available (manual valves);
3. Different pumps (or combinations of pumps) may be used, according to the size of the valve.

These factors severely affect the process dynamics; therefore, the performance of the previously existing control system based on a fixed PI controller was considered unsatisfactory.

The scheduling variable selection is a crucial step when applying a parameter-scheduling controller. While the nonlinearity (a) alone may be easily solved using scheduling from Δp_v, the condition (b), in particular, makes the problem considerably more difficult, because it is necessary that the system should find the proper tuning autonomously, not by entering the unknown process parameters manually. Process modelling was used[2] to determine a suitable scheduling variable, namely the quotient Δp_v / Q_m, which is easily computed from the available signals [bib-GDVK06]. To improve the reliability of the computation of this quotient at low measurement values of Q_m, the latter was replaced by the control signal u filtered by a first-order lag filter that approximates the pump dynamics.

Once the scheduling variable s is configured, the commissioning of the ASPECT controller is an empirical procedure, supported by the automatic experimentation functionality of the OS. Firstly, the Safe Mode is tuned, so that its PI controller maintains stable control over the whole operating region. Then, the local model/controller positions are selected; a default equidistant distribution of six positions over the operating range of s is used in this application. Because experimentation with the process is allowed, the typical procedure involving a batch of experiments is used, with each experiment being made in the vicinity of one local model position. In practice, this is the simplest way to ensure proper excitation of the signals. Using the Safe Mode, the process is consecutively brought to each of the s positions, where auto-tuning experiments are activated by the push of a button. The OS conducts a mode switch (open-loop experiments are preferred), injects the excitation signal containing four step changes, invokes model identification and CAA tuning at the end of the experiment, and finally restores the original mode. For the first two local models, the excitation signal amplitude is 4%. Due to lower process gain, it is increased to 8% for other local models, in order to improve the signal-to-noise ratio. An overview of the MIA status shows that all the local models have been identified successfully. The Auto Mode is configured after

[2] Generally, the ASPECT controller is intended to be tuned empirically through experimentation, and process modelling is not required for simpler scheduling control applications.

the engineer confirms the new controller parameters. Table 1 displays the results of this experimental tuning procedure on the process.

Table 1. Experimental tuning results – local model and controller parameters

Local model positions	Local model parameters										Local controller parameters	
	OLA model parameters (T_{samp} = 0.5 s)						Derived parameters[a]					
$s = \alpha \frac{\bar{y}}{u}$	*du*	b_1	b_2	a_1	a_2	*r*	K_{ol}	$T_{90\%}$	T_1	T_2	*Kp*	*Ti*
0.10	1	0.002	0.009	-1.212	0.256	-0.002	0.25	20.0	8.01	0.38	5.78	8.03
0.26	1	0.003	0.014	-1.369	0.410	-0.002	0.41	16.5	6.55	0.61	2.58	6.56
0.42	1	0.004	0.017	-1.392	0.432	-0.005	0.52	16.5	6.41	0.66	1.95	6.29
0.58	1	0.003	0.022	-1.401	0.442	-0.005	0.61	15.5	6.09	0.68	1.54	6.12
0.74	1	0.001	0.026	-1.471	0.508	-0.008	0.73	15.0	5.77	0.85	1.14	5.92
0.90	1	-0.001	0.044	-1.397	0.440	-0.019	1.00	15.0	5.80	0.69	0.86	5.97

[a] K_{ol} is the open-loop model gain. $T_{90\%}$ is the rise time from 0 to 90% of the open-loop step response in s. T_1 and T_2 are the denominator time constants of the continuous-time equivalent model in s. α = 1.89 is a scaling factor.

The open-loop gain of the local models obviously rises with *s*, which results in a decrease in *Kp*. A decreasing trend of T_1 with s can also be noticed, which is associated with the pump dynamics and which has the most influence on *Ti*. There is a considerable but acceptable difference in *Kp* between the first two local controllers. The differences between the local controller parameters in the higher range of *s* are small; fewer local controllers could be used in that region.

The control performance is shown for the PI controller, realised using the Safe Mode of the ASPECT controller, and the FPSC controller. Fig. 15 shows the measured process response to a sequence of step changes in the set-point signal over the whole operating range when using the PI controller. The parameters of the PI controller were determined so that the optimal response was achieved at lower values of Δp_v. As the pressure increases, the response became oscillatory. Fig. 16 shows the response when using the FPSC control algorithm. Here, the performance is very good over the entire operating range. In addition to the signals shown in Fig. 16, the scheduling variable *s* is also shown in the bottom graph of Fig. 16.

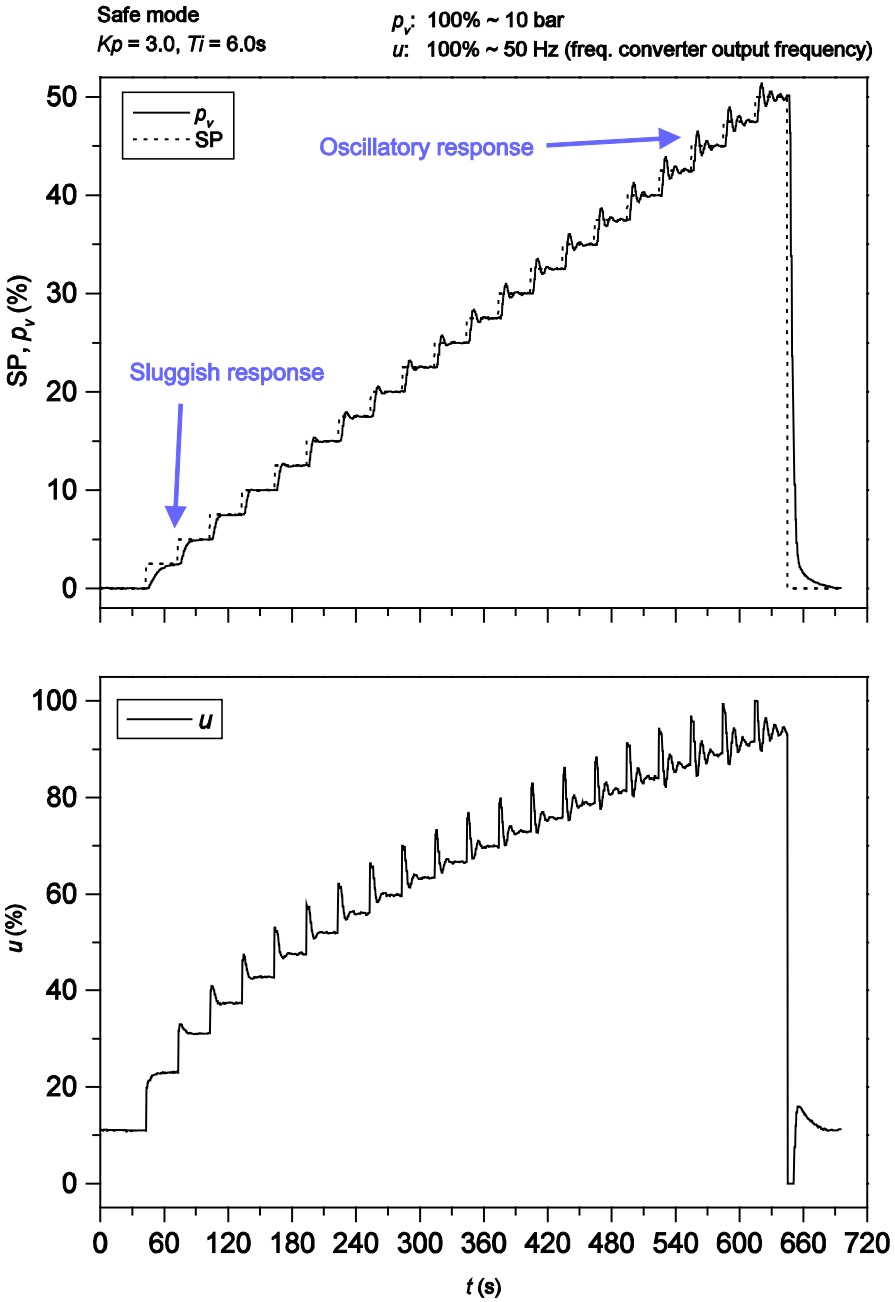


Fig. 15. Control of pressure difference p_v using the PI controller. Process output p_v and its set-point (top), pump control signal u (bottom).

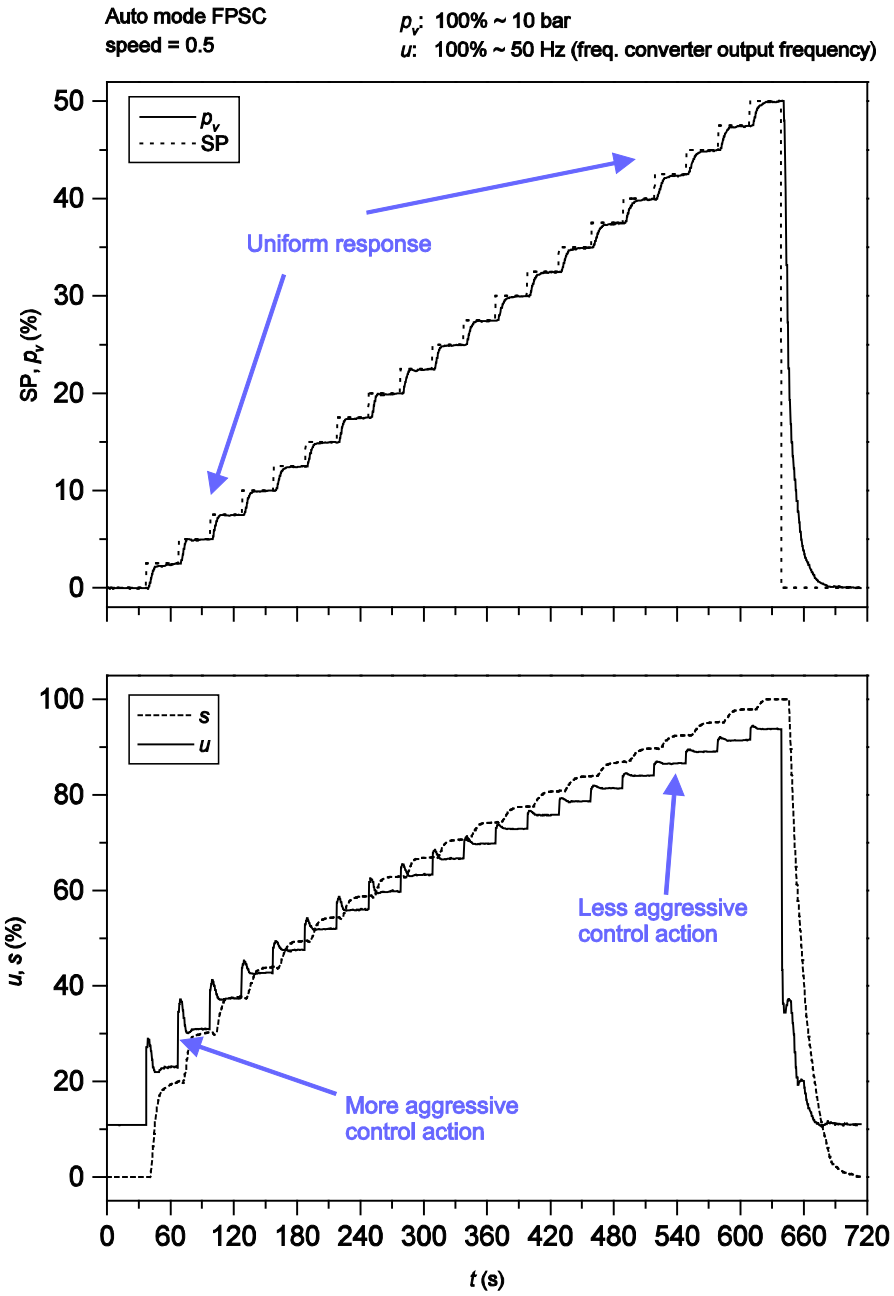


Fig. 16. Control of pressure difference p_v using the FPSC algorithm. Process output p_v and its set-point (top), pump control signal u and scheduling variable s (bottom).

Discussion in the context of theory/practice issues

The main motivation in pursuing the present work was to build an advanced control system for non-linear processes that would be simple to use and would run on PLC platforms. In this way, some theoretically already-established approaches could be brought closer to everyday use. Although the basic goals of the project were achieved, some important problems were encountered that substantially influenced the final results, such as the following:

- The entire concept of the control system relies on models identified from process data; therefore, the quality of control depends on the quality of models. We were aware of this fact right from the beginning of the project and planned to have an even larger set of models available for various situations. Although such an approach would certainly improve the versatility and usability of the control system, this idea had to be abandoned due to time and financial limitations.
- The three CAAs come from substantially different theoretical backgrounds. However, when tested in pilot applications in an industrial environment, they produced similar results, with only a slight performance advantage with regard to the DTCC. This is mainly due to the same underlying model structure and because the performance is limited by the achievable model accuracy. The latter is constrained by the time considered feasible for modelling and tuning in the process industry. The differences among the algorithms become more pronounced in the case of specific process requirements; for example, the DTCC shows additional advantages with processes that involve more significant dead-time, while the RBNC could show more value in the case of additional safety requirements.
- One of the main concerns was how to implement the rather complex structure of the control system on the target PLC platforms. To do this gradually, we decided to first develop the algorithms in Matlab, then build the controller on the intermediate PC platform in the C programming language, and to finally transfer it to the target PLC platform. It turned out that overcoming the platform differences was underestimated; major implementation redesigns of the software were required due to the differences in the operating systems and GUI implementation.
- An important lesson learned was that the main restrictions for the implementation of advanced algorithms on the PLC platform are not the computation and storage capabilities, but rather the limitations of the human-machine interaction for operator interaction and even more for application development.
- One of the basic ideas of the project was to develop a system which would be appropriate for less-experienced users. Such a system must have a high

degree of autonomy in its actions, a large set of "safety jackets" to prevent undesired behaviour, and as small an interaction with operators as possible. Although an important step in this direction was made in the frame of this project, it turned out that there is still much room for improvement. It is also quite obvious that the solution to these problems has very little to do with control algorithms, and much more to do with the system around the algorithms. Therefore, much more research-and-development effort will have to be devoted to this area. However, to make a real breakthrough, some knowledge from other fields (e.g., cognitive systems) will also have to be employed.

Conclusion

An advanced, self-tuning, nonlinear controller was successfully implemented on an industrial PLC platform. An experimental pilot application for pressure control in a hydraulic apparatus was presented. Compared to the industry-standard PI controller, a considerable improvement in control performance was achieved using the advanced control algorithms. Moreover, this performance was easily achieved in practice with self-tuning using an online learning procedure, by performing a sequence of short experiments around a few operating points. The modular, multi-agent structure contributes to the remarkable flexibility of the control system, such that it is easily reconfigured for various requirements. Parts of the applied algorithms are incorporated into the software for the design of the PLC systems control solutions of Mitsubishi Electric.

Acknowledgments

The contributions of all other partners in the ASPECT project are gratefully acknowledged. The ASPECT project was financially supported by the EC under contract IST-1999-56407. ASPECT ©2002 software is the property of INEA d.o.o., Indelec Europe S.A., and Start Engineering JSCo. This chapter is based on: Gerkšič S et al. (2006) Advanced control algorithms embedded in a programmable logic controller, *Control Engineering Practice* 14:935–948 ©Elsevier.

References

[bibABLM01] Anderson BDO, Brinsmead T, Liberzon D, Morse AS (2001) Multiple model adaptive control with safe switching. *International Journal of Adaptive Control and Signal Processing*, 15:445–470

[bibAF04] Angelov PP, Filev DP (2004) An approach to online identification of Takagi–Sugeno fuzzy models. *IEEE Transactions on Systems Man and Cybernetics*, part B, 34(1):484–497

[bibAH95] Åström KJ, Hägglund T (1995) *PID Controllers: Theory, Design, and Tuning*. ISA, International Society for Measurement and Control

[bibBOOB02] Babuska R, Oosterhoff J, Oudshoorn A, Bruijn PM (2002) Fuzzy self-tuning PI control of pH in fermentation. *Engineering Applications of Artificial Intelligence*, 15:3–15

[bibBeq91] Bequette BW (1991) Non-linear control of chemical processes: a review. *Industrial and Engineering Chemistry Research*, 30:1391–1413

[bibBSGD03] Blažič S, Škrjanc I, Gerkšič S, Dolanc G, Strmčnik S, Hadjiski MB, Stathaki A (2003) On-line Fuzzy Identification for an Advanced Intelligent Controller. *Proc. IEEE International Conference on Industrial Technology ICIT 2003*. Maribor, Slovenia, 912–916

[bibBSGD09] Blažič S, Škrjanc I, Gerkšič S, Dolanc G, Strmčnik S, Hadjiski MB, Stathaki A (2009) Online fuzzy identification for an intelligent controller based on a simple platform. *Engineering Applications of Artificial Intelligence*, 22:628–638

[bibFra95] Frantz FK (1995) A taxonomy of model abstraction techniques, Proceedings of the 1995 Winter Simulation Conference, Arlington, VA, USA, 1413–1420

[bibDC03] Dougherty D, Cooper D (2003) A practical multiple model adaptive strategy for multivariable model predictive control. *Control Engineering Practice*, 11: 649–664

[bibDS11] Dovžan D, Škrjanc I (2011) Recursive clustering based on a Gustafson–Kessel algorithm. *Evolving Systems*, 2:15–24

[bibDR09] Dahleh M, Rinehart M (2009) Networked Decision Systems. In: Samad T, Annaswamy A (eds), *The Impact of Control Technology*. IEEE Control Systems Society report, online: http://ieeecss.org/main/IoCT-report

[bibGDVK06] Gerkšič S, Dolanc G, Vrančić D, Kocijan J, Strmčnik S, Blažič S, Škrjanc I, Marinšek Z, Božiček M, Stathaki A, King R, Hadjiski M, Boshnakov K (2006) Advanced control algorithms embedded in a programmable logic controller. *Control Engineering Practice*, 14:935–948

[bibGHP02] Gundala R, Hoo KA, Piovoso MJ (2000) Multiple Model Adaptive Control Design for a Multiple-Input Multiple-Output Chemical Reactor. *Industrial and Engineering Chemistry Research*, 39:1554–1564

[bibHA00] Hägglund T, Åström KJ (2000) Supervision of adaptive control algorithms. *Automatica*, 36:1171–1180

[bibHS94] Henson MA, Seborg DE (1994) Adaptive non-linear control of a pH neutralisation process. *IEEE Transactions on Control Systems Technology*, 2:169–182

[bibHS97] Henson MA, Seborg DE (1997) *Non-linear process control*. Upper Saddle River NJ: Prentice-Hall PTR

[bibIne01] INEA d.o.o. (2001) *IDR BLOK Process Control Tools for Mitsubishi Electric PLC's, User's Manual, Ver. 4.20*. Ljubljana: INEA d.o.o.; http://www.inea.si/index.php?kategorija=158

[bibKZSV02] Kocijan J, Žunič G, Strmčnik S, Vrančić D (2002) Fuzzy gain-scheduling control of a gas+liquid separation plant implemented on a PLC. *International Journal of Control*, 75(14):1082–1091

[bibKVDG03] Kocijan J, Vrančić D, Dolanc G, Gerkšič S, Strmčnik S, Škrjanc I, Blažič S, Božiček M, Marinšek Z, Hadjinski MB, Boshnakov K, Stathaki A, King R (2003) Autotuning non-linear controller for industrial use. *Proc. IEEE International Conference on Industrial Technology ICIT 2003*. Maribor, Slovenia, 906–910

[bibKKS02] King RE, Koumboulis FN, Stathaki A (2002) Intelligent hybrid industrial control. *Proc. 1st Intl. IEEE Symp. Intelligent Systems*, Varna, September 2002. 2:2–6

[bibKKS07] Koumboulis FN, King RE, Stathaki A (2007) Logic-based switching controllers – A stepwise safe switching approach. *Information Sciences: an International Journal*, 177(13):2736–2755

[bibLL98] Leith DJ, Leithead WE (1998) Appropriate realisation of MIMO gain-scheduled controllers. *International Journal of Control*, 70(1):13–50

[bibLju87] Ljung L (1987) *System Identification*. Englewood Cliffs NJ: Prentice Hall

[bibMac02] Maciejowski JM (2002) *Predictive control with constraints.* Harlow UK : Prentice Hall

[bibMor95] Morse AS (1995) Control using logic-based switching. In: Isidori A. (ed) *Trends in Control – a European perspective*, London: Springer-Verlag, 69–113

[bibMSJ97] Murray-Smith R, Johansen TA (eds) (1997) *Multiple model approaches to modelling and control*. London - Bristol: Taylor and Francis

[bibQB99] Qin SJ, Badgwell TA, (1999) An Overview of Nonlinear Model Predictive Control Applications. In Allgöwer F, Zheng A (eds) *Nonlinear Model Predictive Control*. Basel: Birkhäuser, 370–392

[bibRic93] Richalet J (1993) *Pratique de la Commande Prédictive*. Paris: Editions Hermès

[bibSA11] Samad T, Annaswamy A (eds) (2011) *The Impact of Control Technology*. IEEE Control Systems Society report, online: http://ieeecss.org/main/IoCT-report

[bibSeb99] Seborg DE (1999) A perspective on advanced strategies for process control (revisited). In: Frank PM (ed) *Advances in Control, Highlights of ECC'99*, London: Springer Verlag, 103–134

[bibSmi59] Smith OJM (1959) A controller to overcome dead-time. *ISA Journal*, 6:2:28–33

[bibSHL90] Stephanopoulus G, Henning G, Leone H (1990) Model LA – A modelling language for process engineering, II. Multifaceted modelling of processing systems. *Computers and Chemical Engineering*, 14:847–869

[bibTIA98] Takatsu H, Itoh T, Araki M (1998) Future needs for the control theory in industries – report and topics of the control technology survey in Japanese industry. *Journal of Process Control*, 8(5-6):369–374

[bibTHC97] Tan S, Hang C-C, Chai J-S (1997) Gain scheduling: from conventional to neuro-fuzzy. *Automatica*, 33(3):411–419

[bibVH07] Vrančić D, Huba, (2007) LEK tuner - program package for tuning PID controllers. In: Mikleš J, Fikar M, Kvasnica M (eds), Proc. 16th International Conference Process Control 2007, Štrbské Pleso, Bratislava: Slovak University of Technology, 225-1–225-5

[bibVSJ01] Vrančić D, Strmčnik S, Juričić Đ (2001) A magnitude optimum multiple integration tuning method for filtered PID controller. *Automatica*, 37:1473–1479

[bibWhi46] Whiteley AL (1946) Theory of servo systems, with particular reference to stabilization. *The Journal of IEE, Part II*, 93(34):353–372

[bibWJ95] Wooldridge M, Jennings NR (1995) Intelligent Agents, Theory and Practice. *Knowledge Engineering Review*, 10(2):115–152

[bibZei79] Zeigler PB (1979) Structuring principles for multifaceted system modelling, In: Zeigler BP, Elzas MS, Klir GJ, Ören TI (eds) Methodology in Systems Modelling and Simulation, North-Holland, 93–135