ISA Transactions 48 (2009) 24–31

Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

Fuzzy-model-based hybrid predictive control

Alfredo Núñez [a], Doris Sáez [a,*], Simon Oblak [b], Igor Škrjanc [b]

[a] Electrical Engineering Department, University of Chile, Santiago, Chile

[b] Faculty of Electrical and Computer Engineering, University of Ljubljana, Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 20 August 2007
Received in revised form
2 October 2008
Accepted 16 October 2008
Available online 22 November 2008

Keywords:
Fuzzy model
Hybrid systems
Predictive control
Genetic algorithm

ABSTRACT

In this paper we present a method of hybrid predictive control (HPC) based on a fuzzy model. The identification methodology for a nonlinear system with discrete state–space variables based on combining fuzzy clustering and principal component analysis is proposed. The fuzzy model is used for HPC design, where the optimization problem is solved by the use of genetic algorithms (GAs). An illustrative experiment on a hybrid tank system is conducted to demonstrate the benefits of the proposed approach.

© 2008 ISA. Published by Elsevier Ltd. All rights reserved.

1. Introduction

Most industrial processes contain continuous and discrete components, for example, discrete valves, discrete pumps, on/off switches or logical overrides. These hybrid systems can be defined as hierarchical systems given by continuous components and/or discrete logic [1]. The mixed continuous–discrete nature renders it impossible for a designer to use conventional identification and control techniques. Thus, in the case of industrial-process control new tools for hybrid-system identification and control design need to be developed.

Hybrid systems have received much attention from computer science and from the control community; nevertheless, there is as yet no general design methodology for the identification of hybrid systems [2]. In recent years, some hybrid-fuzzy-identification methods, based on fuzzy clustering, have been proposed. Palm and Driankov [3] presented a hierarchical black-box identification of the resulting fuzzy switched systems. Girimonte and Babuška [4] described two structure-selecting methods for nonlinear models with mixed discrete and continuous inputs. However, the drawbacks of these methods are lack of generalization and the increase in computation time resulting from the increase in the search horizon.

Regarding hybrid predictive-control (HPC) design, Slupphaug et al. [5,6] described a predictive controller with continuous and integer-input variables that is tuned using nonlinear mixed-integer programming. They showed that it performs better than a predictive control strategy with the separation of continuous and integer variables. Bemporad and Morari [7] presented a predictive control scheme for hybrid systems including operational constraints. In this case, the problem is solved using mixed-integer quadratic programming (MIQP). The main problem of MIQP is computational complexity, which increases the time to find the solution. To reduce the computation time, Thomas et al. [8] proposed partitioning in the state–space domain, and Potočnik et al. [9] proposed building and pruning an evolution tree of an HPC algorithm with a discrete input based on a reachability analysis. Sáez et al. [10] and Cortés et al. [11] introduced a hybrid-predictive-adaptive-control scheme for solving the dynamic multi-vehicle pick-up and delivery problem.

All the previous works related to HPC are based on linear models. However, the majority of industrial processes are nonlinear in nature. Karer et al. [12] introduced a compact formulation of a hybrid fuzzy model and used it for model-predictive control (MPC). In our work we tackle the problems of nonlinearity and the hybrid nature of the system by the inherent use of a fuzzy model in HPC. As the optimization of the objective function in the case of the hybrid fuzzy-predictive control (HFPC) is a highly nonlinear problem, the genetic optimization algorithm was employed [13]. The idea of using genetic algorithms (GAs) in fuzzy predictive control is not new. Van der Lee et al. [14] presented a generalized automated tuning algorithm for MPCs combining GA with multi-objective fuzzy decision-making. Na and Upadhyaya [15] applied a combination of MPC, GA optimization and fuzzy identification to the design of the thermoelectric power control. Sarimveis and Bafas [16] used the

* Corresponding author. Tel.: +56 2 9784 207; fax: +56 2 6720 162.
E-mail addresses: alfnunez@ing.uchile.cl (A. Núñez), dsaez@ing.uchile.cl (D. Sáez), simon.oblak@fe.uni-lj.si (S. Oblak), igor.skrjanc@fe.uni-lj.si (I. Škrjanc).

0019-0578/$ – see front matter © 2008 ISA. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.isatra.2008.10.007

GA in fuzzy-predictive control without discrete variables to provide reasonable solutions in a reduced computation time. One of the strong points of the approach is that the feasibility of the optimization solution in each time sample is guaranteed, in contrast to the conventional optimization techniques, which can potentially fail due to the complexity of the optimization problem. The problem that we deal with in this paper is even more complex because of the discrete states, so that the use of a GA is fully justified as it reduces the computational load substantially.

The outline of the paper is as follows. In Section 2 the fuzzy modelling, based on fuzzy clustering and principal-component analysis (PCA), of a switching hybrid system with discrete states is presented. In Section 3 the HFPC design based on the GA is discussed. Section 4 gives the simulation results of the control of a hybrid tank system, and Section 5 concludes the paper.

2. Hybrid fuzzy modelling

In the paper we deal with hybrid discrete-time models that have mixed continuous and discrete states. We consider systems where the continuous states remain continuous even when the discrete states are changed. The transition of a system state occurs when one or more continuous states satisfy the conditions defined for each transition. This type of hybrid system is known as the Witsenhausen type [17]. It is described in general form as

$$\begin{aligned} x_{k+1} &= f_{q_k}(x_k, u_k) \\ q_k &= g(x_k, q_{k-1}) \end{aligned} \quad (1)$$

where $x_k \in R^n$ is the state vector, $u_k \in R^m$ is the input vector, and $q_k \in Q$ where $Q = \{1, \ldots, s\}$ is the switching-region state vector. This means that the hybrid-system states are described at any time instant by the set of states (x_k, q_k) in the domain $R^n \times Q$. In general the switching state (discrete state) q_k depends on the state x_k and the previous switching state q_{k-1}. The local behaviour of the system is described by the function f_{q_k}, and g is the transition function of the discrete switching-region states. This type of system can be represented by the following two-level fuzzy system, which was described by Tanaka et al. [18]. These two levels are the switching-region level and the local-fuzzy level. The classical form of the system is described as

$$\begin{aligned} x_{k+1} &= \sum_{i=1}^{s} \sum_{j=1}^{R_i} \upsilon_i(q_k)\beta_{ij}(z_k)\left(a_{ij}x_k + b_{ij}u_k + r_{ij}\right) \\ \upsilon_i(q_k) &= \begin{cases} 1, & q_k \in S_i \\ 0, & \text{otherwise} \end{cases} \\ \beta_{ij}(z_k) &= \frac{\prod_{r=1}^{p} A_{ij,r}(z_{k,r})}{\sum_{j=1}^{R_i} \prod_{r=1}^{p} A_{ij,r}(z_{k,r})} \end{aligned} \quad (2)$$

where s is the number of switching regions, r_i is the number of rules in the ith switching region S_i, $\upsilon_i(q_k)$ is a crisp switching-region weighting function, which is defined by the current switching state q_k and defines the current switching region, and $\beta_j(z_k)$ is a local fuzzy-membership value defined by the local premise vector $z_k = \left[z_{k,1}\ z_{k,2} \ldots z_{k,p}\right]^{\mathrm{T}}$, $A_{ij,r}$ $(r = 1, \ldots, p)$ are the local fuzzy sets, and $A_{ij,k}(z_{k,j})$ is the membership degree of the premise variable $(z_{k,j})$ to the membership set $A_{ij,k}$.

The two-level fuzzy form is very appealing in the case when the switching regions are exactly or very precisely known. However, we are dealing with the case where the switching regions are not known in advance and have to be estimated from input–output measurements. Therefore, the model will be structured as a one-level fuzzy model with a modified membership-function distribution. The model can be described as

$$\begin{aligned} x_{k+1} &= \sum_{i=1}^{s\cdot R_i} \mu_i(z_k)\left(a_i x_k + b_i u_k + r_i\right) \\ \mu_i(z_k) &= \frac{\prod_{r=1}^{p} B_{i,r}(z_{k,r})}{\sum_{i=1}^{s\cdot R_i} \prod_{r=1}^{p} B_{i,r}(z_{k,r})} \end{aligned} \quad (3)$$

where $\mu_i(z_k)$ stands for the membership degree of the product between the crisp switching-region membership value and the local fuzzy membership value. This mixed nature of the system behaviour requires more attention in terms of the membership-function arrangement. The criterion for the arrangement is based on analyzing the eigenvalues and the eigenvectors of the covariance matrices of the clusters, obtained by clustering algorithms. Let the centers of the clusters be υ_i, the eigenvalues of the clusters be $\{\lambda_{i,1}, \lambda_{i,2}, \ldots, \lambda_{i,k}\}$, and the eigenvectors be $\{\phi_{i,1}, \phi_{i,2}, \ldots, \phi_{i,k}\}$, where the eigenvalues and the corresponding eigenvectors are arranged in descending order of the eigenvalues, and k denotes the dimension of the data. By analyzing the most important eigenvectors (the principal vectors or the principal components in which directions the most information is given), the switching region can be detected. In its "inverse" form, the criterion is known as the procedure to merge the clusters, and in our case it can be usefully applied to detect the switching regions:

- An experiment should be designed to obtain the information about all the possible clusters (behaviours of the system).
- The ratio between the principal-eigenvector elements is calculated for each cluster:

$$d_i = \left|\frac{\phi_{i,2}}{\phi_{i,1}}\right|. \quad (4)$$

- Consecutive ratios are compared to each other. Normally, the increments are small. However, around the switching region there occurs an abrupt change of ratio. These changes are detected by a predefined limit-value test, e.g., $d_i \geq \overline{d}$.
- Around the point where an abrupt change occurs, we distribute a designated number of membership functions. The purpose of this is to increase the density of the membership-space segmentation, so that the model is able to approximate the switching effect.

3. Hybrid fuzzy predictive control based on genetic algorithms

3.1. Hybrid fuzzy predictive control

The HFPC strategy is a generalization of model-predictive control (MPC), where the prediction model includes both discrete/integer and continuous variables. Here we propose an HPC based on a fuzzy model, described in Section 2.

In general, the HPC minimizes the following objective function:

$$\begin{aligned} \min J &= \min_{\{u(k), u(k+1), \ldots, u(k+N_u-1)\}} (J_1 + \lambda J_2), \\ J_1 &= \sum_{j=N_1}^{N_y} \left(\hat{y}(k+j) - r(k+j)\right)^2, \qquad J_2 = \sum_{j=N_1}^{N_u} \Delta u(k+j-1)^2 \end{aligned} \quad (5)$$

where J is the objective function, $\hat{y}(k+j)$ corresponds to the j-step-ahead prediction for the controlled variable, $r(k+j)$ is the reference, $\Delta u(k+j-1)$ is the increment of the control action, and λ is the weighting factor. N_1, N_y and N_u are the prediction horizons and the control horizon, respectively. The model predictions are given by the fuzzy model of the process, i.e.,

$$\hat{y}(k+j) = f\left(\hat{y}(k+j-1), \ldots, u(k+j-1), \ldots\right) \quad (6)$$

where f is the nonlinear function defined by the fuzzy model. The optimization results in a control sequence $\{u(k), \ldots, u(k+N_u-1)\}$.

As we assume that the HPC problem includes discrete input variables, the optimization could be solved by explicitly evaluating for all the possible feasible solutions (EE), Branch & Bound (BB) and other algorithms [19]. Next, we will present an efficient optimizer based on the GA in detail.

3.2. Optimization based on the genetic algorithm

The genetic algorithm is used to solve the optimization of an objective function because it can efficiently cope with mixed-integer nonlinear problems. Another advantage is that the objective-function gradient does not need to be calculated, which reduces the computational effort.

A potential solution of the genetic algorithm is called an individual. The individual can be represented by a set of parameters related to the genes of a chromosome and can be described in binary or integer form. The individual represents a possible control-action sequence

$$s^i = \begin{bmatrix} u^i(k) \\ u^i(k+1) \\ \vdots \\ u^i(k+N_u-1) \end{bmatrix} \quad (7)$$

where an element $u^i(k+j),\ j \in [0, N_u-1]$ is a gene, i denotes the ith individual from the population of possible individuals, and the individual length corresponds to the control horizon N_u.

Using genetic evolution, the fittest chromosome is selected to ensure the best offspring. The best parent genes are selected, mixed and recombined for the production of an offspring in the next generation. For the recombination of the genetic population, two fundamental operators are used: crossover and mutation. For the crossover mechanism, the portions of two chromosomes are exchanged with a certain probability in order to produce the offspring. The mutation operator alters each portion randomly with a certain probability [13].

In the paper the control-law derivation will be based on the simple genetic algorithm (SGA) [13]. We assume that the range of the manipulated variable is $[\underline{u}, \overline{u}]$, quantized by steps of size q, so that there are q possible inputs at each time instant. Therefore, the set of feasible control actions is

$$\mathcal{U} = \{u \mid u = n \cdot (u_{\max} - u_{\min})/q + u_{\min}\}, \quad n \in \{1, 2, \ldots, q\}. \quad (8)$$

Furthermore, we assume the probability that two selected parent individuals s^i and s^l undergo a crossover is p_c, and for mutation the probability is p_m. The control strategy, shown in Fig. 1, can be represented by the following steps:

1. Set the iteration counter to 1, and initialize a random population of P individuals, i.e., create P random integer feasible solutions of the manipulated variables for the HFPC problem. As the control horizon is N_u, there are q^{N_u} possible individuals.
2. Evaluate the objective function (5) for all the initial individuals of the population.
3. Select random parents from the population P (different vectors of the future control actions).
4. Generate a random number between 0 and 1. If the number is lower than the probability p_c, choose an integer $0 < c_p < N_u - 1$ (c_p denotes the crossover point) and apply the crossover to the selected individuals in order to generate an offspring. Fig. 2 describes the crossover operation for two individuals, s^i and s^l, resulting in s^i_{cross} and s^l_{cross}.

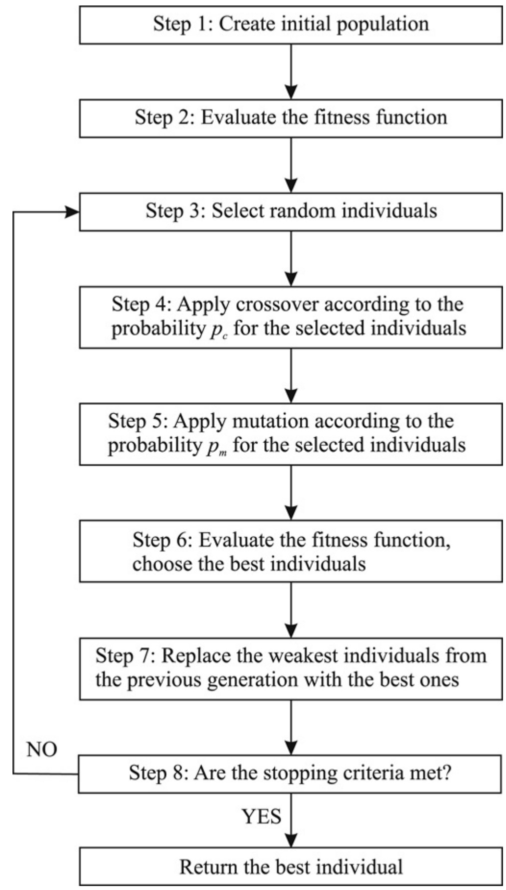

Fig. 1. The SGA-based control strategy.

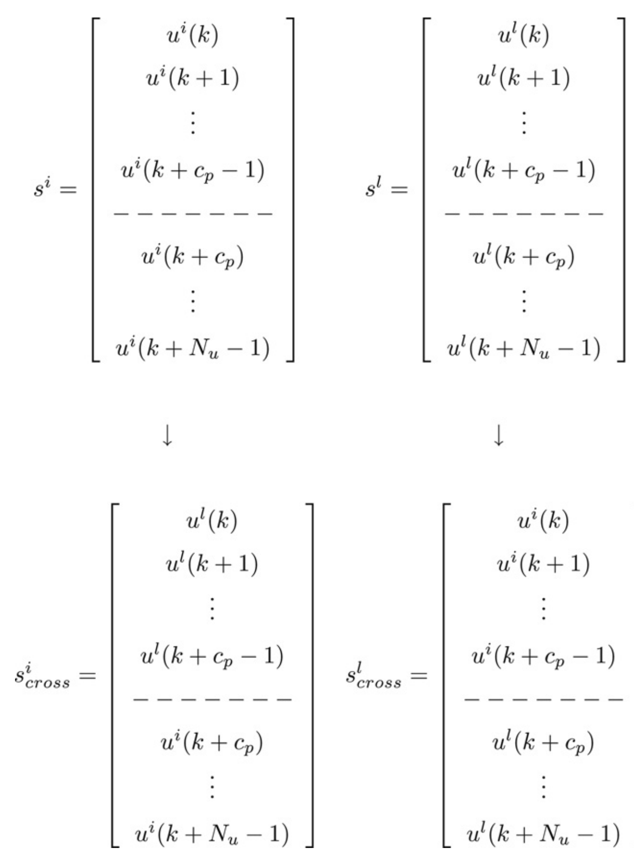

Fig. 2. Crossover of two individuals from c_pth place forward.

$$s^i = \begin{bmatrix} u^i(k) \\ u^i(k+1) \\ \vdots \\ u^i(k+c_m-1) \\ u^i(k+c_m) \\ u^i(k+c_m+1) \\ \vdots \\ u^i(k+N_u-1) \end{bmatrix} \longrightarrow s^i_{mut} = \begin{bmatrix} u^i(k) \\ u^i(k+1) \\ \vdots \\ u^i(k+c_m-1) \\ u^i_m \in \mathcal{U} \\ u^i(k+c_m+1) \\ \vdots \\ u^i(k+N_u-1) \end{bmatrix}$$

Fig. 3. Mutation of an individual in c_mth place.

Table 1
Parameters of the two-tank model.

R_1	25 cm	V_1	0.5 cm^2/s
R_2	15 cm	V_2	0.65 cm^3/s
H_1	100 cm	$H_{1\min}$	5 cm
K_{CP}	1 cm^3/s	$H_{2\min}$	20 cm
K_{ONOFF1}	4 cm^3/s	$H_{1\max}$	50 cm
K_{ONOFF2}	4 cm^3/s	$H_{2\max}$	90 cm

5. Generate a random number between 0 and 1. If the number is lower than the probability p_m, choose an integer $0 < c_m < N_u - 1$ (c_m denotes the mutation point) and apply the mutation to the selected parent in order to generate an offspring. Select a value $u^i_m \in \mathcal{U}$ and replace the value in the c_mth position in the chromosome. Fig. 3 describes the mutation operation for an individual s^i resulting in s^i_{mut}.
6. Evaluate the fitness given by the objective function (5) of all the individuals of the offspring population.
7. Select the best individuals according to the objective function. Replace the weakest individuals from the previous generation with the strongest individuals of the new generation.
8. If the objective-function value reaches the defined tolerance or the maximum generation number is reached (stopping criteria), then stop. In other cases, go to step 3.

The tuning parameters of the GA method are the number of individuals, the number of generations, the crossover probability p_c, the mutation probability p_m and the stopping criteria.

Remark 1. The genetic-algorithm approach in HFPC provides a sub-optimal discrete control law close to the optimal one. When the best solution is maintained in the population, it was shown [20, 16] that the GA converges to the optimal solution. However, due to the limited time between the sampling instances reaching the global optimum is not guaranteed. Nevertheless, the probabilistic nature of the algorithm ensures that it finds an approximately optimal solution. In contrast to that, following the Remark 5.3 in [16], the application of traditional optimization techniques to solve the same problem cannot guarantee even the calculation of a feasible solution because of the complexity of the optimization problem. Since in this case we are dealing with a complex mixed integer and nonlinear programming (MINLP), using the GA optimization is justified.

Remark 2. Using the GA optimization makes it easy to include the input and output constraints in the calculation of the control variable. The procedure is described in [16]; in general, it means a narrowing of the space for feasible solutions in each optimization step. However, this case is beyond the scope of this work.

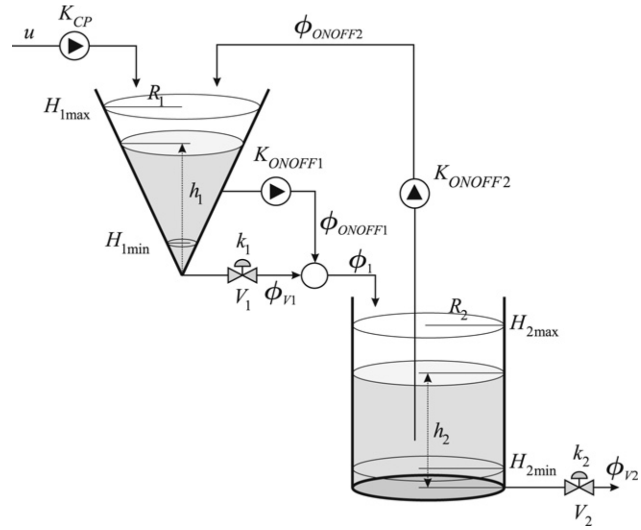

Fig. 4. The tank-system plant.

4. Tank system

4.1. Process description

The behaviour of the tank system, shown in Fig. 4, is defined by the following nonlinear differential and algebraic equations, which define the switching regions:

$$\begin{aligned} \frac{dh_1}{dt} \cdot \pi \cdot \frac{R_1^2}{H_1^2} h_1^2 &= K_{CP} \cdot u + \phi_{ONOFF2} - \overbrace{V_1 h_1}^{\phi_{V1}} - \phi_{ONOFF1} \\ \frac{dh_2}{dt} \cdot \pi \cdot R_2^2 &= \overbrace{V_1 h_1}^{\phi_{V1}} + \phi_{ONOFF1} - \overbrace{V_2 h_2}^{\phi_{V2}} - \phi_{ONOFF2} \end{aligned} \quad (9)$$

$$\begin{aligned} &\text{if } (h_2 \geq H_{2\min}) \text{ and } (h_1 < H_{1\max}) \quad \text{then } \phi_{ONOFF2} = K_{ONOFF2} \\ &\text{if } (h_1 \geq H_{1\max}) \text{ and } (h_2 < H_{2\max}) \quad \text{then } \phi_{ONOFF1} = K_{ONOFF1} \end{aligned} \quad (10)$$

where h_1 and h_2 stand for the level of the liquid in the first and the second tank, and $H_{1\min}$, $H_{1\max}$, $H_{2\min}$, and $H_{2\max}$ stand for the switching levels. The controlled variable in our case will be the level in the first tank h_1, and the manipulated variable is the voltage of the pump at the inlet u, which has discrete levels. It is also assumed that both levels, h_1 and h_2, are measured, and the measurements are corrupted with white noise that has a variance equal to 1. The excitation and the output signals of the plant are shown in Figs. 5 and 6. The signals were sampled with $T_s = 10$ s. Note that the rules in Eq. (10) represent the switching or hybrid behaviour of the system. The parameters used in the model are gathered in Table 1.

4.2. Fuzzy modelling of the hybrid system

The behaviour of the hybrid system will be modelled by the fuzzy-model structure from (3). The design of the membership-function distribution is the key element of the modelling procedure. In our case it is obtained by analyzing the principal eigenvectors of the covariance matrices of the clusters. The clusters are obtained from the data matrix, which is composed of measurements (the variables h_1 and h_2). The analysis of the principal eigenvectors for all the clusters is presented in Fig. 7, where the eigenvector-element ratio corresponds to its own cluster. It is clear that around the level of $h_2 = 50$ cm there is an abrupt change of the eigenvector ratio. This change implies a change in the system's behaviour and potentially indicates the switching region of the system. The idea is to put two

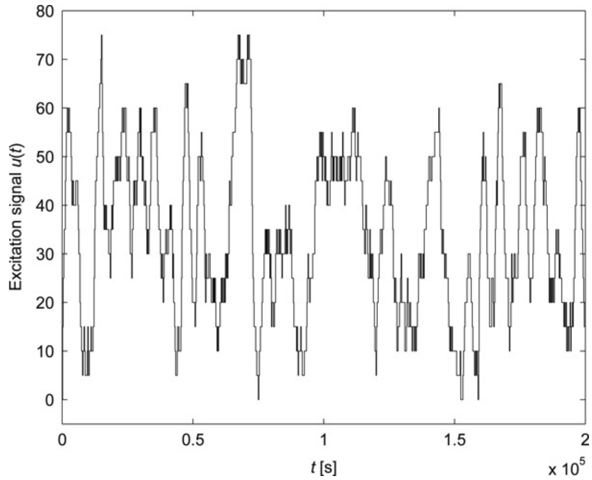

Fig. 5. The input signal u.

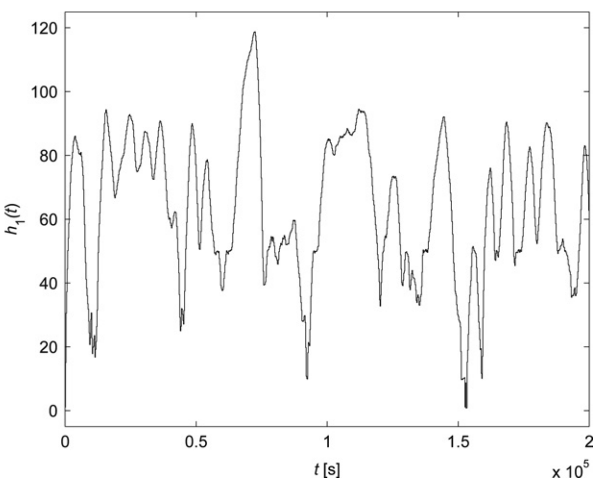

Fig. 6. The output signal h_1.

membership functions around each local extremum (the minimum and maximum of the eigenvector ratios). This is done because the switching region cannot be exactly defined (especially in the case of noisy data). This idea involves a tolerance band around the switching regions. In Fig. 7 the corresponding membership functions are also shown.

The structure of the fuzzy model follows the definition in Eq. (3), where the variable in the premise is $z_k = h_{1,k}$ and the consequent vector is equal to $x_c = \left[h_{1,k}\ u_k\ 1\right]^{\mathrm{T}}$. The parameters of the fuzzy model ($\theta_i = [a_i\ b_i\ r_i]$), obtained by a linear least-squares estimation, are given in Table 2. The validation of the designed fuzzy model is shown in Fig. 8. The proposed model gives a very good estimation of the process output, and inherently incorporates the hybrid (switching) nature of the system.

4.3. Hybrid predictive control based on the fuzzy model (HFPC)

The tuning parameters of the objective function in (5) are given by $N_1 = 1$, $N = N_y = N_u = 3$, and $\lambda = 0.001$. The total computation time required for the HFPC will be evaluated using a Intel® Core(TM) 2 CPU, 2.40 GHz processor and 3.25 GB of RAM. The sampling time is 10 s and the total simulation time is 6000 s. We will compare the results of the proposed method to the results obtained by using a branch-and-bound method (HFPC-BB) and explicit enumeration (HFPC-EE). The latter evaluates all the feasible control actions at every instant, while the HFPC-GA and HFPC-BB consider only a reduced space search. The parameters for HFPC-GA are as follows: mutation probability $p_m = 0.001$, crossover probability $p_c = 0.7$ and for the stopping criterion we used the maximum number of generations, which we obtain by further analyses.

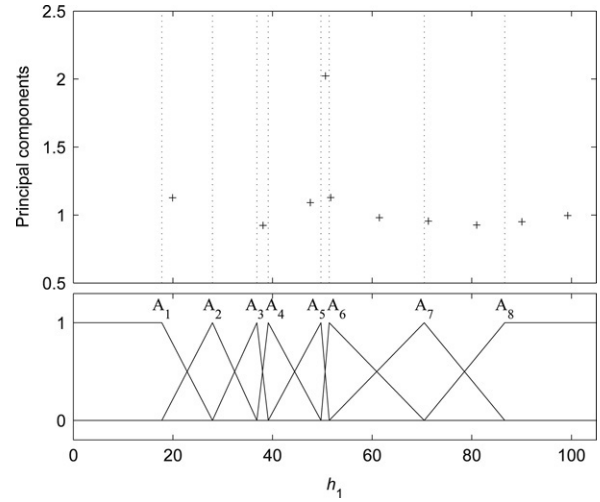

Fig. 7. The analysis of the eigenvectors and the corresponding membership functions.

Table 2
Parameters of the fuzzy model.

i	a_i	b_i	r_i
1	0.8376	0.3403	0.0386
2	0.9764	0.0522	0.0511
3	0.9873	0.0290	0.0305
4	0.9747	0.0196	0.7656
5	0.9933	0.0125	−0.0136
6	0.9946	0.0091	0.0265
7	0.9987	0.0066	−0.2163
8	1.0015	0.0045	−0.4334

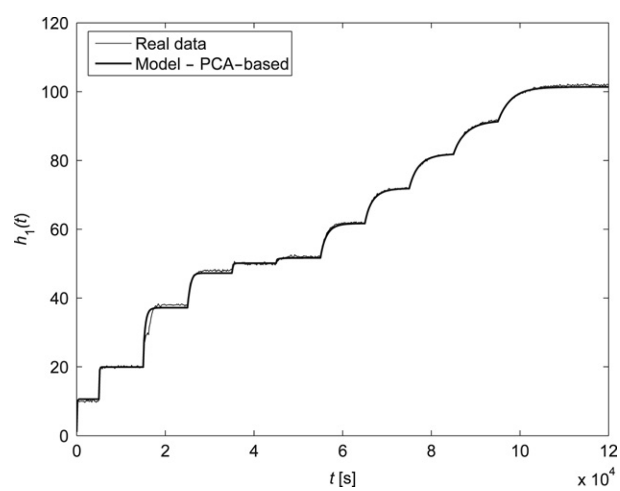

Fig. 8. Validation of the fuzzy model.

Fig. 9 shows the objective function as a function of the generation number for different numbers of individuals. Based on this figure, 30 generations with 14 individuals are selected in our example. Fig. 10 shows how this selection brings a tradeoff between the computation time and the value of the objective function. Fig. 11 presents the computation time as a function of the number of generations for different numbers of individuals.

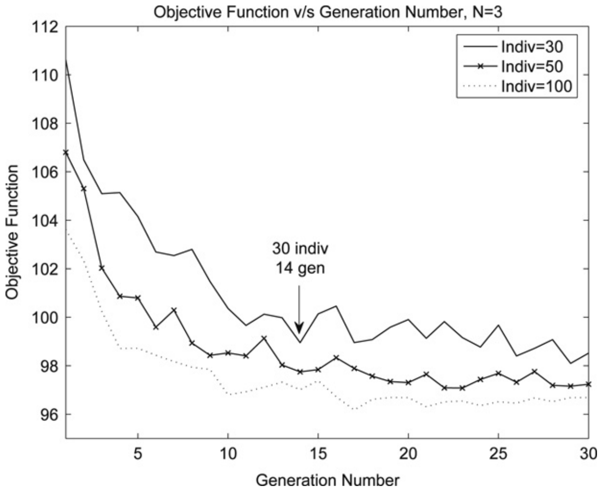

Fig. 9. Evolution of the objective function.

Fig. 10. Computation time vs. the objective function.

The computation time is linearly dependent on the generation number, and its slope increases slightly with the number of individuals. It is clear that the time needed to calculate the solution in each sampling time is shorter than the sampling time for all the cases. This means that all the proposed control strategies are suitable for real-time control in the sense of time consumption. For 30 generations with 14 individuals, the computation time was approximately 84.3 s (1.41% of the total simulation time), and the calculation time during each iteration was smaller than the sampling time.

With optimal values of 30 generation with 14 individuals, the results of the HFPC-GA are obtained. Figs. 12 and 13 show the controlled variable (conic tank level h_1) and the manipulated variable (discrete voltage of pump u), respectively, for the HFPC-GA, HFPC-EE and HFPC-BB. Figs. 14 and 15 show the response detail for 3500–5000 s.

Fig. 16 gives a comparison of mean computation times (MCT) for all three cases, and in Table 3 the mean values of the objective function 5, the total computation times and the mean computation times for the same simulation test are presented. In comparison with the HFPC-EE, a 95.2% reduction in the computation time on account of a 2.37% increase in the cost function is obtained with the HFPC-GA. Comparing the results with the HFPC-BB, a 59.6% reduction in the computation time brings only a 2.03% increase in the cost function. By limiting the number of computations via the selection of the numbers of individuals and generations, we still achieve near optimal tracking results on account of a considerable reduction in the computational load.

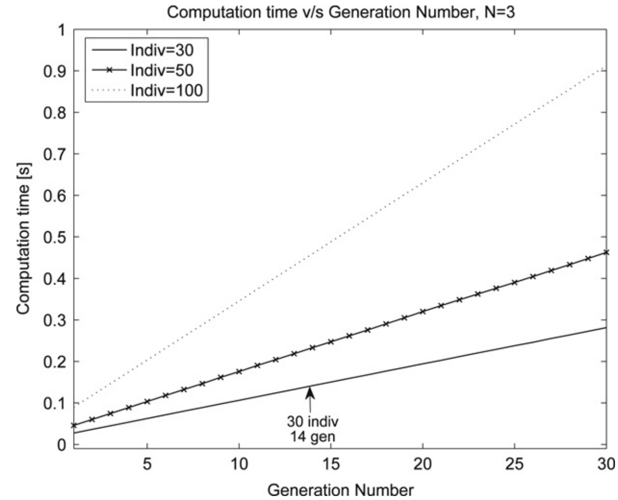

Fig. 11. Evolution of the computation time.

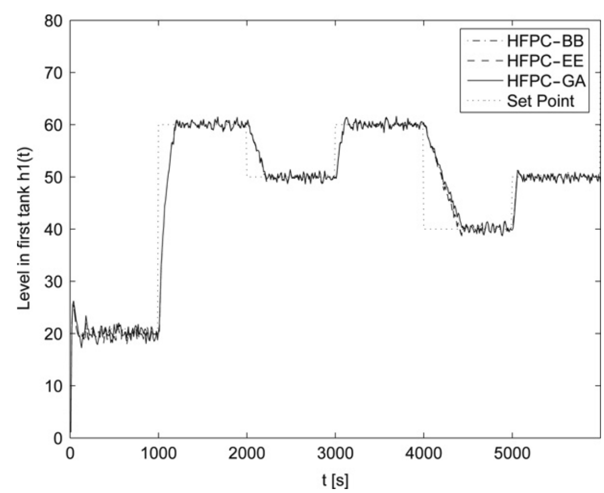

Fig. 12. Simulation test. Controlled-variable response.

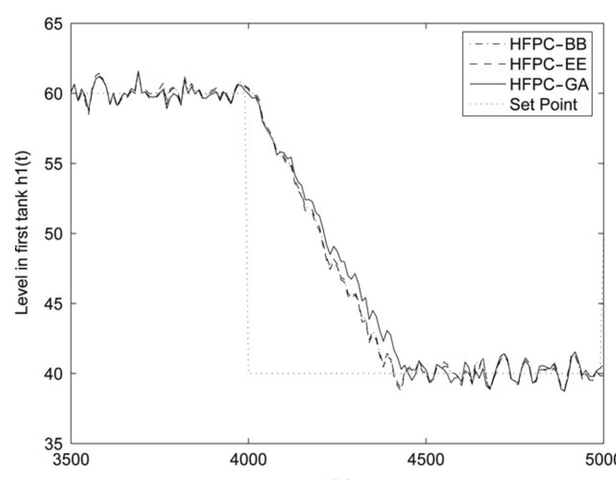

Fig. 13. Simulation test. Manipulated variable.