Direct Adaptive Fuzzy Control for MIMO Processes

Edy Bertolissi, Antoine Duchâteau, Hugues Bersini, Frank Vanden Berghen
IRIDIA, Université Libre de Bruxelles
50, av. Franklin Roosevelt
1050 Brussels, Belgium
eberto@iridia.ulb.ac.be, {aduchate, bersini, fvandenb}@ulb.ac.be

Abstract—**Fuzzy controllers are now part of the common control tools, and several different approaches for their design have been proposed in the literature. This paper presents a direct adaptive fuzzy controller for unknown nonlinear systems. Starting from Single Step Ahead Direct Adaptive Control approach, the Multiple Steps Ahead Direct Adaptive Control scheme is presented. These approaches have been both developed at IRIDIA in the framework of the FAMIMO Esprit project. The strength of the second method is to overcome the problems related with the first approach, such as its inability to control non minimum phase systems, at the price of increased computational load, and a deeper knowledge of the model of the system.**

I. Introduction

Fuzzy control includes a variety of different approaches for solving control problems. In the case of rule-based control, for example, the controller is defined by means of linguistic readable rules. Piecewise linear control defines a set of local linear controllers, acting in different zones of the state-space, which are smoothly connected to produce the desired response. Model-based control, instead, defines a fuzzy model which can be tuned on the basis of the data available from the process. One common denominator in all these visions of fuzzy control is the inherent nature of fuzzy models, a soft combination of piecewise simple (generally linear) models.

Since it has been proven that fuzzy models can be used as universal approximators, they can be successfully used as tunable black-boxes either in an identification or a control context. DAFC [1] (Direct Adaptive Fuzzy Control) is closely related to neurocontrol [2], [3] since the neural control box has been substituted with a fuzzy one. As in the case of neurocontrol, the adaptive algorithm is gradient-based, and allows an on-line continuous adaptation of the controller. In previous papers however [1], it had been shown why the particular structure of fuzzy models makes them better suited to be integrated in this closed-loop gradient-based algorithm. For instance, when restricting the parameter tuning to the consequent part of the fuzzy rules, which are linear respect the parameters to adapt, it had been possible to apply the Lyapunov stability theory in order to prove the asymptotic stability of the overall system (the convergence of the tracking error to zero) [3].

Another important aspect of DAFC is that in some simple versions, like in the case of the single step ahead control, it has been shown that prior knowledge of the process, which can be limited to the Jacobian matrix, can remain very imprecise [2].

The limitations inherent to any control algorithm which restricts its future to just one single step ahead are well known in the control community. Any non minimum phase systems becomes impossible to control by relying on a single step anticipation. The main part of the paper will present the extension of the single step ahead DAFC methodology to adapt the controller on the basis of a multiple steps ahead prediction. Each time an adaptation of the controller is performed, a simulation of the evolution of the states is computed and is used in order to predict the effect changes of the controller parameters. In contrast with the single step ahead version, the algorithm now needs an accurate multiple steps ahead simulation which consequently demands a much better prior knowledge of the process. This new work has been inspired by work previously done by the authors on adaptive fuzzy controller for state-feedback optimal control [4] and on the model based predictive control scheme developed by Babuska [5].

II. System description

Figure 1 shows the structure of a closed loop control system which is composed by:

- a plant to be controlled;
- a model of the plant;
- a predictor;
- a fuzzy controller;

Assume that the n inputs, m outputs plant is expressed in terms of its input-output representation by:

$$y_i(k+1) = F_i\big(\mathbf{y}(k),\ldots,\mathbf{y}(k-p_y+1), \mathbf{u}(k),\ldots,\mathbf{u}(k-p_u+1)\big), \quad i=1,\ldots,m \quad (1)$$

where $y_i(k)$ is the i^{th} output of the plant at time k, $\mathbf{u}(k)$ is the input vector $[u_1(k), u_2(k), \ldots, u_n(k)]^T$ and $\mathbf{y}(k)$ is the output vector $[y_1(k), y_2(k), \ldots, y_m(k)]^T$, F_i is an unknown nonlinear function, and p_y and p_u are the known structure orders of the system for the output i.

If the regressor vector ϕ_k is defined as follows:

$$\phi_k = [\mathbf{y}(k), .., \mathbf{y}(k-p_y+1), \mathbf{u}(k-1), ..., \mathbf{u}(k-p_u+1)] \quad (2)$$

equation (1) can be rewritten as:

$$y_i(k+1) = F_i[\phi_k, \mathbf{u}(k)], i=1, ..., m \quad (3)$$

which in vector notation becomes:

$$\mathbf{y}(k) = \mathbf{F}[\phi_k, \mathbf{u}(k)] \quad (4)$$

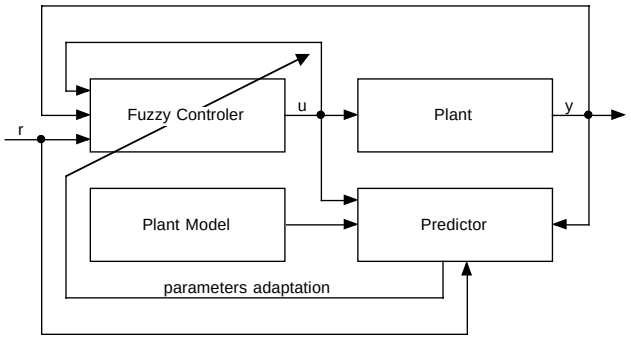

Fig. 1. Schematic representation of the DAFC system

The controller may be any kind of fuzzy system and its output $\mathbf{u}(k)$ is fed into the plant. It can be defined as follows:

$$\mathbf{u}(k) = \mathbf{u}_k = \mathcal{F}(\mathbf{r}(k), \psi_k; \mathbf{w}) \tag{5}$$

where ψ_k is a regressor similar to ϕ_k, $\mathbf{w}$ is the set of parameters describing the fuzzy controller and $\mathbf{r}(k)$ is the vector of the reference signals $[r_1(k), r_2(k), ..., r_m(k)]^T$. The controller will produce a set of control actions $\mathbf{u}(k)$ which will drive the plant outputs $\mathbf{y}(k+1)$ at the values specified by the vector **r**(k). Since the dynamics of the controller is much faster than the one of the plant to be controlled it is supposed that the inputs at time k immediately influence the outputs of the controller.

The model of the plant provides the predictor with the parametric description of the plant to be controlled. The predictor performs a forward simulation of the system composed by the plant and the controller. Each predicted output of the plant is equal to:

$$\hat{y}_i(k+1) = \hat{F}_i[\phi_k, \mathbf{u}(k)], i = 1, ..., m \tag{6}$$

where $\hat{F}_i$ is the estimate of F_i, based on the parametric description provided by the plant model. This information is then used to perform the parametric adaptation of the controller.

III. Single Step Ahead Adaptive Control

The single step ahead fuzzy adaptive controller provides at each time step k the n inputs $u_i(k)$ of the plant which minimize a chosen error criterion. Some assumptions about the controlled process and the fuzzy controller are necessary [3].
1. the form of the expression 1 results from the fact that, at any time k, it is possible to reconstruct the state of the process from the observations of the last $p_y - 1$ plant outputs and the last $p_u - 1$ plant inputs.
2. we assume that, whenever the state of the process in the operating region, there exists a uniquely defined control vector $\mathbf{u}^\star(k)$ realizable, depending on this state, that allows to reach the desired targets $\mathbf{r}(k+1)$ at the time $(k+1)$.
3. the plant process is inverse stable. This and the previous assumption mean that there exists a unique asymptotically stable control law that allows perfect tracking of the signal $\mathbf{r}(k+1)$.
4. the fuzzy controller $\mathcal{F}(\mathbf{r}(k), \psi_k; \mathbf{w})$ can approximate the control law $\mathbf{u}(k)$ to any degree of accuracy in the region of interest, for some "perfectly tuned" weights $\mathbf{w} = \mathbf{w}^\star$;
5. the speed of adaptation of the weights is low in order to be able to separate in the measurement of the error the effects of the parameters adjustment from the input signal variations [6];
6. Every $\frac{\partial F(...)}{\partial \mathbf{u_t}}$, for $t < k$ is equal to 0 since past actions are considered constant;
These assumptions allow us to design a learning algorithm, based on a gradient descent, which can be used to train the weights of the fuzzy controller [7], [8]. Let's select the following error criterion:

$$J = \frac{1}{2}(\mathbf{r}_{k+1} - \hat{\mathbf{y}}_{k+1})^T Q(\mathbf{r}_{k+1} - \hat{\mathbf{y}}_{k+1}) + \frac{1}{2}\mathbf{u}_k^T R \mathbf{u}_k \tag{7}$$

where $\mathbf{r}_{k+1}$ are the values of the setpoints at time $k+1$, $\hat{\mathbf{y}}_{k+1}$ are the predicted outputs of the plant calculated using its parametric model, and Q and R are square matrixes which weight the importance of the different inputs and outputs in the computation of the error criterion.

At each time step k the fuzzy controller will try to reduce the error J on-line by a achieving a gradient descent in the weight space:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \frac{\partial J}{\partial \mathbf{w}} \tag{8}$$

where $\eta > 0$ is the learning rate. Differentiating J with respect to the weights $\mathbf{w}$ we obtain:

$$\frac{\partial J}{\partial \mathbf{w}} = (\mathbf{r}_{k+1} - \hat{\mathbf{y}}_{k+1})^T Q \frac{\partial \hat{\mathbf{y}}_{k+1}}{\partial \mathbf{w}} + \mathbf{u}_k^T R \frac{\partial \mathbf{u}_k}{\partial \mathbf{w}} \tag{9}$$

where $\partial \hat{\mathbf{y}}_{k+1}/\partial \mathbf{w}$ is defined as a matrix having elements $[\partial \hat{\mathbf{y}}(k+1)/\partial \mathbf{w}]_{ij} = \partial \hat{y}_i(k+1)/\partial w_j$. Substituting this expression in equation 8 we obtain:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \left((\mathbf{r}_{k+1} - \hat{\mathbf{y}}_{k+1})^T Q \frac{\partial \hat{\mathbf{y}}_{k+1}}{\partial \mathbf{w}} + \mathbf{u}_k^T R \frac{\partial \mathbf{u}_k}{\partial \mathbf{w}} \right) \tag{10}$$

Considering the state relations expressed in 4 and 5 it is possible to write [3]:

$$\frac{\partial \hat{\mathbf{y}}_{k+1}}{\partial \mathbf{w}} = \frac{\partial \hat{\mathbf{F}}(\phi_k, \mathbf{u}_k)}{\partial \mathbf{w}} = \frac{\partial \hat{\mathbf{F}}(\phi_k, \mathbf{u}_k)}{\partial \mathbf{u}_k} \frac{\partial \mathbf{u}_k}{\partial \mathbf{w}} = \frac{\partial \hat{\mathbf{y}}_{k+1}}{\partial \mathbf{u}_k} \frac{\partial \mathbf{u}_k}{\partial \mathbf{w}} \tag{11}$$

$$\frac{\partial \mathbf{u}_k}{\partial \mathbf{w}} = \frac{\partial \mathcal{F}(\mathbf{r}_k, \psi_k; \mathbf{w})}{\partial \mathbf{w}} \tag{12}$$

and therefore equation 10 becomes:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \left((\mathbf{r}_{k+1} - \hat{\mathbf{y}}_{k+1})^T Q \left(\frac{\partial \hat{\mathbf{y}}_{k+1}}{\partial \mathbf{u}_k} \frac{\partial \mathcal{F}_k}{\partial \mathbf{w}} \right) + \mathbf{u}_k^T R_k \frac{\partial \mathbf{u}_k}{\partial \mathbf{w}} \right) \tag{13}$$

which is a simple adaptation rule, and can be used on-line for tuning the parameters of the controller.

IV. Multiple Steps Ahead Adaptive Control

The algorithm described in the previous section can be improved using the approach used in generalized predictive control theory [9], where the design procedure considers future values of the control signals as well as the present ones. Using this approach the future values of the setpoints and the system outputs are needed to calculate the weight update rule.

If the structure of the system depicted in figure 1 is maintained, it is possible to design another learning algorithm, based on the same principles, which performs an adaptation of the weights of the fuzzy controller using information about the future behaviour of the system. In this case the predictor will provide the fuzzy controller with information regarding the futures values of the $\hat{\mathbf{y}}$ and $\mathbf{u}$ up to the prediction horizon. In this case it is necessary to make the following assumptions:

1. the state of the process at any time can be reconstructed using the information inside the regressor ϕ_k;
2. it exists a unique series of inputs $[\mathbf{u}_k\, \mathbf{u}_{k+1} \ldots \mathbf{u}_{k+H_c}]$ which lead the output signals $\mathbf{y}$ towards $\mathbf{r}$ at time $k+H_p$. H_c is the control horizon (the length of the time horizon where a control signal is applied) H_p is the prediction horizon (the length of the time horizon where the future states of system are simulated). The prediction horizon must be bigger or equal to the control horizon ($H_p \geq H_c$). Control actions are considered constant once the control horizon is reached ($\mathbf{u}_t = \mathbf{u}_{k+H_c}$ for $t > k+H_c$);
3. Every $\frac{\partial \hat{F}(\ldots)}{\partial \mathbf{u_t}}$, for $t < k$ is equal to 0 since past actions are considered constant;
4. The fuzzy controller $\mathcal{F}(\mathbf{r}(k), \psi_k; \mathbf{w})$ can approximate the series of perfect control actions $[\mathbf{u}_k \mathbf{u}_{k+1} \ldots \mathbf{u}_{k+H_c}]$ to any degree of accuracy in the region of interest for some "perfectly tuned" weights $\mathbf{w} = \mathbf{w}^\star$;
5. the speed of adaptation of the weights is low in order to be able to separate in the measurement of the error the effects of the parameters adjustment from the input signal variations [6].

These assumptions allow the design of an adaptation algorithm based on the gradient descent approach defined in equation 8.

In order to train the parameters $\mathbf{w}$ the following cost function is selected:

$$J = \frac{1}{2} \sum_{t=k}^{k+H_p} (\mathbf{r}_t - \hat{\mathbf{y}}_t)^T Q_t (\mathbf{r}_t - \hat{\mathbf{y}}_t) + \frac{1}{2} \sum_{t=k-1}^{k+H_c} \Delta \mathbf{u}_t^T R_t \Delta \mathbf{u}_t \quad (14)$$

where $\Delta \mathbf{u}_t = \mathbf{u}_t - \mathbf{u}_{t-1}$ and $\mathbf{u}_{k-2} = 0$. The matrix $Q \in \Re^{n \times n \times H_p}$ weights the errors $(\mathbf{r}_t - \hat{\mathbf{y}}_t)$ while the matrix $R \in \Re^{n \times n \times H_c}$ has the effect to penalize the large variations of $\Delta \mathbf{u}_t$ which could destabilize the system.
Substituting this expression in equation 8 and recalling equation 4 it is possible to obtain:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \left(\sum_{t=k}^{k+H_p} (\mathbf{r}_t - \hat{\mathbf{y}}_t)^T Q_t \frac{\partial \hat{\mathbf{y}}_t}{\partial \mathbf{w}} + \sum_{t=k-1}^{k+H_c} \Delta \mathbf{u}_t^T R_t \frac{\partial \Delta \mathbf{u}_t}{\partial \mathbf{w}} \right) \quad (15)$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \left(\sum_{t=k}^{k+H_p} (\mathbf{r}_t - \hat{\mathbf{y}}_t)^T Q_t \frac{\partial \hat{\mathbf{F}}(\mathbf{u}_{t-1}, \phi_{t-1})}{\partial \mathbf{w}} + \sum_{t=k-1}^{k+H_c} \Delta \mathbf{u}_t^T R_t \frac{\partial \Delta \mathbf{u}_t}{\partial \mathbf{w}} \right) \quad (16)$$

The central part of this expression can be expanded as follows:

$$\frac{\partial \hat{\mathbf{F}}(\mathbf{u}_{t-1}, \phi_{t-1})}{\partial \mathbf{w}} = [\frac{\partial \hat{F}_1(\mathbf{u}_{t-1}, \phi_{t-1})}{\partial \mathbf{w}} \ldots \frac{\partial \hat{F}_m(\mathbf{u}_{t-1}, \phi_{t-1})}{\partial \mathbf{w}}]^T \quad (17)$$

Each element of the previous vector can be expanded into its partial derivatives:

$$\frac{\partial \hat{F}_i(\mathbf{u}_{t-1}, \phi_{t-1})}{\partial \mathbf{w}} = \frac{\partial \hat{F}_i(\ldots)}{\partial \phi_{t-1}} \frac{\partial \phi_{t-1}}{\partial \mathbf{w}} + \frac{\partial \hat{F}_i(\ldots)}{\partial \mathbf{u}_{t-1}} \frac{\partial \mathbf{u}_{t-1}}{\partial \mathbf{w}} \quad (18)$$

however from 5 it is possible to write:

$$\frac{\partial \mathbf{u}_{t-1}}{\partial \mathbf{w}} = \frac{\partial \mathcal{F}(\mathbf{r}_{t-1}, \psi_{t-1}; \mathbf{w})}{\partial \mathbf{w}} = \frac{\partial \mathcal{F}(\ldots)}{\partial \mathbf{w}} + \frac{\partial \mathcal{F}(\ldots)}{\partial \psi_t} \frac{\partial \psi_t}{\partial \mathbf{w}} \quad (19)$$

which substituted in equation 18 gives:

$$\frac{\partial \hat{F}_i(\mathbf{u}_{t-1}, \phi_{t-1})}{\partial \mathbf{w}} = \frac{\partial \hat{F}_i(\ldots)}{\partial \phi_{t-1}} \frac{\partial \phi_{t-1}}{\partial \mathbf{w}} + \frac{\partial \hat{F}_i(\ldots)}{\partial \mathbf{u}_{t-1}} \left(\frac{\partial \mathcal{F}(\ldots)}{\partial \mathbf{w}} + \frac{\partial \mathcal{F}(\ldots)}{\partial \psi_{t-1}} \frac{\partial \psi_{t-1}}{\partial \mathbf{w}} \right) \quad (20)$$

Equations 19 and 20 need to be recursively computed in order to calculate the solution of equation 16.

At the first step, due to assumption that past actions are considered constant, the partial derivatives of the regressors respect to the weights $\mathbf{w}$ are all equal to zero. Therefore at time $t = k$:

$$\frac{\partial \mathbf{u}_{k-1}}{\partial \mathbf{w}} = \frac{\partial \mathcal{F}(\mathbf{r}_{k-1}, \psi_{k-1}; \mathbf{w})}{\partial \mathbf{w}} \quad (21)$$

$$\frac{\partial \hat{\mathbf{y}}_k}{\partial \mathbf{w}} = \frac{\partial \hat{\mathbf{F}}(\mathbf{u}_{t-1}, \phi_{k-1})}{\partial \mathbf{w}} = \frac{\partial \hat{\mathbf{F}}(\ldots)}{\partial \mathbf{u}_{k-1}} \frac{\partial \mathcal{F}(\ldots)}{\partial \mathbf{w}} = \frac{\partial \hat{\mathbf{y}}_k}{\partial \mathbf{u}_{k-1}} \frac{\partial \mathcal{F}(\ldots)}{\partial \mathbf{w}} \quad (22)$$

It is possible to note that equation 22 has the same form of the one used in the single step approach presented in equation 13.

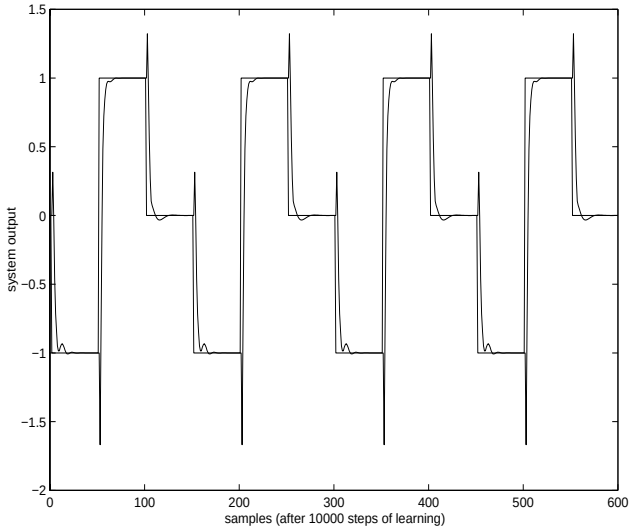

Fig. 2. Performance of the multiple step ahead DAFC on a non minimum phase system after 10000 steps of learning

Then it is necessary to compute the other values of $\frac{\partial \mathbf{y}_t}{\partial \mathbf{w}}$ and $\frac{\partial \mathbf{u}_t}{\partial \mathbf{w}}$ up to the prediction horizon. The values calculated at each step are used for the computation of the derivatives of the regressors respect to the weights since:

$$\frac{\partial \phi_{t'}}{\partial \mathbf{w}} = \frac{\partial \mathbf{y}(t')}{\partial \mathbf{w}} + ... + \frac{\partial \mathbf{y}(t'-p_y+1)}{\partial \mathbf{w}} + \frac{\partial \mathbf{u}(t'-1)}{\partial \mathbf{w}} + ... + \frac{\partial \mathbf{u}(t'-p_u+1)}{\partial \mathbf{w}} \quad (23)$$

where and all the terms $\frac{\partial \mathbf{y}(t_i)}{\partial \mathbf{w}}$ and $\frac{\partial \mathbf{u}(t_i)}{\partial \mathbf{w}}$ are equal to zero for $t_i < k+1$.

The multiple step algorithm is a much more complicated algorithm in comparison with the single step version but it allows to alleviate two strong assumptions: the system does not need anymore to be minimum phase and any desired output does not need to be reachable in one step. This comes from the fact that instead of looking one step in the future in order to select a control policy, we consider a longer time horizon. It implies that even if the system starts in the "wrong direction" (non minimum phase behaviour), the learning algorithm will have the time to "realise" that it changes direction and adapt its behaviour. It also implies that the learning algorithm can now learn on the basis of a series of control actions instead of only one control action in order to reach the desired system output.

The relaxation of these assumption has two costs: a higher computation load and the need for a more precise model. The first cost is easily understandable and is due to the simulation at each step of the closed loop behaviour over a long time horizon. The second cost comes from the fact that the long term predictions are more difficult to achieve and need better precision, at each step, in order to avoid errors accumulation.

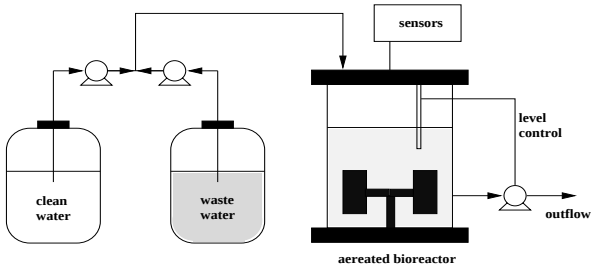

Fig. 3. Schematic representation of the aerated bioreactor for waste-water treatment

V. Simulation studies

The two DAFC approached have been tested on two different systems. One toy problem, and a waste water treatment plant, a benchmark which has been used by all the partners taking part in the FAMIMO project (ESPRIT LTR Project 21911) [10].

A. The toy problem

Multiple Steps Ahead method has been tested on the following, simple nonlinear, non minimum phase system.

$$y_{k+1} = y_k + sin(3 * u_{k-1}) - 3u_k \quad (24)$$

It is impossible to control this system with the single step ahead method, and therefore we had to use the multiple steps ahead procedure. The plant was modeled by a Takagi-Sugeno fuzzy system. It had been identified using 5000 points collected by exciting the plant with a pseudo random sinusoidal input, by means of the Gustafson-Kessel algorithm [11] for calculating the centers and the linears of the Takagi-Sugeno system. Figure 2 shows the result of the reference signal and the output of the plant obtained using a controller made of 9 fuzzy Takagi-Sugeno rules whose consequents have been initialized to 0 with the learning rate $\eta = 0.0005$. At each time step, a 5 steps ahead prediction of the system was performed and used to optimize the parameters of the linears consequents of the controller.

In this example it is interesting to notice the typical pattern present in the non minimum phase systems. At each set point change, the system starts in the opposite direction with respect to the current value of the reference signal. This system, which is impossible to control using the one step ahead method, is easily controlled using the multiple steps ahead one.

B. The waste water process problem

A biological system, which consists of a continuous flow aerated bioreactor for waste water treatment in pulp and paper industry, has been the target of our second experimental study. The bioreactor contains a mixed microbial population growing in a blending of two types of substrates, an energetic and a xenobiotic one. The first one is easily biodegradable, while the second one, is less degradable. This makes

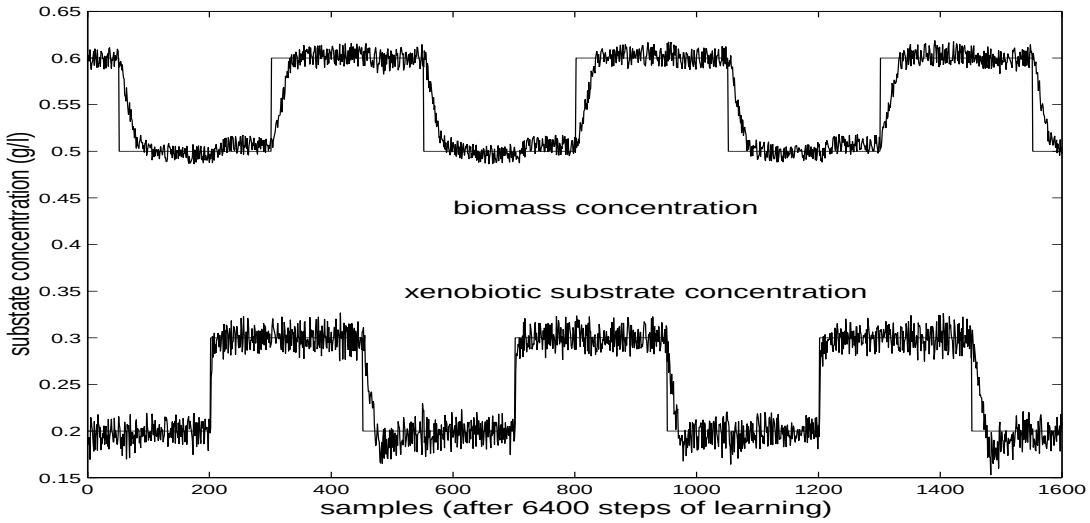

Fig. 4. Performance of the 3 steps ahead adaptive controller on the waste water processing plant after 6400 learning steps

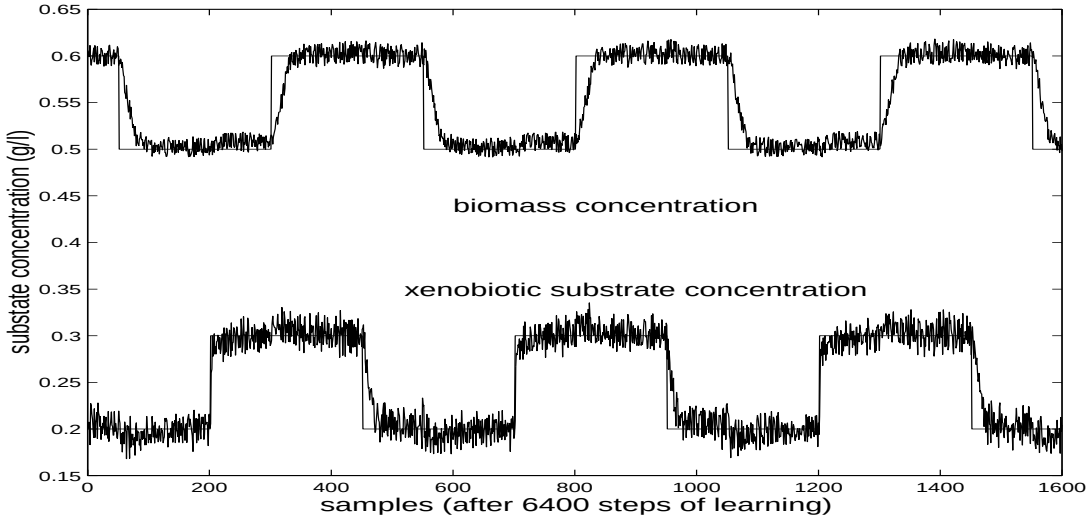

Fig. 5. Performance of the 10 steps ahead adaptive controller on the waste water processing plant after 6400 learning steps

the xenobiotic substrate as the main pollutant in the waste water, since the energetic one can be easily degraded by the micro-organisms present in the bioreactor. The energetic substrate and the biomass concentration are the measurable outputs signals, while the xenobiotic concentration is not available on-line, and must be reconstructed through an observer. The waste-water is continuously fed into the reactor, and the volume of the water is kept constant, thus the inflow is equal to the outflow. It is possible to control the concentration of the dilution rate and the concentration values of the two substrates, by mixing two flows, one from a clean water tank and one form a waste-water tank.

The waste water plant was modeled by a Takagi-Sugeno fuzzy system. It had been identified using 5000 points collected by exciting the plant with a pseudo random sinusoidal input, by means of the Gustafson-Kessel algorithm [11] for calculating the centers and the linears of the Takagi-Sugeno system. The adaptation algorithm had been used on the system with three different steps horizons, 1, 3, and 10 steps ahead predictions. The results of the 1 step ahead adaptive controllers were quite poor, while the graphs reporting the performance of the system in controlling the biomass concentration and the xenobiotic substrate concentration for the other configurations are reported in figure 4 and 5. The controller uses four Takagi-Sugeno fuzzy rules for controlling the biomass concentration and two for the xenobiotic substrate concentration. All the consequents of the rules had been initialized to 0 at the start of the experience. The learning rate for the 3 steps ahead controller is equal to 0.05, while it was set to 0.0050 for the 10 steps ahead controller. Only the the parameters of the linears consequents of rules of the controller had been optimized during the adaptation. The outputs of the plant are affected with a random square noise of amplitude between -2% and $+2\%$ of the output value. Parametrics variation of the biological ki-

netic parameters which regulate the waste water plant are also included in the system to take into the account the imprecise knowledge of the dynamics of the system.

It can be seen on the figures 5 and 4 that after a long enough time of learning (6400 steps), both systems achieve very good control results under noisy conditions. The main difference between the two algorithms appears in the time needed to learn the control policy. Since the 10 steps ahead based adaptation method looks further at each step, it is able to learn faster than the 3 steps ahead one. However, it also needs more computation what implies that if the learning is faster in real time (it needs less time steps to learn) it can be slower in simulation time (the time of the simulation can be longer).

VI. Conclusions

The adaptive algorithms presented in this paper are very close to algorithms implemented on neural controllers, and as a matter of fact can be easily grafted on any type of black-box "parametrized" controllers whatever fuzzy or not. Nevertheless, fuzzy models, due to their inherent decomposition of any non-linear functional mapping into a set of local linear maps softly composed, appear like en ideal candidate for acting as the black-box. These adaptive algorithms could work with any initial randomized fuzzy models but it is clear that, like for any optimization application, the better the initial controller to be adapted will be, the faster and the better the adaptation will work. This strongly justifies the need to use some well-designed controllers (designed on the basis of some prior knowledge of the process or available data of the process) to be initially installed in the controller box even if they will still be subject to some later on-line adaptation.

Even if the controller is optimally designed from the very beginning, some on-line adaptation can still be welcome, for instance in response to unexpected disturbances or non stationarities in the process. Provided that the very qualitative knowledge needed by DAFC still applies after this incidental change in the process behavior, the controller could be able to rapidly absorb the perturbation.

The preliminary results demonstrate the feasibility of the multiple steps ahead DAFC even if some important issues have still to be treated. We would like to explicitly integrate information like the saturation of the actuators into the learning process in order to improve and accelerate it. Another problem that needs to be overcome is the sensibility of the learning process to the quality of the plant model. Even if the performance of the controller will always depend strongly on it, we think that the learning process could be made more robust to a bad model.

Acknowledgment

This research has been supported by the European FAMIMO project (LTR Project 21911 Reactive Scheme, Task 4.2) and by a Marie Curie Fellowship awarded to Edy Bertolissi (CEC-TMR Contract No. ERBFMBICT972495)

References

[1] J. M. Renders, M. Saerens, and H. Bersini, "On the stability of direct adaptive fuzzy controllers," Tech. Rep., IRIDIA/ULB, Bruxelles, Belgium, 1994.

[2] J. M. Renders, M. Saerens, and H. Bersini, "Neurocontrol based on the backpropagation algorithm," in *Intelligent Control Systems*, M. M. Gupta and N. K. Sinha, Eds., pp. 292–326. IEEE Press, 1996.

[3] J. M. Renders, M. Saerens, and H. Bersini, "Adaptive neurocontrol of a certain class of MIMO discrete-time processes based on stability theory," in *Neural Network Engineering in Dynamic Control Systems*, K. J. Hunt, G. R. Irvin, and K. Warwick, Eds., pp. 43–60. Springer Verlang, 1995.

[4] H. Bersini, "Adaptive fuzzy control for the state feedback optimal control," in *Proceedings of the XIIIth IFAC World Congress*, San Francisco, July 1996.

[5] R. Bakuŝka, *Fuzzy Modeling and Identification*, Ph.D. thesis, TU Delft, Delft, Holland, 1997.

[6] Y. Landau, *Adaptive Control. The Model Reference Approach*, Marcel Dekker, 1979.

[7] K. S. Narendra and K. Parthsarathy, "Identification and control for dynamic systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.

[8] K. S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems," *Int. Journal of Approximate Reasoning*, vol. 6, pp. 109–131, 1992.

[9] D. W. Clarke, C. Mohtadi, and P. S. Tuffs, "General predictive control - part 1. The basic algorithm," *Automatica*, vol. 23, no. 2, pp. 137–148, 1987.

[10] C. Ben Youssef, *Filtrage, estimation et commande adaptive d'un procédé de traitement des eaux usuées*, Ph.D. thesis, Polytechnic National Institute, Tolouse, France, 1996.

[11] D. E. Gustafson and W. C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," in *Proc. IEEE CDC*, San Diego, CA,USA, 1979, pp. 761–776.