PSO-Based Fuzzy Predictive Control

Oussama Ait Sahed [1], Kamel Kara [1] & Mohamed Laid Hadjili [2]
Department of electronics, Lab. SET, Saad Dahleb University , Blida, Algeria
High School of Computer Sciences (ESI), rue royale 67, 1000 Bruxelles, Belgium.
aitsahed.oussama@hotmail.com, kamel.kara@ieee.org, mhadjili@heb.be

***Abstract* – In this work, a modified particle swarm optimization (PSO) algorithm is proposed to solve the optimization problem of fuzzy predictive control. The main objective is to obtain an efficient control algorithm with a low computational burden and a good control performance. This is done essentially by reducing both the PSO population and the number of iterations needed for each sampling time.**

The control accuracy and computational efficiency of the proposed algorithm are illustrated by considering the control of the continuous stirred tank reactor.

I. INTRODUCTION

Model predictive control (MPC) is a popular control strategy which has been successfully used in many practical applications; such as refining, petrochemical, chemical, and food processing industries [1], [2]. In the last years, model predictive control was also introduced in speed and positioning control of electrical machines and control of robotic manipulators [1]. MPC techniques are able to handle complex control problems which involve multivariable process interactions, constraints in the system variables, variable or unknown time delays, etc.

Model predictive control algorithms use an explicit model to predict the future values of the system response and then derive the control law by minimizing a given cost function. This function usually takes the form of a quadratic function of the errors between the predicted values and the reference trajectory, and includes in most cases the control effort. The model choice is a critical task; the controller efficiency and performance extremely depend on the accuracy of the used model. Classical MPC algorithms, which are based on linear models, are likely not to work properly in the case of nonlinear systems and don't give satisfactory performance [3], and since the most practical processes are nonlinear by nature, new efficient Nonlinear MPC (NMPC) techniques have to be derived to ensure higher control performance. One of the promising ways to obtain an efficient nonlinear controller is to use fuzzy models. In fact, fuzzy models have received particular attention in the area of nonlinear modeling, especially the Takagi-Sugeno (TS) fuzzy models, due to their simplicity and capability to approximate any nonlinear behavior [4], [5]. However obtaining an efficient solution to the related complex optimization problem is not without difficulties; the computational time has to ensure the real time feasibility of the control algorithm and a good accuracy of the on line solutions must be achieved.

An analytical solution to any optimization problem is the most preferred, but when dealing with complex nonlinear optimization problems, this solution becomes unfeasible. Several approaches were proposed in the literature to solve the nonlinear predictive control problem. Some of them are based on numerical optimization algorithms, which have a slow convergence rate and could easily be trapped in local minima. Some others are based on meta-heuristic optimization algorithms such as the genetic algorithms (GA) which are considered to be one of the powerful optimization algorithms. Their main drawback is their high computationally complexity which makes them very hard to be implemented in real time. Particle Swarm Optimization (PSO) algorithms are another powerful meta-heuristic based algorithms that require much less computationally effort than the GA, and are relatively easy to implement. Indeed, many papers have presented different PSO based approaches to solve the optimization problem found in nonlinear predictive control strategy [6]-[10]. The common point of these PSO-based algorithms is the important number of particles and iterations needed to find a solution; this requires more computational effort.

In this paper, a fuzzy predictive control algorithm using particle swarm optimization technique is proposed. In order to ensure the real time feasibility of this algorithm a modified PSO algorithm that uses a small number of particles and iterations was used. To highlight the algorithm efficiency and performance, the control of a highly nonlinear process, a Continuous Stirred Thank Reactor (CSTR), is considered.

The outline of the paper is as follow: Section 2 gives the design stages of Takagi-Sugeno fuzzy models whereas section 3 presents the general fuzzy predictive control formulation. The canonical PSO algorithm and the proposed control algorithm are given in sections 4 and 5, respectively. Efficiency and control performance of the proposed algorithm are analyzed and tested by considering the CSTR control in Section 6. Finally, conclusions are drawn in Section 7.

II. TAKAGI-SUGENO FUZZY MODELING

Fuzzy inference systems (FIS) are universal approximators capable of approximating any continuous function with certain level of accuracy, this important property led to the appearing of fuzzy modeling, where a given process can be represented using a FIS. Another important property is the ability to incorporate human knowledge into this FIS using linguistic data.

Different types of fuzzy models have been proposed in the literature such as the Mamdani models where both the

978-1-4673-2421-2/12/$31.00 ©2012 IEEE

antecedents and the consequences are fuzzy propositions, whereas, the Takagi-Sugeno (TS) fuzzy models have fuzzy proposition only in their antecedents while their consequences are linear functions of the antecedents or other variables [5]. The TS models, based on the IF-THEN rule, are described by a set of *r* fuzzy rules as follows [11]:

$$R_j: \quad \begin{matrix} \text{IF } \mathcal{F} \text{ is } A_j \\ \text{then } \hat{y}_j(t+1)=\theta_j^T \mathcal{X} \end{matrix} \tag{1}$$

where:

$\mathcal{F} = \{ y(t),\ y(t-1) \ \dots\ y(t-n_a+1),\ u(t),\ u(t-1) \ \dots\ u(t-m_a+1)\}$

is the set of n_a+ m_a premise values,

$\mathcal{X}^T = \{ y(t), y(t-1) \dots\ y(t-n_r+1), u(t), u(t-1) \dots u(t-m_r+1), 1\}$

is the set of m_r inputs and n_r outputs values used in the consequent regressors,

$A_j = \{ A_{1,j}, A_{2,j} \dots\ A_{n_a,j}, B_{1,j}, B_{2,j} \dots\ B_{m_a,j} \}$

is the set of membership functions associated to the j^{th} rule and

$\theta_j^T = \{\ a_{j,1}, a_{j,2} \dots\ a_{j,n_r}, b_{j,1}, b_{j,2} \dots\ b_{j,m_r}, c_j \}$

is the parameter vector of the j^{th} sub-model.

By defining the parameter matrix as:

$$\Theta = [\theta_1, \theta_2\ \dots\ \theta_r], \tag{2}$$

and the normalized membership grade vector as :

$$\mu^T = [\mu_1, \mu_2 \dots\ \mu_r], \tag{3}$$

The inferred output of the TS fuzzy model is given by:

$$\hat{y}(t+1) = (\Theta\, \mu)^{\mathrm{T}}\, \chi \tag{4}$$

Using the input/output representation, the TS model given by (4) can be rewritten as follows:

$$\hat{y}(t+1) = \sum_{p=1}^{n_a} \bar{a}_p(t) y(t-p+1) + \sum_{p=1}^{m_a} \bar{b}_p(t) u(t-p+1) + \bar{c}(t) \tag{5}$$

where:

$\bar{a}_p(t) = \sum_{l=1}^{r} \mu_l(t) a_{l,p}$, $\bar{b}_p(t) = \sum_{l=1}^{r} \mu_l(t) b_{l,p}$ and $\bar{c}(t) = \sum_{l=1}^{r} \mu_l(t) c_l$.

The global output of the fuzzy model can be written as:

$$\hat{y}(t+1) = \sum_{p=1}^{r} \mu_p(t) y_p(t+1) \tag{6}$$

$$\hat{y}(t+1) = [\mu_1(t) y(t), \dots, \mu_1(t) u(t-m_r+1), \dots, \mu_r(t)] \tilde{\theta} \tag{7}$$

where:

$$\hat{y}(t+1) = \varphi^T(t) \tilde{\theta}$$
$$\tilde{\theta}^T = [\theta_1^T\ \ \theta_2^T \dots\ \theta_r^T]$$

Assuming that a set of *N* input-output data pairs $(u(t), y(t))$ is available and defining $L = max\{m_r, n_r\}$, a regression matrix which has the following expression can be constructed

$$\Phi = \begin{bmatrix} \varphi^T(L) \\ \varphi^T(L+1) \\ \dots \\ \varphi^T(N-1) \end{bmatrix} \tag{8}$$

The vector $\tilde{\theta}$ can be calculated by solving the following least square problem:

$$Y = \Phi \tilde{\theta} \tag{9}$$

where: $Y = [y(L+1), \dots, y(N)]^T$.

In order to solve the problem in (9), the inputs and the membership function parameters (number, position and shape) must be first defined. Several methods exist to fully construct fuzzy models from input-output data such as the Automatic Fuzzy Rule Extractor with Linguistic Integrity (AFRELI) algorithm [5].

III. FUZZY PREDICTIVE CONTROL PRINCIPLE

Several model based predictive control techniques have been proposed during the last years. All these techniques have two common features (Fig. 1):

- A process model is used to predict the future behavior of the process over a finite moving horizon.
- The control input is computed such as to minimize a quadratic criterion.

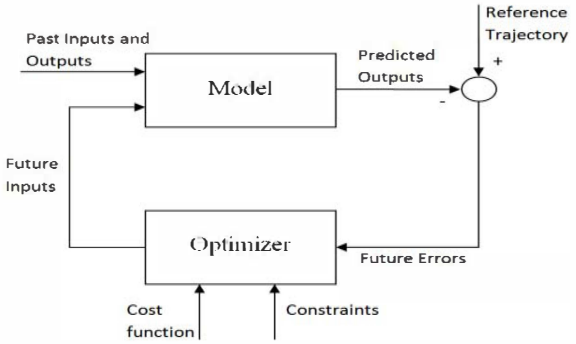

Fig. 1. Predictive control scheme

As in all model based predictive control algorithms, the predictive control using fuzzy models can be described by the following steps:

- For every sampling time, it is necessary to predict the future values of the system response over a certain horizon, using a prediction model.
- A reference trajectory is specified, over at least the prediction horizon.
- A future control sequence over the control horizon is calculated by minimizing a determined objective function. This function usually takes the form of a quadratic

function of the errors between the predicted outputs and the reference trajectory, and includes in most cases the control effort.

- Only the first element of the calculated sequence is applied to the system.
- For the next sampling time the preceding steps are repeated according to the receding horizon idea.

The predictive control problem can be stated as follow:
At each sampling time compute the optimal control sequence that minimizes, over a finite moving horizon called control horizon, a cost function usually of the following form:

$$J = \sum_{j=N_1}^{N_2} [\hat{y}(t+j/t) - w(t+j)]^2 + \lambda \sum_{j=1}^{N_u} \Delta u^2(t+j-1) \quad (10)$$

subject to: $\Delta u(t+j-1)=0$, for $j > N_u$

where $\hat{y}(t+j/t)$ is a j-step ahead prediction of the system output on data up to time t, N_1 and N_2 are the minimum and maximum prediction horizons, N_u is the control horizon, λ is a weighing factor penalizing variations in the control signal and $w(t+j)$ is the future reference trajectory. N_1, N_2, N_u, and λ are the design parameters.
In order to optimize the cost function, the optimal prediction of $y(t+j/t)$ must be obtained for $N_1 \leq j \leq N_2$. Several fuzzy models can be used to design the prediction model according to the following approaches:

- The future values of the process response are recursively generated using the process fuzzy model.
- The future values of the process response are generated using a multi-step fuzzy prediction model.

When using a nonlinear fuzzy model, the predicted response is nonlinear with respect to the control input, and the minimization of the cost function with respect to the control increments will be a nonlinear and non convex optimization problem, the solution of which is difficult and generally expensive in computing time. Some simplified solutions to this problem were proposed such as the fuzzy predictive controller based on the Takagi-Sugeno fuzzy model linearization [12]. Another approach based on the approximation of the free and forced responses of the fuzzy prediction model was developed by Espinosa [5]; in this approach, like in the linear generalized predictive control, the control law is analytically obtained. In reference [11] similar fuzzy generalized predictive controllers based on local, global and multi-step TS fuzzy models were proposed. More interesting solutions of the fuzzy predictive control can be found in [13], [14]. Genetic algorithms were also used to solve the optimization problem in fuzzy predictive control [15]. However, these solutions require higher computational time.

IV. CANONICAL PSO ALGORITHM

The PSO algorithms mimic the social behavior observed in flocks of birds and schools of fishes. They were first introduced by Kennedy and Eberhart in 1995 as a solution for the optimization of a single objective continuous problem. Since then, many features and enhancements were proposed in the literature to address other optimization problems (binary, discrete, constrained, multi-objective, etc.). Every PSO population consists of a group of particles that surf the search space looking for potentials solutions. When moving from one point to another, each particle is influenced by its history, the neighborhood history and some random behavior.

While surfing the search space, each particle movements are bound by these equations:

$$v(m+1) = v(m) + a(m+1) \quad (11)$$

$$x(m+1) = x(m) + v(m+1) \quad (12)$$

where v, a, x and m are velocity, acceleration, position and number of iterations. Taking the Clerc-Kennedy PSO as the canonical PSO, the acceleration of a given particle i will have the expression [16]:

$$a_i = \chi\left[c\epsilon_1(P_g - x_i) + c\epsilon_2(P_i - x_i)\right] - (1-\chi)v_i \quad (13)$$

where $\chi = 0.729843788$, $c = 2.05$, ϵ_1 and ϵ_2 are random number from the uniform distribution $U[0,1]$, x_i current position of particle i, v_i current velocity of particle i, P_i personal best position of particle i and P_g is the global best position of all particles. Remark that a_i could take negative values as well as positive. The parameter χ guaranties a decreasing velocity for each particle as the number of iteration increases. This property will improve the convergence quality as the particles approach the solution, their movement will be more and more limited, which help fine tuning the found solution.

Using the canonical PSO algorithm to solve the optimization problem of fuzzy predictive control cannot provide an efficient solution; it will require a large population and an important iterations number. In the next sections, the modified version of this algorithm is proposed to obtain an efficient fuzzy predictive control algorithm with a low computational time.

V. FUZZY PREDICTIVE CONTROLLER BASED ON THE MODIFIED PSO ALGORITHM

A fuzzy predictor must be constructed to provide the necessary future output needed to solve the optimization problem. From (5) we can write:

$$\hat{y}(t+j) = \sum_{p=1}^{n_a} \bar{a}_p(t) y(t-p+j) + \sum_{p=1}^{m_a} \bar{b}_p(t) u(t-p+j) + c(t) \quad (14)$$

where $N_1 \leq j \leq N_2$.

$\hat{y}(t+j)$ is estimated using measured and estimated past values of the system output.

The modified PSO algorithm is considered to optimize the cost function given by (10). The idea is to modify and adapt the canonical PSO algorithm to increase its efficiency in regard of the execution time and the solution accuracy. Each particle position x_i of the PSO population will represent a possible optimization solution; i.e. a control actions sequence of the form:

$$x_i = \{u(t), u(t+1), \dots, u(t+N_u-1)\} \qquad (15)$$

The implementation of this algorithm is based on the following points:

1) The solution:

$$U_t = \{u_t(t), u_t(t+1), \dots, u_t(t+N_u-1)\},$$

found at the sampling time t, is one of the possible solutions for the next sampling time $t+1$. That is:

$$U_{t+1} = U_t.$$

2) Another probable solution for the sampling time $t+1$ is that given by:

$$U_{t+1} = \{u_t(t+1), u_t(t+2), \dots, u_t(t+N_u-1), u(t+N_u)\}.$$

The last control action is taken such as:

$$u(t+N_u) = u_t(t+N_u-1).$$

3) Prior knowledge of the system is used to initialize the algorithm rather than randomly initializing the particles in the whole search space.

4) Evaluate the fitness of the solutions obtained in 1) and 2). The fittest one will be the global best position (P_g) of the entire population.

5) The particles are not randomly distributed over the whole search space. They are randomly distributed, using a Gaussian distribution, within a limited radius r_d around the global best position (P_g). This radius is adaptively varied and is given, at the sampling time $t+1$, by:

$$r_d = u_t(t+1) - u_t(t) \qquad (16)$$

A minimal value r_{dmin} is imposed to r_d to ensure, to a certain level, particles diversity. Its value can either be fixed or varied depending on the reference trajectory amplitude changing.

6) The cost function given by (10) is taken as the fitness function of the PSO algorithm.

7) The termination criterion is the maximal number of iterations.

The control algorithm can be summarized as follow:

1. Set the MPC design parameter values.
2. Set the initial solution values U_t over the control horizon. Prior knowledge of the system can be used to do this initialization.
3. Specify the desired reference trajectory over the prediction horizon: $w(t+N_1)\dots w(t+N_2)$.
4. Compute the future values of the system output $\hat{y}(t+j)$, $N_1 \le j \le N_2$, using the prediction model.
5. Evaluate the fitness of the possible solutions. The solution with a smallest fitness value is the global best position P_g.
6. Compute the r_d value using (16).
   If $r_d < r_{dmin}$ then
   $r_d = r_{dmin}$
7. For each particle *i*
   randomly initialize v_i,
   according to point 5), randomly distribute x_i within r_d around P_g.
   set $P_i = x_i$ and evaluate $f_{itness}(P_i)$
   If $f_{itness}(P_i) < f_{itness}(P_g)$ then
   $P_g = P_i$
8. Repeat
   for each particle *i*
   update x_i using (11), (12) and (13)
   evaluate $f_{itness}(x_i)$
   If $f_{itness}(x_i) < f_{itness}(P_i)$ then
   $P_i = x_i$
   If $f_{itness}(P_i) < f_{itness}(P_g)$ then
   $P_g = P_i$
   Until termination criterion is reached
9. The optimal solution:
   $U_t = \{u_t(t), u_t(t+1) \dots u_t(t+N_u-1)\}$ is given by the position of the particle having the global best position P_g.
10. According to points 1) and 2) update the possible initial solutions for the next sampling time and go to step 3.

VI. SIMULATION RESULTS

To highlight the efficiency of the proposed algorithm, the control of a highly nonlinear process, a Continuous Stirred Tank Reactor, is considered. The CSTR is a highly nonlinear chemical process where a given product *A* will be converted into another product *B* via an exothermic chemical reaction. A coolant flow controls the reactor temperature, which control in turn the concentration.

This process is governed by the following differential equations:

$$\dot{C_a}(t) = \frac{q}{v}(C_{a0} - C_a(t)) - k_0 C_a(t) e^{-\frac{E}{RT(t)}} \qquad (17)$$

$$\dot{T}(t) = \frac{q}{v}(T_0 - T(t)) + k_1 C_a(t) e^{-\frac{E}{RT(t)}} + k_2 q_c(t)\left(1 - e^{-\frac{k_3}{q_c(t)}}\right)(T_{c0} - T(t)) \qquad (18)$$

where:

C_{a0}, T_0 represent respectively the inlet feed concentration and temperature, T_{c0}, q_c are the coolant temperature and flow rate

respectively. T is the mixture temperature, and C_a is the concentration of product A. The numerical values of all chemical and thermodynamic constants can be found in [5].

To construct the TS fuzzy model of the CSTR, the data set containing 7500 samples [19] was used. The model structure is given by:

$$\hat{C}_a(t+1) = f\left(\hat{C}_a(t), \hat{C}_a(t-1), \hat{C}_a(t-2), q_c(t-1)\right) \quad (19)$$

where f represents the FIS.

The TS fuzzy model parameters were identified using 5000 input/output samples, 16 rules, a triangular membership function and the LS algorithm. The validation test results are shown in figures 2 and 3.

The constructed TS fuzzy model and the following values of the design parameters are used to determine the control law of the CSTR process.

$$N_1 = 1, N_2 = 10, N_u = 2, \text{and } \lambda = 10^{-4}$$

The control performance when a population of 10 particles and a maximal number of 10 iterations were chosen is illustrated in figures 4, 5 and 6. A good tracking of the reference trajectory with a smooth control signal is obtained.

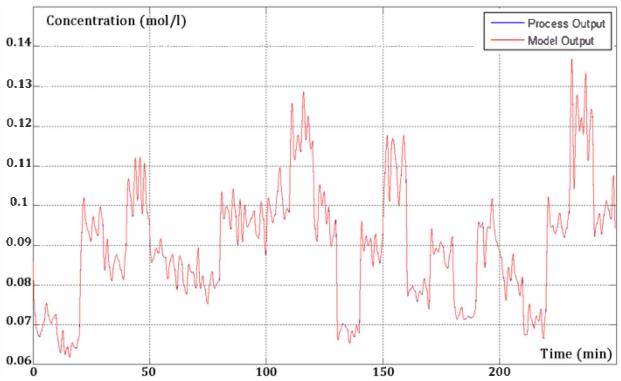

Fig. 2. Validation results (model and process output)

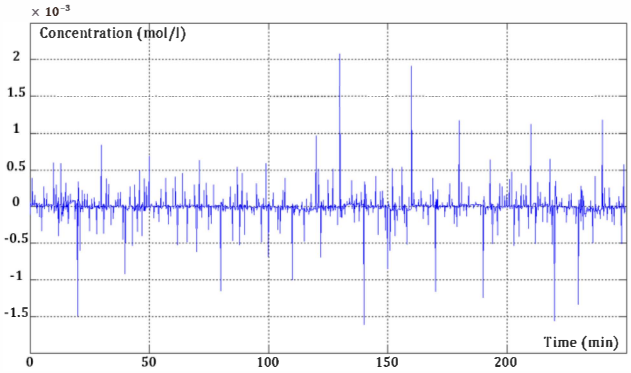

Fig. 3. Validation error.

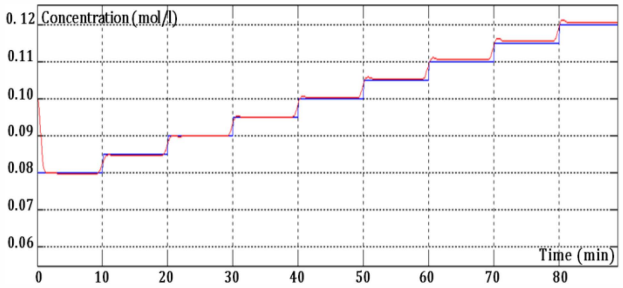

Fig. 4. Process output and reference trajectory (10 particles and 10 iterations)

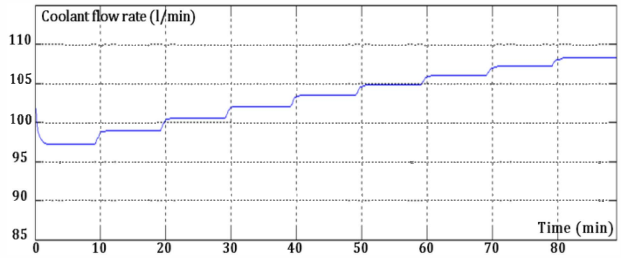

Fig. 5. Control signal (10 particles and 10 iterations)

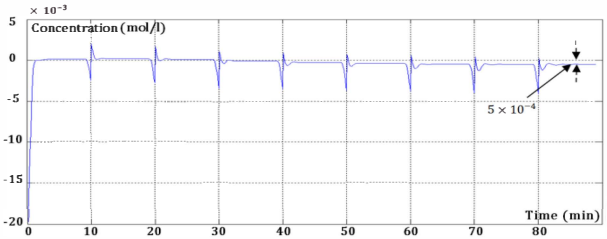

Fig. 6. Tracking error (10 particles and 10 iterations)

Figures 7, 8 and 9 show the control results when a population of 3 particles and a maximal number of 3 iterations are used. It is important to observe that a good tracking accuracy can be obtained although a small number of particles and iterations are used. This makes possible to reduce the algorithm computational time without degrading the control performance.

The execution time is a very important parameter in evaluating the applicability of control algorithms. The proposed algorithm was implemented using a PC-3.07 GHz (with i3 processor), and MatLab. Table I gives the mean squared value of the tracking error (MSE) and the possible maximal value of the sampling frequency $f_{Max} = \frac{1}{T_{exe}}$ for different values of the population size and the iterations number. T_{exe} is the execution time of the control algorithm and the MSE value is given by:

$$MSE = \frac{1}{N}\sum_{k=1}^{N}(y(k) - w(k))^2 \quad (20)$$

where N is the number of samples.

Table I shows that a small size population with few iterations could provide accurate solutions.

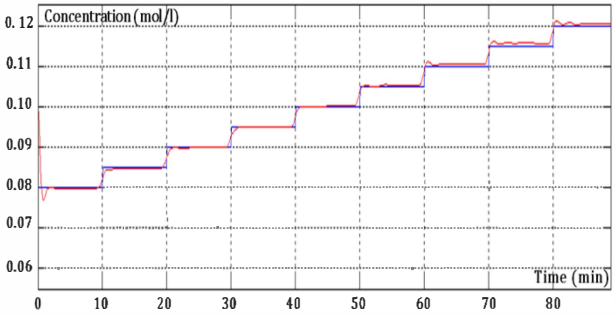

Fig. 7. Process output and reference trajectory (3 particles and 3 iterations)

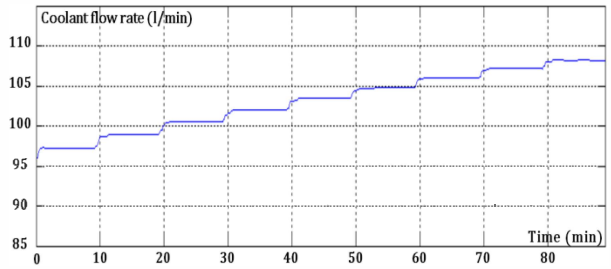

Fig. 8. Control signal (3 particles and 3 iterations)

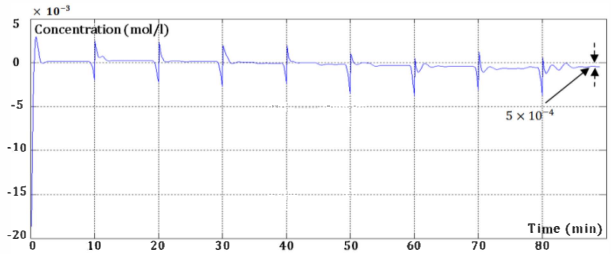

Fig. 9. Tracking error (3 particles and 3 iterations)

TABLE I
MSE AND f_{Max} FOR DIFFERENT VALUES OF POPULATION SIZE AND ITERATIONS NUMBER

Population x Iteration	MSE ($N = 890$)	f_{Max} (Hz)
10 x 10	$1.84\ 10^{-6}$	8
10 x 5	$1.47\ 10^{-6}$	15
10 x 3	$1.53\ 10^{-6}$	23
5 x 10	$1.41\ 10^{-6}$	16
5 x 5	$1.38\ 10^{-6}$	29
3 x 10	$1.44\ 10^{-6}$	27
3 x 3	$1.33\ 10^{-6}$	64

VII. CONCLUSION

In this work a fuzzy predictive control algorithm based on particle swarm optimization for nonlinear systems was developed. This algorithm uses a TS fuzzy model to predict the future values of the system output. These models have a simple structure and are known to be a good approximation for nonlinear system. It was shown that accurate solutions to the obtained nonlinear optimization problem can be achieved using a population of small size and few iterations. Then the control law in this algorithm can be implemented in real time for nonlinear systems without using numerical optimization methods that are expensive in computational time. This is essentially due to the modifications introduced on the PSO algorithm to increase its convergence rate. The computational efficiency and the control accuracy of the proposed algorithm were investigated by considering the control of the CSTR process.

REFERENCES

[1] J.Richalet, D.O'Donovan, *Predictive Functional Control, Principles and Industrial Applications*, Springer-Verlag London Limited, 2009.

[2] E.F.Camacho, C.Bordons, *Model Predicitve Control,* Springer-Verlag London Limited, 1999.

[3] B.Kouvaritakis, M.Cannon, *Nonlinear Predicitve Control, theory and practice,*The Institution of Engineering and Technology, London, United Kingdom, 2001.

[4] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control", *IEEE transactions on systems, man and cybernetics*, Vol. SMC-15, No. 1, January/February 1985. pp. 116-132.

[5] J.Espinosa, J.Vandewalle and V.Wertz, *Fuzzy logic, identification and predicitve control* , Springer-Verlag: London, 2005.

[6] J. Solis, D. Saez and P. A. Estevez, "Particle swarm optimization-based fuzzy predictive control strategy", 2006 *IEEE International Conference on Fuzzy Systems.* Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, 2006, pp. 1866-1871.

[7] Y. Song, Z. Chen and Z. Yuan, "New chaotic PSO-based neural network predictive control for nonlinear process", *IEEE Trans.on neural networks*, vol. 18, No. 2, march 2007. pp. 595-601.

[8] Z. Hou g, H. Chen and H. Li , "Neural networks predictive control using AEPSO", *Proceedings of the 27th Chinese Control Conference*, July 16-18, 2008, Kunming,Yunnan, China. pp. 180-184.

[9] J. P. Coelho, P.B. de Moura Oliveira and J. Boaventura Cunha. "Greenhouse air temperature predictive control using the particle swarm optimization algorithm". *In Computers and Electronics in Agriculture, vol.* 49, 2005, pp. 330-344.

[10] X. Wang and J. Xiao. "PSO-based model predictive control for nonlinear processes". *Lecture Notes in Computer Science 3611*. Springer-Verlag: Berlin Heidelberg, 2005, pp. 196-203.

[11] M.L. Hadjili, K. Kara, "Modelling and control using Takagi-Sugeno fuzzy models", *Saudi International Electronics, Communications and Photonics Conference*, 24-26 april 2011, Ryadh, Saudi Arabia.

[12] J. Roubos, R. Babuska, P. Bruijn, H. Verbruggen, "Predictive control by local linearization of a Takagi-Sugeno fuzzy model", *Proceedings of IEEE International Conference on Fuzzy Systems*,May 4-9, 1998, Anchorage, Alaska, USA, pp. 37-42.

[13] M. Mahfouf, S. Kandiah and D. A. Linkens.. "Fuzzy model-based predictive control using an ARX structure with feedforward". *Fuzzy Sets and Systems,* vol. 125, pp. 39-59, 2002.

[14] R. Babuska, J. Sousa, H. Verbruggen "Predictive control of nonlinear systems based on fuzzy and neural models". *Proceedings of the European Control Conference, ECC'99*, August 31- September 3, Karlsruhe, Germany, 1999, pp. 667-672.

[15] H. Sarimveis, G. Bafas. "Fuzzy model predictive control of nonlinear processes using genetic algorithms". *Fuzzy Sets and Systems,* vol.139. pp.59-80, 2003.

[16] C. Blum, D. Merkle, *Swarm Intelligence, Introduction and Applications*, Springer-Verlag: Berlin Heidelberg, 2008.

[17] M. Khairy, M. B. Fayek, E. E. Hemayed, "PSO2: Particle Swarm Optimization with PSO-Based Local Search", *IEEE 2011*. pp. 1826-1832.

[18] C. Sun, J. Zeng, S. Chu, J. F. Roddick, "Solving Constrained Optimization Problems by an Improved Particle Swarm Optimization", *2011 Second International Conference on Innovations in Bio-inspired Computing and Applications*, pp. 124-128.

[19] B. De Moor, "Daisy: Database for the identification of systems" Department of Electrical Engineering -ESAT- K.U. Leuven, Belgium. http://www.esat.kuleuven.ac.be/sista/daisy/, Used data set: CSTR, Section: Process Industry Systems, code: 98-002.