Conférence Internationale en Automatique & Traitement de Signal (ATS-2017)
Proceedings of Engineering and Technology – PET
Vol.22 pp.90 -95

An Adaptive Neuro-Fuzzy Controller for Robotic Manipulators

Sana BOUZADA[#1], Anis SAKLY[#2]

[#] *Research Unit of Industrial Systems Study and Renewable Energy (ESIER)*
National Engineering school of Monastir: University of Monastir
Ibn El Jazzar;Skaness;5019;Monastir;Tunisia

[1]bouzaida_sana@hotmail.fr
[2]sakly_anis@yahoo.fr

***Abstract*—This paper presents an adaptive Neuro-Fuzzy control scheme for robotic manipulators in the presence of parametric uncertainties and external disturbances. The development of control system is based on adaptive learning algorithms to adjust online Neuro-Fuzzy Controller (NFC). In this context, two adaptive learning algorithms called Adaptive Cooperative Particle Sub-Swarms Optimization (ACPSSO) and online PSO (OPSO) for tuning NFC controller. Using adaptive learning ability NFC controller can treat real systems dynamics.**
In this study, a two degree of freedom robotic manipulator subjected to parametric uncertainties and disturbance is considered in order to demonstrate the superior tracking performance of the proposed control scheme.

***Keywords*— TSK-type Neuro-Fuzzy model; Adaptive Cooperative Particle Sub-Swarms Optimization; Online PSO; Time-varying systems; Nonlinear systems.**

I. INTRODUCTION

The tracking control of robots manipulators have attracted considerable attention in last decade ([1], [2]). In fact, it is well known that the robot system possess highly nonlinearities and complex dynamics. Therefore, the control design presents a challenging problem ([3], [4]). Several control schemes including fuzzy logic and neural networks have been proposed for controlling the robot manipulators ([5]-[8]).

More recently Neuro-Fuzzy control has been applied with some success to the control of robotic systems ([9]-[11]). Various learning algorithms have been proposed for tuning Neuro-Fuzzy systems ([12], [13]). However, time varying behavior make it difficult to control dynamic systems using conventional techniques. Thus, to deal with the unknown robotic dynamics, adaptive learning algorithms have been employed, in this paper, for tuning neuro-fuzzy controllers. Many adaptive algorithms were designed to deal with time varying systems ([14]-[16]).These algorithms have adaptive performance to treat real system dynamics and environmental changes. In this way, online learning ability with adaptive Cooperative Particle Sub swarms optimization ACPSSO is applied to adjust Neuro-Fuzzy parameters.

The paper is organized as follows: in the next section, an overview of TSK-type Neuro-Fuzzy model is given. Then we present adaptive learning algorithms applicable to online systems in Section 3. Simulation results are evaluated in Section 4, and conclusions are given in the last section.

II. NEURO-FUZZY CONTROLLER

In this section, a TSK type Neuro-Fuzzy controller is presented. For the configuration of the proposed controller, a four layer neural network is used. As shown in Figure 1, layer 1 and 2 represent the antecedent part, and layer 2 and 4 represent the consequent part.

Layer1: input layer:

$$\begin{aligned} net_1^1 &= e(t),\ \ y_1^1 = f_1^1(net_1^1) = net_1^1 \\ net_2^1 &= \dot{e}(t),\ \ y_2^1 = f_2^1(net_2^1) = net_2^1 \end{aligned} \tag{1}$$

Where $e(t)$, $e(t)$ are inputs y_1^1 and y_2^1 are outputs of the input layer.

Layer 2: Membership function:

$$net_{ij}^2 = -\frac{(y_i^1 - m_{ij})^2}{s_{ij}^2},\ \ y_{ij}^2 = f_j^2(net_j^2) = \exp(net_j^2) \tag{2}$$

Where m_{ij}, and s_{ij} are the center mean and the standard deviation of the Gaussian membership, respectively.

Layer 3: rule layer: The number of nodes in this layer is equal to the number of fuzzy rules. Each node multiplies the incoming signals and outputs the result of product.

$$net_k^3 = (y_{1j}^2 \times y_{2l}^2),\ \ y_k^3 = f_k^3(net_{jk}^3) = net_k^3 . \tag{3}$$

Layer 4: output layer

The output layer computes the overall output .it represents the inference and defuzzification used in the fuzzy logic control (FLC).

$$\begin{aligned} y^4 &= f^4(net^4) = net^4 \\ a &= \sum_k y_k^3 w_k \qquad b = \sum_k y_k^3 \ \ net^4 = \frac{a}{b} \end{aligned} \tag{4}$$

where w_k is a linear function, which represents the consequent part.

ISSN 2356-5608

In this paper, adaptive learning algorithm is applied to find the optimal parameters of Neuro-Fuzzy controller.

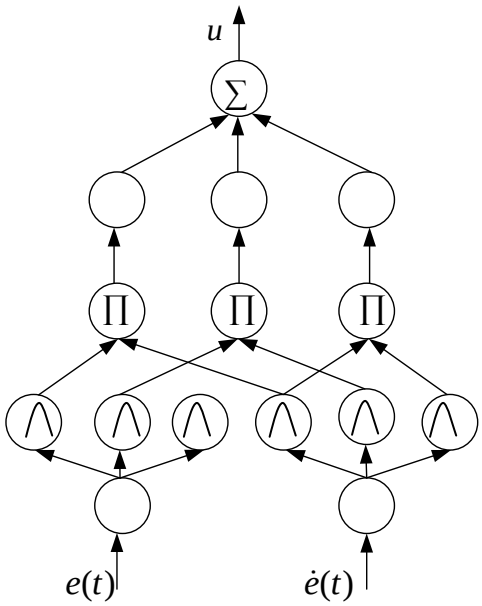

Fig. 1. Structure of TSK Neuro-Fuzzy model

III. ADAPTIVE LEARNING ALGORITHMS

A. *Online PSO* (*OPSO*)

The online PSO algorithm was designed for time-varying search space [17]. The proposed algorithm offer high adaptability to environmental changes. The process of OPSO approach is summarized in the following steps:

Step 1: Initialize the positions and velocities of all particles.

Step 2: The optimal solution of each particle at the previous instant x_{k-1}^{best} are revaluated at the current instant and the global best solution is found using the following equation:

$$x_k^g = \arg\min\left\{ f_k(x_{k-1}^{best}) \right\} \tag{5}$$

Step 3: update the velocities and positions of each particle using the following equations:

$$\begin{aligned} v_k &= w.v_{k-1} + c_1.r_1(x_k^g - x_{k-1}) + c_2.r_2(x_{k-1}^{best} - x_{k-1}) \\ x_k &= x_{k-1} + v_k \end{aligned} \tag{6}$$

Step 4: Evaluate the fitness of each particle and update x_k^{best}.

Step 5: Update the global best solution by the following equation:

$$x_k^g = \arg\min\left\{ f_k(x_k^{best}) \right\} \tag{7}$$

Step 6: The algorithm returns to Step 2 at k=k+1.

B. *Adaptive Cooperative Particle Sub-Swarms Optimization*

The Adaptive Cooperative Particle Sub-Swarms Optimization is a modified version of CPSSO algorithm [18], which adapts to online changes. In this algorithm, several sub-swarms are employed to search the space more efficiently and the information are exchanged among them during iteration process. The best solution of each sub-swarm *Xpbest*, and the global best solution *Xgbest* are assumed to be time-varying values.

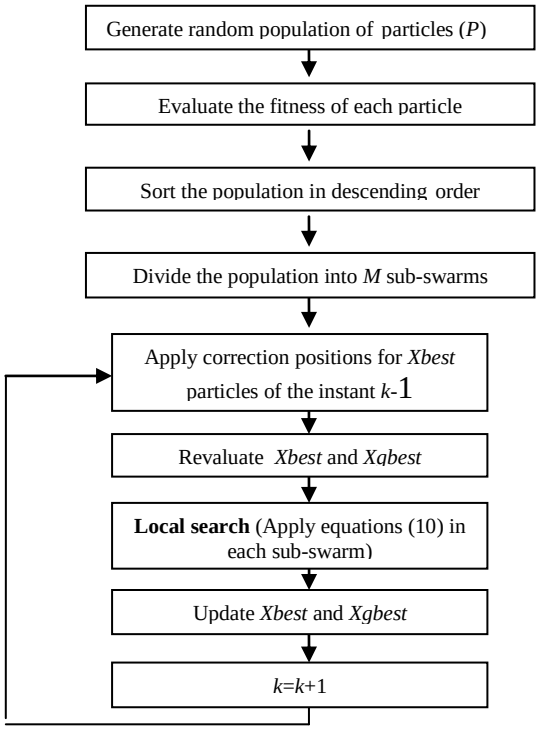

Fig. 2. Flowchart of ACPSSO Algorithm

The process of ACPSSO is summarized in the following steps:

Step 1: Initialize the positions and velocities of the sub-swarms.

Step 2: a position correction is applied for the best local particles in each sub-swarm of the previous instant $Xpbest_{k-1}$ using a probability rate (*Pr*).

$$Xpbest_i^{j'}{}_k \begin{cases} Xpbest_i^{j'}{}_k \in \left(Xpbest_{i\,k-1}^{1}, ..., Xpbest_{i\,k-1}^{M} \right) & \text{with probability } PR \\ Xpbest_i^{j'}{}_k = Xpbest_{i\,k-1}^{j} + RA & \text{with probability } (1-PR) \end{cases} \tag{8}$$

where *M* is the number of sub-swarms, *N* is the number of parameters, and *RA* is the Distance Radius [18].

The global best solution *Xgbest* of all the sub-swarms is revaluated at the current instant k as follows:

$$Xgbest_k = \arg\min\left\{ f(Xbest_k) \right\} \tag{9}$$

where $f(x)$ is assumed to be time-varying evaluation function.

Step 3: Update the velocities and positions of the particles using the following equations:

$$\begin{aligned} v_k &= w.v_{k-1} + c_1.r_1(Xgbest_k - x_{k-1}) + c_2.r_2(Xpbest_k - x_{k-1}) \\ x_k &= x_{k-1} + v_k \end{aligned} \tag{10}$$

ISSN 2356-5608

where c_1 and c_2 are positive constants namely acceleration constants, r_1 and r_2 are random numbers between 0 and 1, and w is the inertia weight.

Step 4: Evaluate the fitness of particles and update the *Xgbes*t and *Xpbest* of each sub-swarm.

Step 5: The algorithm returns to Step 1 at k=k+1.

The Flowchart of the proposed algorithm is shown in Figure 2.

IV. Application

To verify the availability and the efficiency of the proposed controller tuned with adaptive learning algorithm, computer simulations were conducted on a two-degree of freedom manipulator. The manipulator has one transitional and rotational joint in the (x,y) plane. As shown in Figure 3, the control structure is composed of two Neuro-Fuzzy controllers. Each controller is characterized by 4 rules, two inputs $e(k)$ and $\dot{e}(k)$, and the control action $u(k)$ as output. All simulation studies are carried out using Matlab 7.01. For computer simulations, the initial parameters are given in Table 1.

TABLE 1

INITIAL PARAMETERS VALUE

Parameters	***Value***
Population size (P)	50
Probability consideration rate (P_R)	0.4
RA_{max}, RA_{min}	1e-2, 1e-7
Acceleration parameters c_1, c_2	1.5, 1.5

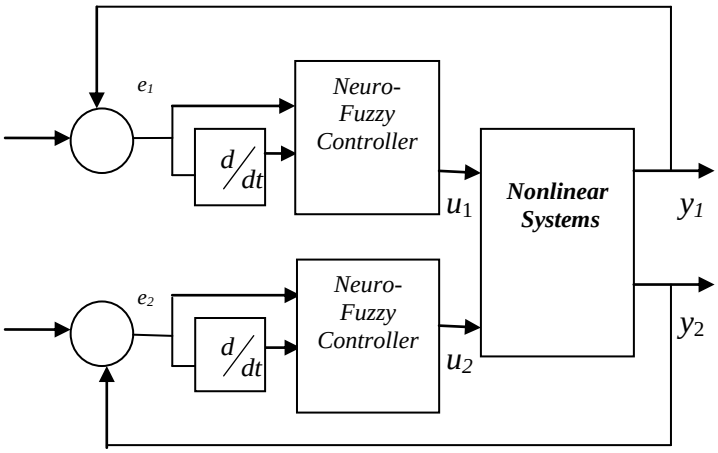

Fig. 3. Control structure of nonlinear systems.

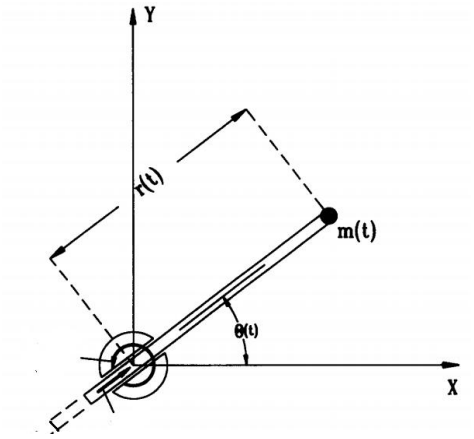

Fig. 4. A two degree of freedom manipulator.

The mathematical model of the two-degree of freedom manipulator is given by the following differential equations [19]:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f_1(x,t) + \delta f_1(x,t) + (1+\delta b_1(x,t))u_1(t) \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = f_2(x,t) + \delta f_2(x,t) + (1+\delta b_2(x,t))u_2(t) + d(t) \end{cases} \tag{11}$$

where

$$f_1(x,t) = x_1 x_4^2$$

$$f_2(x,t) = x_2 x_4$$

$$\delta f_1(x,t) = \frac{-x_4^2}{(2+2m)}$$

$$\delta f_2(x,t) = \frac{\left(-(1+m)x_1^2 - x_1(1+2m) + \left(\frac{1}{6}\right)\right)x_2 x_4}{\left(\left(\frac{5}{6}\right) - x_1 + (1+m)x_1^2\right)}$$

$$\delta b_1(x,t) = \frac{-m}{(1+m)}$$

$$\delta b_2(x,t) = \frac{(1+6x_1 - 6x_1^2(1+m))}{(5-6x_1 - 6x_1^2(1+m))}$$

$d(t)$ is an unknown but bounded external disturbance, and m is a variable payload bounded as

$$m_{min} \le m \le m_{\max}.$$

The state variables are as follows:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} r(t) \\ \dot{r}(t) \\ \theta(t) \\ \dot{\theta}(t) \end{bmatrix} \tag{12}$$

ISSN 2356-5608

and the control inputs

$$\begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = \begin{bmatrix} F(t) \\ T(t) \end{bmatrix} \tag{13}$$

Case 1:

To demonstrate the tracking performance of the proposed controller, the desired trajectories $r_d(t)$ et $\theta_d(t)$ were set as:

$$\begin{aligned} r_d(t) &= 0.75(1-\sin(\pi t/10))+0.25 \ [m] \\ \theta_d(t) &= 2\pi \sin(\pi t/10) \ [rad] \end{aligned} \tag{14}$$

For the computer simulation, time step is adopted to be 0.01 sec, and the initial states are chosen to be:

$$x(0) = \left[0.85, \ -0.075\pi, \ \pi/18, \ 0.2\pi^2\right] \tag{15}$$

Figure 5 and Figure 6 show the positions of joints 1 and 2 of the Neuro-Fuzzy controller tuned online with ACPSSO and OPSO learning algorithms, respectively. As seen in fig the proposed controller tuned with ACPSSO algorithm has high tracking performance with a smaller error than "OPSO" algorithm.

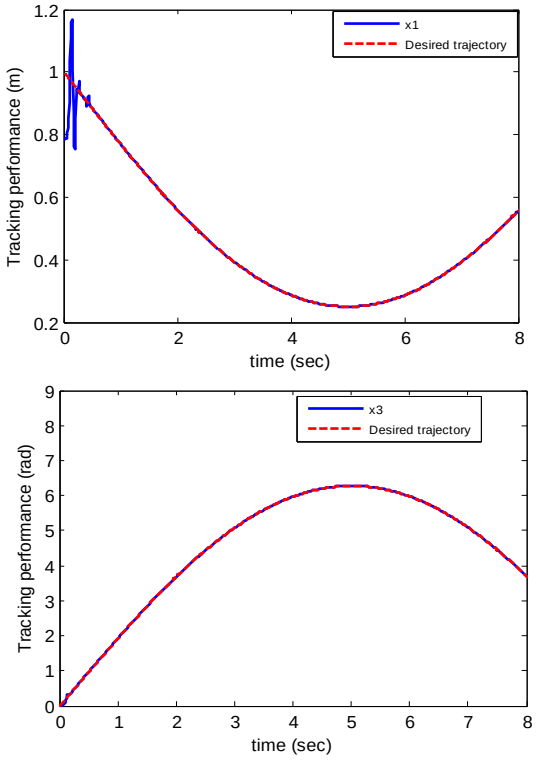

Fig. 5. Tracking responses of jont1 and joint2 with ACPSSO algorithm.

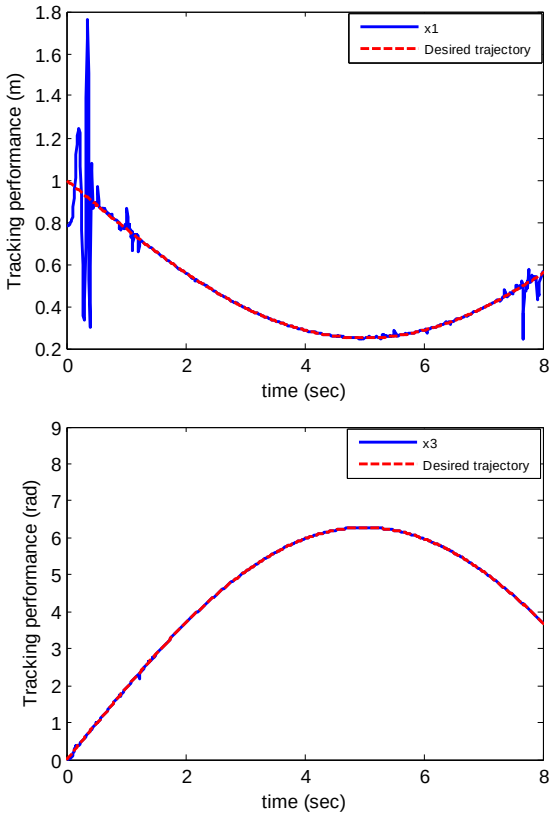

Fig. 6. Tracking responses of jont1 and joint2 with OPSO algorithm

Case 2:

To demonstrate the robustness of the proposed control scheme, an external disturbance $d(t)$=0.3sin(10t), and a variable payload $m(t)$=0.3+sin(3t) are imposed.

Figure 7 and Figure 8 present control responses of NFC tuned with ACPSSO and OPSO learning algorithms, respectively, in the presence of system uncertainties. Simulation results indicate that the proposed control scheme with adaptive learning algorithm can overcome various external disturbances to achieve excellent tracking performance.

ISSN 2356-5608

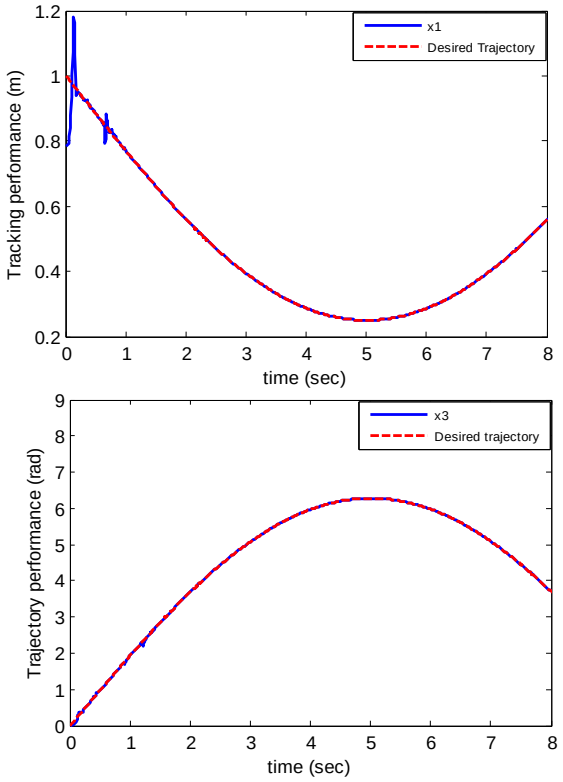

Fig. 7. Tracking responses of jont1 and joint2 with ACPSSO algorithm (Case 2).

To have quantitative comparison between adaptive learning algorithms, the root mean square error $RMSE = \left(\frac{1}{T} \int_0^T |e|^2 dt \right)^{1/2}$ is used as performance indice. Table II shows the learning results of ACPSSO and OPSO.

As seen in table II, the results show that the proposed algorithm outperforms the adaptive Particle Swarm Optimization "OPSO".

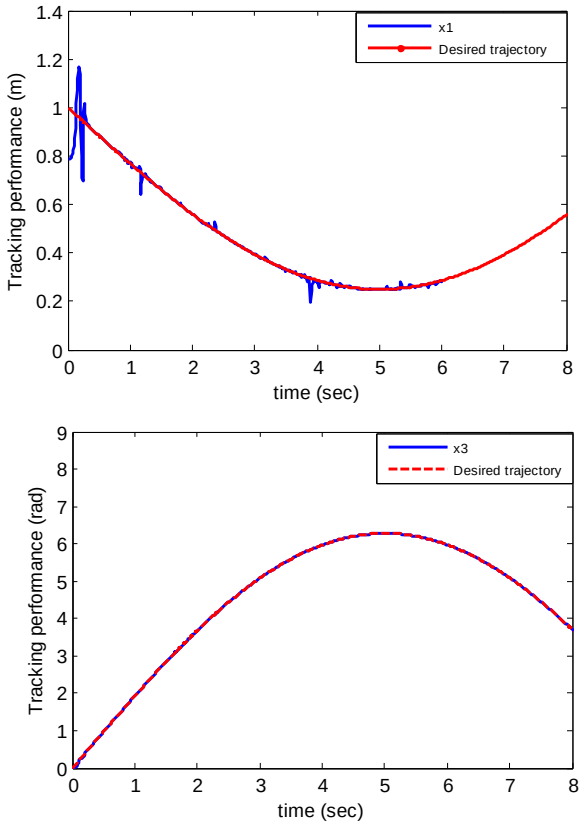

Fig. 8. Tracking responses of jont1 and joint2 with OPSO algorithm (Case 2).

TABLE II

PERFORMANCE COMPARISON OF ADAPTIVE LEARNING ALGORITHMS

	OPSO		ACPSSO	
	Joint 1	**Joint 2**	**Joint 1**	**Joint 2**
RMSE (Case 1)	0.0788	0.0124	0.0217	0.0071
RMSE (Case 2)	0.0351	0.0259	0.0195	0.0056

V. CONCLUSION:

In this study, an effective adaptive tracking control strategy is presented for robotic manipulators in the presence of disturbance and uncertainties. Based on adaptive learning algorithms the NFC controller can deal with the high nonlinearities and time-varying behavior. Simulation results show the superior tracking performance of the proposed adaptive control scheme.

REFERENCES

[1] Z. Li, S. S. Ge, M. Adams, W. S. Wijesoma, "Robust adaptive control of uncertain force/motion constrained nonholonomic mobile manipulators," *Automatica*, vol. 44, pp. 776-784, 2008.

ISSN 2356-5608

[2] P. Ruaux, G. Bourdon, S. Delaplace, " Dynamic control of wheeled mobile robot using sliding mode," in *ROMANSY* 11, 1997, p. 205-212.
[3] S. Islam, X. P. Liu, « Robust sliding mode control for robot manipulators," *IEEE Transactions on Industrial Electronics*, vol. 58, pp. 2444-2453, 2011.
[4] R. J. Wai, M. C. Lee, "Intelligent optimal control of single-link flexible robot arm," *IEEE Transactions on Industrial Electronics*, vol. 51, pp. 201-220, 2004.
[5] H. A Talebi, K. Khorasani, R. V. Patel, "Neural network based control schemes for flexible-link manipulators: simulations and experiments," *Neural Networks*, vol. 11, pp. 1357-1377, 1998.
[6] M. K. Chang, J. J. Liou, M. L. Chen, "T–S fuzzy model-based tracking control of a one-dimensional manipulator actuated by pneumatic artificial muscles," *Control Engineering Practice*, vol. 19, pp. 1442-1449, 2011.
[7] A. Mannani, , H. A. Talebi, "A fuzzy Lyapunov-based control strategy for a macro–micro manipulator: Experimental results," *IEEE transactions on control systems technology*, vol. 15, pp. 375-383, 2007.
[8] B. Subudhi, A. S. Morris, "Soft computing methods applied to the control of a flexible robot manipulator," *Applied Soft Computing*, vol. 9, pp. 149-158, 2009.
[9] L. Tian, C. Collins,"Adaptive neuro-fuzzy control of a flexible manipulator," *Mechatronics*, vol. 15, pp. 1305-1320, 2005.
[10] F. Sun, Z. Sun, L. Li, H. X. Li, « Neuro-fuzzy adaptive control based on dynamic inversion for robotic manipulators," *Fuzzy Sets and Systems*, vol. 134, pp. 117-133, 2003.
[11] S. Alavandar, M. J. Nigam, "Adaptive Neuro-Fuzzy Inference System based control of six DOF robot manipulator," *Journal of Engineering Science and Technology Review*, vol. 1, pp. 106-111, 2008.
[12] G. Leng, , T. M McGinnity, G. Prasad, "Design for self-organizing fuzzy neural networks based on genetic algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 14, pp. 755-766, 2006.
[13] A. Chatterjee, K. Watanabe, "An optimized Takagi-Sugeno type neuro-fuzzy system for modeling robot manipulators," *Neural Computing & Applications*, vol. 15, pp. 55-61, 2006.
[14] Blackwell, J. Branke, "Multi-swarm optimization in dynamic environments," In EvoWorkshops, 2004, pp. 489-500.
[15] A. Carlisle, and G. Dozier, "Tracking changing extrema with particle swarm optimizer," Auburn Univ., Auburn, AL, Tech. Rep. CSSE01-08, 2001.
[16] T.M. Blackwell, "Swarms in dynamic environments," In Genetic and Evolutionary Computation—Gecco, 2003, pp. 1-12.
[17] T. Nishida, T. Sakamoto, "Adaptive PSO for online identification of time-varying systems," *IEEJ Transactions on Electronics, Information and Systems*, vol. 131, pp.1642-1649, 2011.
[18] S. Bouzaida, and A. Sakly, "Online Control of Nonlinear Systems using Neuro-Fuzzy Design tuned with Cooperative Particle Sub-Swarms Optimization," In *Proceedings Engineering & Technology*, 2013, pp. 6-10.
[19] E. Freund, "Fast nonlinear control with arbitrary pole-placement for industrial robots and manipulators," *The International Journal of Robotics Research*, vol. 1, pp. 65-78, 1982.

ISSN 2356-5608