Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 9 (4): 639-650, 2009
ISSN 1812-5654

Hybrid Control of Flexible Manipulator

[1]F. Farivar, [2]M. Aliyari Shoorehdeli, [2]M. Teshnehlab and [2]M.A. Nekoui
[1]Department of Mechatronics Engineering, Science and Research Branch,
Islamic Azad University, Tehran, Iran
[2]Department of Electrical Engineering, K.N.T University of Technology, Tehran, Iran

Abstract: This study describes hybrid control methods to control a flexible manipulator with payload. The dynamic equation of the system has been derived by Lagrange's method. The designed controllers consist of two parts, classical controllers, PID and Linear Quadratic Regulation (LQR) and hybrid controllers, Fuzzy Neural Network (FNN) controller with Feedback Error Learning (FEL) and Sliding mode control using Gaussian Radial Basis Function Neural Network (RBFNN). The fuzzy neural network and radial basis function neural network are trained during control process and they are not necessarily trained off-line.

Key words: Flexible manipulator, hybrid control, fuzzy neural network, feedback error learning, sliding mode, sliding surface, radial basis function neural network

INTRODUCTION

From a mechatronic point of view, the performance of electro-mechanical motion systems can be improved by changing both the mechanical design and the controller. Flexible link robot manipulators play an important role in modern industry. The potential advantages that arise from the use of light-weight flexible-link manipulators are faster operation, lower energy consumption and less costly structures.

The control of flexible link systems is very important in some fields, for instance the aerospace industry, mainly due to the use of light-weight materials in large space structures and flexible space robots. Thus, modeling and controlling of robots with elastic links, has its own problems. In addition dynamical structure of such robots includes nonlinear and uncertainty factors in the model. So, designing an adaptive controller that is able to provide the performance characteristics such as tracing the reference input, eliminating the disturbance and the conditions of response speed is needed. Dynamic equation of the system has been obtained by Lagrange's method (Tian and Collins, 2005; Korayem and Basu, 1994).

In this study two hybrid control methods are used to control a flexible manipulator with payload. The designed controllers consist of two parts, classical controllers; PID and Linear Quadratic Regulation (LQR) and hybrid controllers; Sliding mode control using Gaussian Radial Basis Function Neural Network (RBFNN) and Feedback Error Learning (FEL) technique.

The FEL method was used to develop a neural network controller to learn the inverse dynamics of the flexible-link system (Nakanishi and Schaal, 2004; Talebi *et al*., 1998). The NN learning is done with gradient descend method and error propagation algorithm (Miyamura and Kimura, 2002). After that the reverse of the process has been made completely and due to the zero error, the classic controller output becomes zero automatically.

The FEL method exploited in this paper is consisted of a PID controller and Fuzzy Neural Network (FNN) controller which is adapted based on output of PID controller. In closed loop system, the FNN controller is positioned in forward path which is to learn the inverse dynamics of the flexible manipulator system. The FNN is a four-layer network with fast parameter learning. So, the adaptation of controller and system conditions is online and fast. Also, the simulation results show these points.

Cheng and Wen (1993) developed a neural network controller for a flexible-link manipulator. The hub position and velocity measurements were used to stabilize the system and a neural observer/controller was proposed to drive the flexible arm to track a desired trajectory (Torfs *et al*., 1998). Donne and Ozguner (1994) proposed a neural controller assuming partial knowledge of the dynamics of the flexible-link. The unknown part of the dynamics was identified by a supervised learning algorithm. The control was constructed of two stages, an optimal controller and an unsupervised neural network controller using model-based predictive control.

Corresponding Author: F. Farivar, Department of Mechatronics Engineering, Science and Research Branch,
Islamic Azad University, P.O. Box 14155/775, Tehran, Iran
Tel: +982144869655

The scheme was based on an identification stage that also requires feedback from the states of the system. Newton and Xu (1993a, b) considered the joint tracking control problem for a space manipulator using feedback error-learning technique. However, tip position tracking cannot be guaranteed specially for high-speed desired trajectories.

MATERIALS AND METHODS

Dynamical modeling of single flexible manipulator: The manipulator is shown in Fig. 1 and is modeled as a pinned-free flexible with a payload at one end.

To obtain the equations of motion from Lagrange's formulation, we consider these assumptions:

- Using the Euler-Bernoulli theory of beam, to analyze the robot's manipulator
- Using the linear elastic theory
- The relative deformation that exists in the motor of robot is neglected
- The link is considered to be light. Therefore, the properties of each link are assumed to be a function of its length
- We neglect the effect of gravitational potential energy
- The angular displacement in the elastic links of the robots in the direction of X, Y axis is zero

The moment of inertia of the arm about the hub O is denoted by J_f and ρ is the linear mass density. The link has length L and the payload mass is given by M_e.
The control torque τ is applied at the hub of the manipulator through the rotary actuator. The angular displacement of the manipulator, moving in the XOY plane, is denoted by θ. The parametric values are shown in Table 1.

For an angular displacement θ and an elastic deflection y(x, t) the total displacement u(x, t) of a point, measured at a distance x from the hub can be described as a function of both of the above, measured from the direction of OX.

$$u(x,t) = x\theta(t) + y(x,t) \tag{1}$$

According to Eq. 1, the total kinetic energy of the link is:

$$T = \frac{1}{2}J_f\dot{\theta}^2 + \frac{1}{2}\int_0^t \dot{u}^2(x,t)\rho A dx + \frac{1}{2}M_e\dot{u}^2(L,t) \tag{2}$$

Axis of OX is considered as the base.

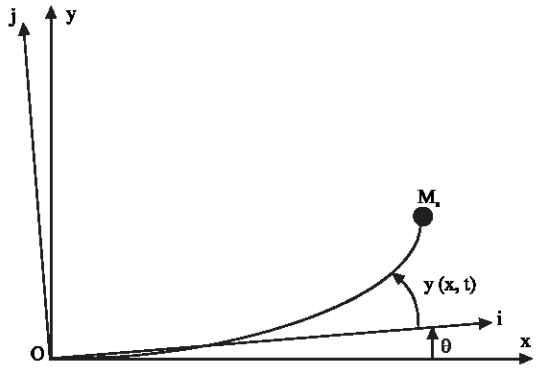

Fig. 1: Schematic representation of the flexible manipulator system

Table 1: Parametric value for the flexible manipulator

Symbols	Physical parameters	Values
L	Length	0.61 m
A	Section area	3×10^{-5} m^2
ρ	Density	7.8×10^{3} kg m^{-3}
E	Young modulus	200×10^{9} N m^{-2}
I	Second moment area	2.5×10^{-12} m^4
M_e	Payload mass	31.7×10^{-3} kg
J_f	Moment of inertia of hub	4.3×10^{-3} kg m^2

The potential energy, caused from elasticity of the system can be written as:

$$V = \frac{1}{2}EI\int_0^t \left(\frac{d^2u(x,t)}{dx^2}\right)^2 dx \tag{3}$$

E is the modulus of elasticity for the beam material and we denote the second moment of area of the beam cross-section. The lagrangian of the system can be written as:

$$L = T - V \tag{4}$$

It is possible to obtain the dynamic equation of the system, using the Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial l}{\partial \dot{q}_i}\right) - \left(\frac{\partial L}{\partial q_i}\right) = Q_i, \quad i = 1,2,...,n \tag{5}$$

where, Q_i is the generalized force related to the generalized coordinate q_i.

The generalized force can be specified, according to the virtual work, done by no conservative force that acts on the system.

According to Eq. 2 and 5, the equation of manipulator motion can be obtained as follows:

$$E_I\frac{\partial^4 y(x,t)}{\partial x^4} + \rho A\left(x\ddot{\theta} + \frac{\partial^2 y(x,t)}{\partial t^2}\right) = 0 \tag{6}$$

The dynamic equation of the manipulator is described as:

$$J_s\ddot{\theta} + M_eL\left.\frac{\partial^2 y(x,t)}{\partial t^2}\right|_{x=L} + \rho A\int_0^A \frac{\partial^2 y(x,t)}{\partial t^2}dx = \tau \quad (7)$$

Where:

$$J_s = J_f + M_eL^2 + \frac{1}{3}\rho AL^3$$

The corresponding boundary and initial conditions are given by:

$$\begin{aligned} &y(0,t) = 0\\ &y(x,0) = 0\\ &\left.\frac{\partial y(x,t)}{\partial x}\right|_{x=0} = 0\\ &\left.\frac{\partial y(x,t)}{\partial x}\right|_{t=0} = 0\\ &\left.\frac{\partial y^2(x,t)}{\partial x^2}\right|_{x=L} = 0\\ &E_I\left.\frac{\partial y^3(x,t)}{\partial x^3}\right|_{x=L} = M_e\left.\frac{\partial y^2(x,t)}{\partial t^2}\right|_{x=L} \end{aligned} \quad (8)$$

Note that the displacement y(x, t) can be considered according to the hermitic polynomial terms.

$\varphi_i(x)$ are the function of hermitic figures and indicate the displacement and flexural slope in the beam.

$q_i(t)$ indicates the degrees of freedom of manipulator.

$$y(x,t) = \sum_{i=0}^{n}\varphi_i(x)q_i(t) \quad ,i = 1,2,...,n \quad (9)$$

Substituting Eq. 9 into Eq. 6 and 7 by applying boundary and initial conditions Eq. 8, the following ordinary differential equations can be derived.

$$J_s\ddot{\theta} + \sum_{i=1}^{n} a_i\frac{\partial^2 q_i(t)}{\partial t^2} = \tau \quad (10)$$

$$\frac{\partial^2 q_i(t)}{\partial t^2} + \omega_i^2 q_i(t) + b_i\ddot{\theta} = 0 \quad (11)$$

If X is assumed as state space variable $X^T = \left[\theta, \dot{\theta}, q_1, \dot{q}_1, ..., q_n, \dot{q}_n\right]$ and y is assumed as the output, $Y^T = \left[\theta, q_1, ..., q_n\right]$. The form of the state equation is:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & 0 & \frac{a\omega^2}{J_s - ab} & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & -\frac{J_S\omega^2}{J_s - ab} & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ 1\\ 0\\ -\frac{b}{J_s - ab}\end{bmatrix}\tau$$

Table 2: Mode dependent parameters

Mode	β_i	ω_i	a_i	b_i
1	6.9626	5.0215×10^{3}	9.6539×10^{-3}	2.2248×10^{-1}
2	11.9680	4.3837×10^{4}	3.2674×10^{-3}	6.9997×10^{-2}

$$y = \begin{bmatrix}1 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} \quad (12)$$

Designing the PID controller: Here, we are going to design a PID controller in order to improve the step response of the system. Different methods are suggested for PID controller designing. All these methods, notice the process of choosing the control parameters in order to provide the desired operation characteristics. The method that we have used here is Ziegler-Nichols approximate method.

Considering the numerical quantities of Table 2 and the determined state equations for two modeling modes in the previous section, the poles of the opened- loop system will be:

$$P_1 = \{0, \quad 0, \quad 77.8i, \quad -77.8i\} \text{ and } P_2 = \{0, \quad 0, \quad 2.09i, \quad -2.09i\}$$

Thus, the existence of mixed coupled poles in the system transformation function in second mode, make the Ziegler-Nichols design method, impossible. In order to solve the problem and make the system stable, we have assigned the system eigen value.

This problem leads to designing a state feedback for the system and then a PID controller is designed by state feedback for the stabilized system. The transformation function of the PID controller for both modes is obtained as follows:

$$Gl_{PID} = \frac{0.0002404S^2 + 0.0004808S + 0.0002404}{2S}$$

$$G2_{PID} = \frac{7.379e-5S^2 + 0.0001054S + 3.765e-5}{2.8S}$$

Linear quadratic regulation and non-zero regulation: This method determines the state feedback gain matrix that minimizes J in order to achieve some compromise between the use of control effort, the magnitude and the speed of response that together guarantee a stable system.

After linearization of the system:

$$\dot{x}(t) = Ax(t) + Bu(t) \quad (13)$$

Determine the matrix k of the LQR vector:

$$u(t) = -kx(t) \quad (14)$$

So in order to minimize the performance index:

$$J = \int_0^\infty \left(x^T Q x + u^T R u\right) dt \quad (15)$$

where, Q and R are the positive definite hermetical or real symmetric matrix.

Note that the second term on the right hand side, accounts for the expenditure of the energy on the control efforts. The matrix of Q and R determine the relative importance of the error and the expenditure of this energy. According to the full rank of controllability matrix (A, B) and matrix $\begin{bmatrix} A & B \\ C & D \end{bmatrix}$, the LQR is designed with integral control and then u (t) is applied to the nonlinear system in order to track of the unit step function.

Structure of fuzzy neuro network: Here, we will construct a feed forward four layers fuzzy neural network. To implement the fuzzy control rules stated in Eq. 16, first layer accepts input variables. Its nodes represent input linguistic variables. Second layer is used to calculate Gaussian Membership values. Nodes in this layer represent the terms of the receptive linguistic variables. Nodes at the third layer represent fuzzy rules, the layer where each node is related to an individual output of the system. The links between third layer and fourth layer are connected by the weighing values w_p^i.

For a multi-input single output FNN system, let x be the input linguistic variable and α_j as the firing strength of rule j, which is obtained by the product of the grades of the membership function $\mu_{A_{ij}}(x)$ in the antecedent. The proposed FNN is shown in Fig. 2. If w_j represents the j-th consequence link weight, the inferred value y, is then obtained by taking the weighted sum of its input (Lee and Teng, 2000).

The proposed FNN realizes the following fuzzy control rules:

$$R^j: \text{If } u_{1j} \text{ is } A_{1j},\ldots,u_{nj} \text{ is } A_{nj} \text{ then } y = w_j \quad (16)$$

where, for i = 1, 2,...,n, $u_{ij} = x_i$, $A_{1j},\ldots,A_{nj}$ are fuzzy sets, w_j is a fuzzy singleton and n is the number of inputs. Finally, the output of FNN is obtained:

$$y^* = \sum_{j=1}^{m} \alpha_j w_j \quad (17)$$

Where:

$$\alpha_j = \prod_{i=1}^{n} \mu_{A_{ij}}(u_{ij}) \quad (18)$$

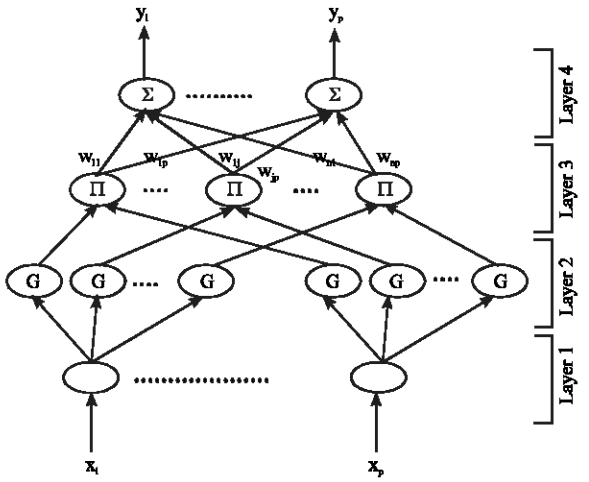

Fig. 2: The configuration of the proposed FNN

Layered operation of the FNN: Here, we shall indicate the signal propagation and operation functions of the nodes in each layer. In the following description U_i^k denotes the i-th input of a node in the k-th layer, O_j^k denotes the i-th node output in layer k.

Layer 1: Input layer: The nodes in this layer only transmit input value to the next layer directly.

$$O_i^1 = u_i^1 \quad (19)$$

From Eq. 19, the link weight at first layer w_i^1 is unity.

Layer 2: Membership layer: In this layer, each node performs a membership function and acts as a unit of memory. The Gaussian function is adopted here as a membership function, thus we have:

$$O_{ij}^2 = \exp\left(-\frac{(u_{ij}^2 - m_{ij})}{\sigma_{ij}^2}\right) \quad (20)$$

where, m_{ij} and σ_{ij} are the center (or mean) and width (or Standard Deviation-STD) of the Gaussian membership function. The subscript ij indicates the j-th term of the i-th input x_i.

Layer 3: Rules layer: The nodes in this layer are called rate nodes. The following AND operation is applied to each rule node to integrate this fan-in values:

$$O_i^3 = \prod_i u_i^3 = \exp\left(-(D_i(u_i^2 - m_i))^T (D_i(u_i^2 - m_i))\right) \quad (21)$$

Where:

$$D_i = \text{diag}\left[\frac{1}{\sigma_{1i}}, \frac{1}{\sigma_{2i}}, \ldots, \frac{1}{\sigma_{ni}}\right],\ u_i = [u_{1i}, u_{2i}, \ldots, u_{ni}]^T \text{ and } m_i = [m_{1i}, m_{2i}, \ldots, m_{ni}]^T$$

The output O_i^3 of a rule node represents the firing strength of its corresponding rule.

Layer 4: Output layer: Each node in this layer is called an output linguistic node. This layer performances the defuzzification operation. The node output is a linear combination of consequences obtained from each rule, That is:

$$y_i = O_j^4 = \sum_{i=1}^{m} u_{ij}^4 w_i^4 \quad (22)$$

where, $u_i^4 = O_i^3$ and w_{ij}^4 (the link weight) is the output action strength of the j-th output associated with the i-th rule. The w_{ij}^4 are the tuning factors of this layer.

Finally, the overall representation of input x and the m-th output y is:

$$y_m(k) = O_m^4(k) = \sum_{j=1}^{m} w_{mj} \prod_{i=1}^{n} \exp\left(-\frac{(x(k)-m_{ij})^2}{\sigma_{ij}^2}\right) \quad (23)$$

where, m_{ij}, σ_{ij} and w_{mj} are tuning parameters.

In Fig. 3, the signals that are fed into the FNN can be calculated as (Zhou *et al*., 2002):

$$\begin{aligned} e_k &= u_r - y_{m,k} \\ eck &= y_{m,k} - y_{m,k-1} \end{aligned} \quad (24)$$

During the on line learning process, The fuzzy control rule stated in Eq. 13 and the input command of the plant V_{com} have been tuned. From Fig. 3, we have the following equation.

$$V_{com} = V_{KT} + \Delta V_K \quad (25)$$

The method used in Fig. 4 is called Feedback Error learning.

FEL structure: From controlling point of view, this method is an adaptive method. Something that propounded earlier as FEL is Fig. 4 (Topalov *et al*., 1998):

As it is shown in Fig. 4 reverse process is built on the way of feed-forward. Something that is very important in this case is the error that is used after the PID classic controller as a signal for training the reverse process.

After making the reverse process, completely, because the error is zero, the classic controller output will exit the circuit by itself. Now, if every kind of disturbance or parameter changing occurs, the classic controller will come in again and will take the control and at this moment, new reverse process will be built. So, by this method we would have a dynamic control strategy. To express this method, below formulation is presented that is shown as a feedback adaptive controller in Fig. 5.

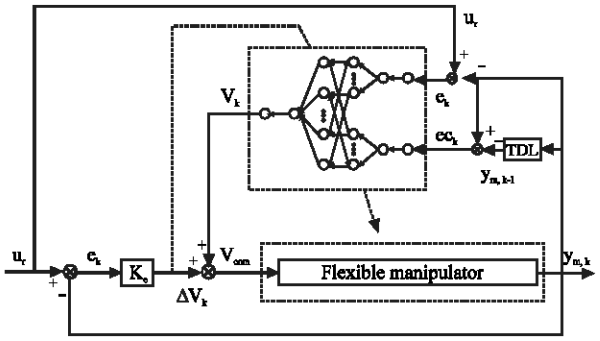

Fig. 3: Block diagram of the fuzzy neural network control system for Flexible Manipulator.

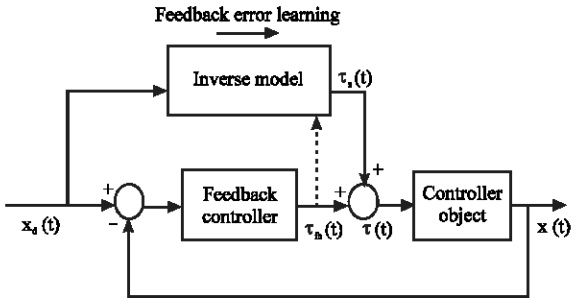

Fig. 4: Feedback error learning scheme

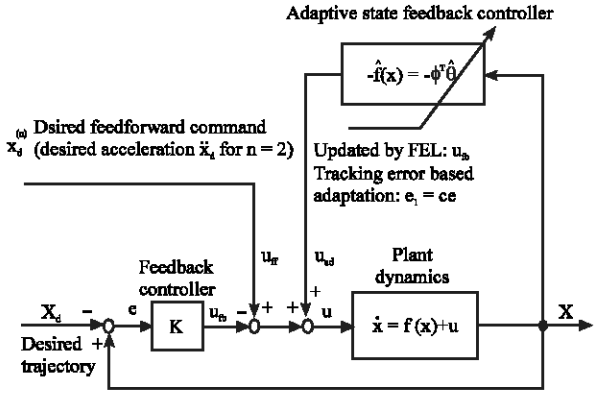

Fig. 5: Nonlinear adaptive control and FEL with an adaptive state feedback controller for a class of n-order nonlinear SISO systems

In this representation, real states are used to feed the reverse system, as an input. By considering this structure and by using the existing principles in adaptive control, equations for updating of value can be written.

Learning algorithm: Consider the single output case for simplicity. Present goal is to minimize the following cost function:

$$E(k) = \frac{1}{2}(y(k) - \hat{y}(k))^2 = \frac{1}{2}\sum_j (y(k) - O^4(k))^2 \quad (26)$$

where, y (k) is the desired output and $\hat{y}$ (k) = O^4 (k) is the current output for each discrete time k, in each training cycle starting at the input nodes in the current output $\hat{y}$ (k).

By using BP learning algorithm, the weighing vector of the FNN is adjusted such that the error defined in Eq. 26 is less than a designed threshold value after a given number of training cycles. The well-known algorithm may be written briefly as:

$$W(k+1) = W(k) + \Delta W(k) = W(k) + \eta(-\frac{\partial E(k)}{\partial W}) \quad (27)$$

where, in this case η and w represent the learning rate and tuning parameters of the FNN. Let e (k) = y (k)−$\hat{y}$ (k) and W = [m, σ, w]^T be the training error and weighting vector of the FNN, then the gradient of error E(.) in Eq. 26 with respect to an arbitrary weighting vector w is:

$$\frac{\partial E(k)}{\partial W} = -e(k)\frac{\partial \hat{y}(k)}{\partial W} = e(k)\frac{\partial O^4(k)}{\partial W} \quad (28)$$

By recursive application of the chain rule, the error term for each layer is first calculated and the parameters in the corresponding layers are adjusted. With the FNN Eq. 23 and cost function defined in Eq. 26, the update rule of w_{ij} are derived:

$$w_{ij}(k+1) = w_{ij} - \eta^w \frac{\partial E(k)}{\partial w_{ij}} \quad (29)$$

Where:

$$\frac{\partial E(k)}{\partial w_{ij}} = -e(k).O_i^3$$

Similarly, the update laws of m_{ij} and σ_{ij} are:

$$m_{ij}(k+1) = m_{ij} - \eta^m \frac{\partial E(k)}{\partial m_{ij}} \quad (30)$$

$$\sigma_{ij}(k+1) = \sigma_{ij} - \eta^\sigma \frac{\partial E(k)}{\partial \sigma_{ij}} \quad (31)$$

Where:

$$\frac{\partial E(k)}{\partial m_{ij}} = -\sum_k e(k) w_{ik} O_k^3 \frac{2(x_i - m_{ij})^2}{\sigma_{ij}^2} \quad (32)$$

$$\frac{\partial E(k)}{\partial \sigma_{ij}} = -\sum_k e(k) w_{ik} O_k^3 \frac{2(x_i - m_{ij})^2}{\sigma_{ij}^3} \quad (33)$$

The BP algorithm is a widely used algorithm for training multilayer network by means of error propagation via variation calculus. But its success depends upon the quality of the training data.

Sliding mode control: In the design of sliding mode controller for flexible manipulator, the control objective is to drive the output to the desired unit step function. So by defining the tracking error to be in the following from (Slotine and Li, 1991):

$$e = y - y_d \quad (34)$$

The sliding surface can be written as:

$$S = \ddot{e} + 3\delta\,\ddot{e} + 3\delta^2\dot{e} + \delta^3 e \quad (35)$$

where, δ is a positive definite constant and

$$e = y - y_d = y - 1 \quad (36)$$

$$\dot{e} = \dot{y} - \dot{y}_d = x_2 + x_4 \quad (37)$$

$$\ddot{e} = \ddot{y} - \ddot{y}_d = \dot{x}_2 + \dot{x}_4 \quad (38)$$

$$\dddot{e} = \dddot{y} - \dddot{y}_d \quad (39)$$

Structure of radial basis function neural network: Radial Basis Function Neural Network (RBFNN) can be used to control the flexible joint. Control structure is shown in Fig. 6.

The input of RBFNN is a sliding surface (Önder Efe *et al*., 2001). The architecture of radial basis function consists of three layers, the input, the hidden and the output layer as shown in Fig. 7.

Learning algorithm: The goal is to minimize the following cost function:

$$E(k) = S(k).\dot{S}(k) \quad (40)$$

where, S(k) is the sliding surface that was described earlier.

By using BP algorithm, the weighting vector of the RBFNN is adjusted such that the cost function defined in

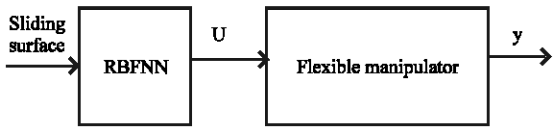

Fig. 6: The general block diagram of control

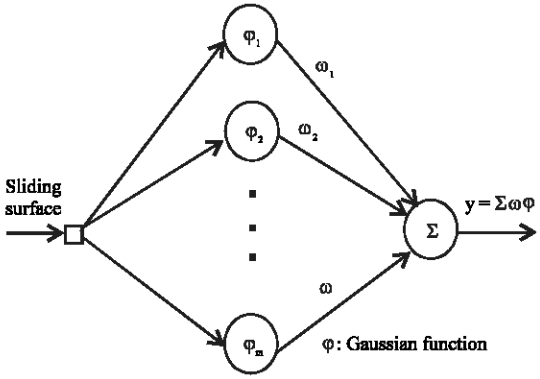

Fig. 7: The configuration of RBF neural network

Eq. 40 is less than a designed value. The well-known algorithm may be written briefly as:

$$w(k+1) = w(k) + \Delta w(k) = w(k) + \eta(-\frac{\partial E(k)}{\partial w}) \quad (41)$$

where, η and w represent the learning rate and tuning parameter of RBFNN. The gradient of E(.) in Eq. 41 with respect to a weighting w is:

$$\frac{\partial E(k)}{\partial w} = S(k).\frac{\partial \dot{S}(k)}{\partial O_{RBF}}.O^{S-1} \quad (42)$$

Where:

$$\frac{\partial \dot{S}(k)}{\partial O_{RBF}} = \left(\frac{a\omega_i^2 - J_S\omega_i^2}{J_S - ab}\right)\left(\frac{1}{J_S - ab}\right) + 3\delta^2\left(\frac{1-b}{J_S - ab}\right) \quad (43)$$

RESULTS AND DISCUSSION

Simulation of PID controller: By exerting this controller to the system, the step response and the closed- loop response in both modes will be as in Fig. 8.

It is clear that the PID controller doesn't create a suitable step response for the system. Intense transient oscillations and high overshoot are shortcomings of such controller, also the parameters of this controller are constant, no adaptation with system dynamical changes, occur. In the next section, by designing a compound controller, we try to improve the system response.

Simulation of LQR and non-zero regulation: The result of simulations after using LQR controllers for mode 1 and 2 are to be shown in Fig. 9 and 12.

The amplitude of oscillations around zero for mode 1 is about 0.001. These oscillations will damp after some times. And also the amplitude of oscillations around zero

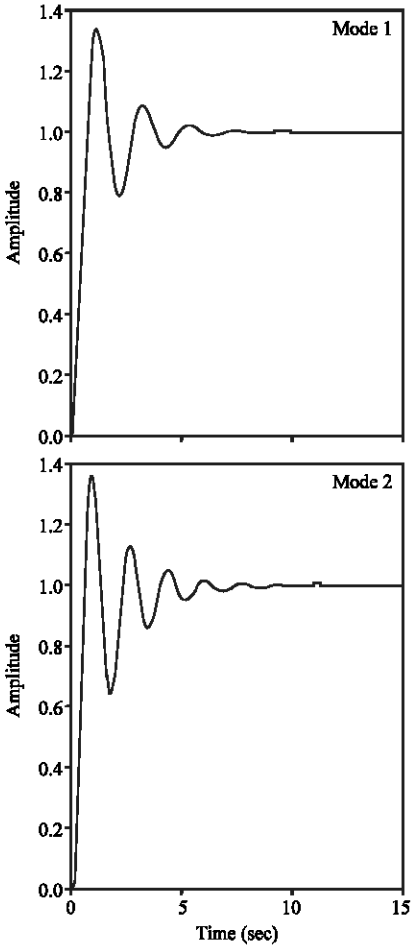

Fig. 8: Step response by exerting PID controller, mode 1 and mode 2

for mode 2 is about 0.0015. Comparing these two damped oscillations, shows that the frequency of oscillations in mode 2 is more than mode 1 but the amplitude of its damped oscillation is less than mode1.

We used non zero regulation controller for regulating the unit step response. The result of simulations is to be shown in Fig. 10 and 13. After using this controller, the oscillations have to be done around 1 and the amplitude of damped oscillations is not high. Figure 11 and 14 represent the amplitude of oscillations.

Simulation of hybrid control (FNN and FEL): Here, we present the results of the performed simulations for system hybrid control. We consider the step torque as system's input and will determine the system's output based on the state equations in modeling.

By applying a PD controller and FNN gird that was designed in the previous section, on plant, we examine the

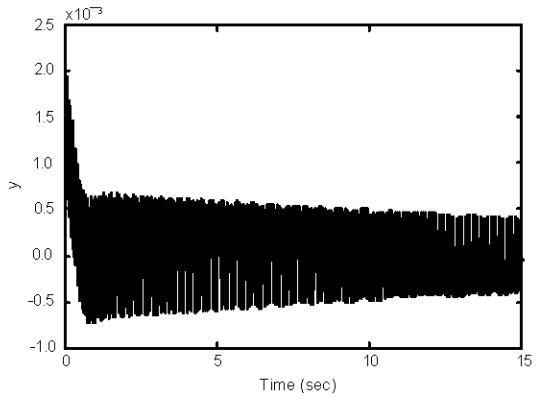

Fig. 9: Output of flexible manipulator with LQR controller (mode 1)

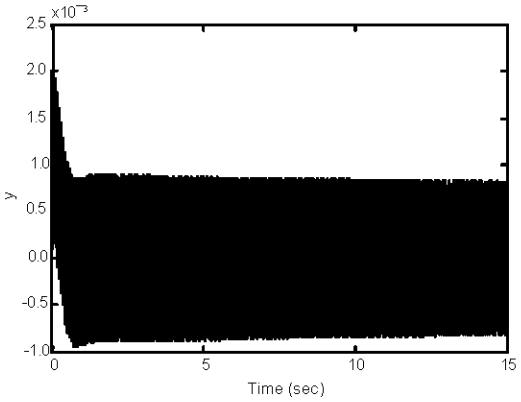

Fig. 12: Output of flexible manipulator with LQR controller (mode 2)

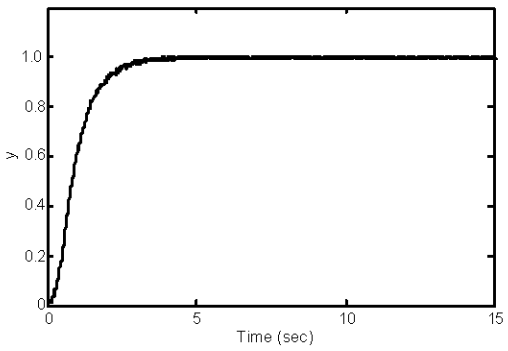

Fig. 10: Output of flexible manipulator with LQR and non-zero regulation controllers (mode 1)

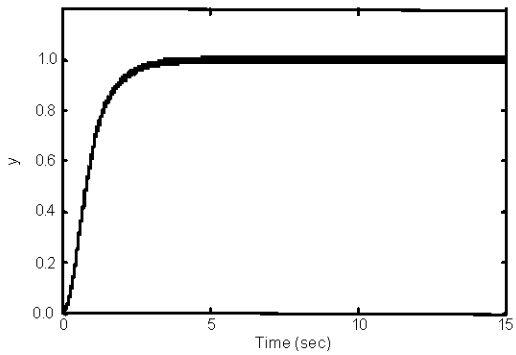

Fig. 13: Output of flexible manipulator with LQR and non-zero regulation controllers (mode 2)

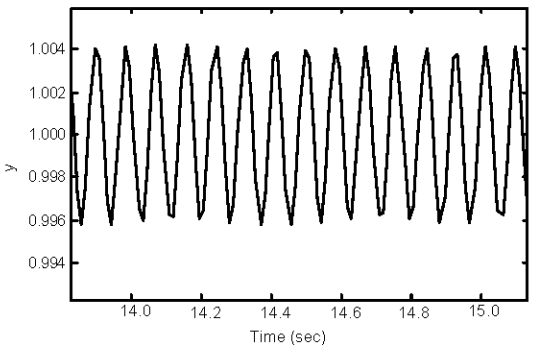

Fig. 11: Zoom of Output of flexible manipulator with LQR and non-zero regulation controllers (mode 1)

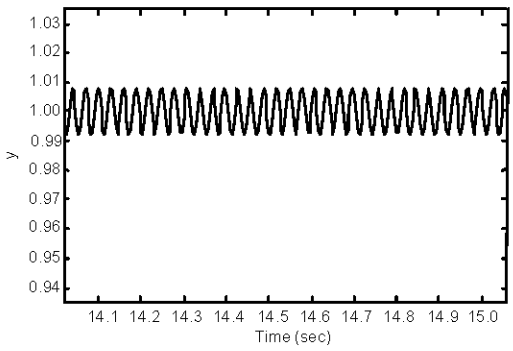

Fig. 14: Zoom of output of flexible manipulator with LQR and non-zero regulation controllers (mode 2)

system step response. The designed controller simulation has done in two modes. The reason for choosing these two modes is to show the controller conformance with different states of the plant. In second mode, frequency of vibration is twice as much of first mode as shown in Fig. 15.

It is obvious that the response overshoot has appeared at 1.3 sec and the system response involves an overshoot of 28%. But at 3.7 sec the system is in its stable state and error is less than 1%. It is clear that amplitude level of output vibration has gone down to 0.00001 mm.

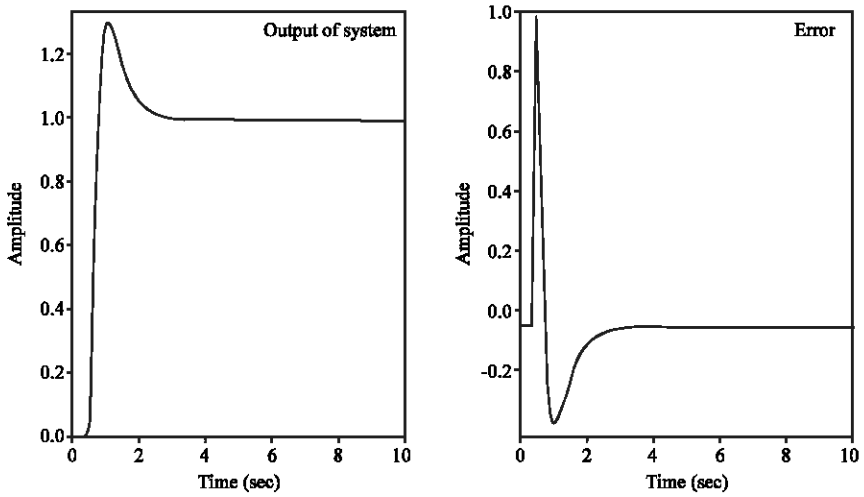

Fig. 15: Step response and error for mode 1

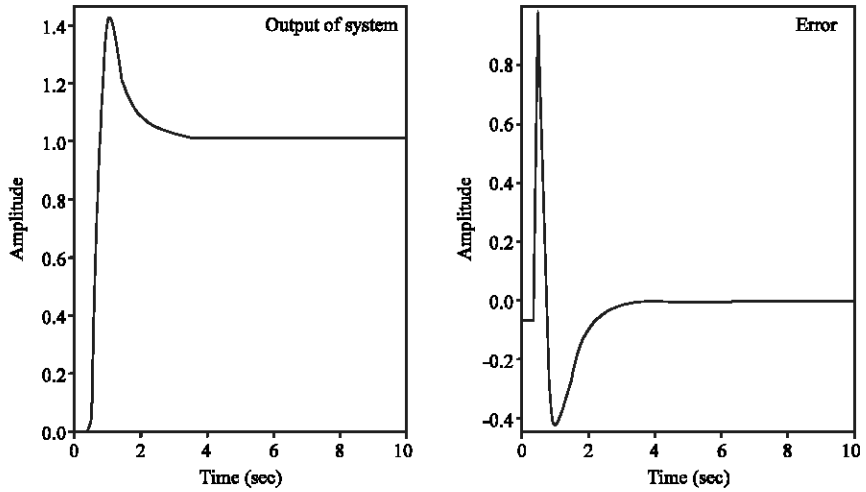

Fig. 16: Step response and error for mode 2

The second mode response is shown in Fig. 16. System response in this mode has achieved its stable state and the error is less than 2%. The difference between this mode and the first mode is more obvious. The system response involves an overshoot of 40%. In this mode the vibration amplitude level is reached to 0.0001 mm.

It is observed that the hybrid controller has a good adaptation with the dynamic of the system. In the second mode, frequency of vibration, with respect to first mode is doubled, but amplitude of the tip deflective is limited to 0.0001 mm.

As you have seen, at first in order to make the system step response stable and improve it, we used the PID controller, but in this part by combining the PD controller and FNN network, it achieved the demanded state.

Simulation of sliding mode control using gaussian radial basis function neural networks: We applied Sliding Mode Control Using Gaussian Radial Basis Function Neural Networks which is designed in material and methods part to our system. The output of the system after using this controller is the unit step. Figure 17 and 19 shows the various states of system's response in modes 1 and 2. Regarding these figures we found that after using this controller all sates are stable. The oscillations in output will be removed by using this controller. Figure 18 and 20 show the output of the system and the control signal that was applied to the system for mode 1 and 2. The amplitude of output without any oscillation is one and the amount of controlling signal is finite.

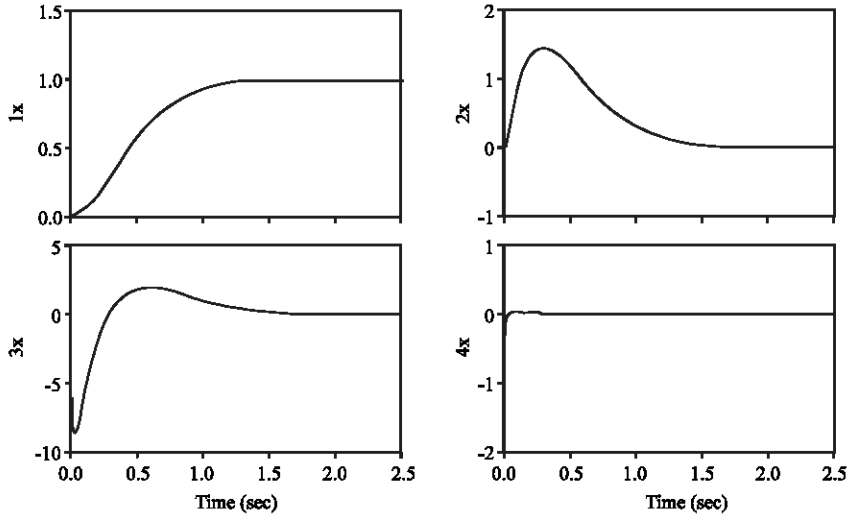

Fig. 17: State of flexible joint with sliding mode control using Gaussian RBFNN (mode 1)

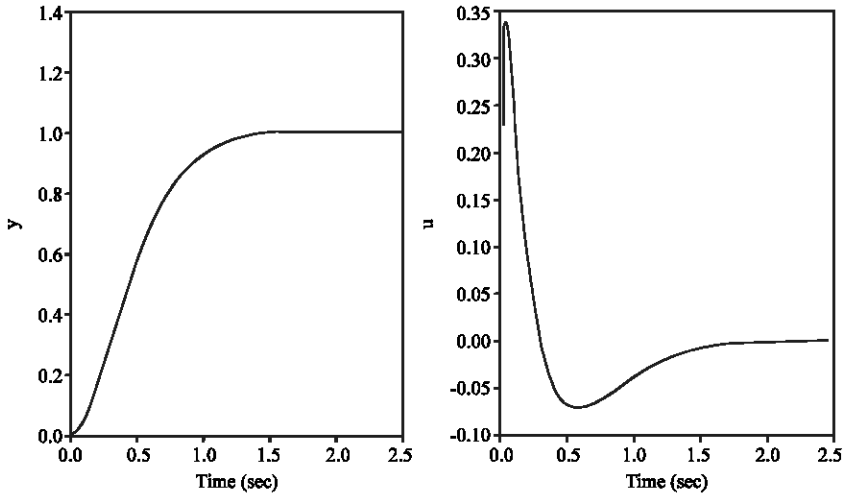

Fig. 18: Output of flexible joint (y) and control effort (u) with sliding mode control using Gaussian RBFNN (mode 1)

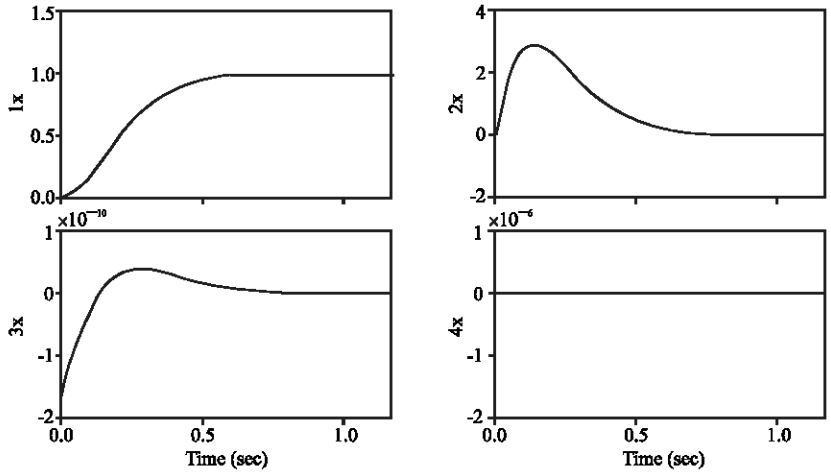

Fig. 19: State of flexible joint with Sliding Mode Control Using Gaussian RBFNN (Mode 2)

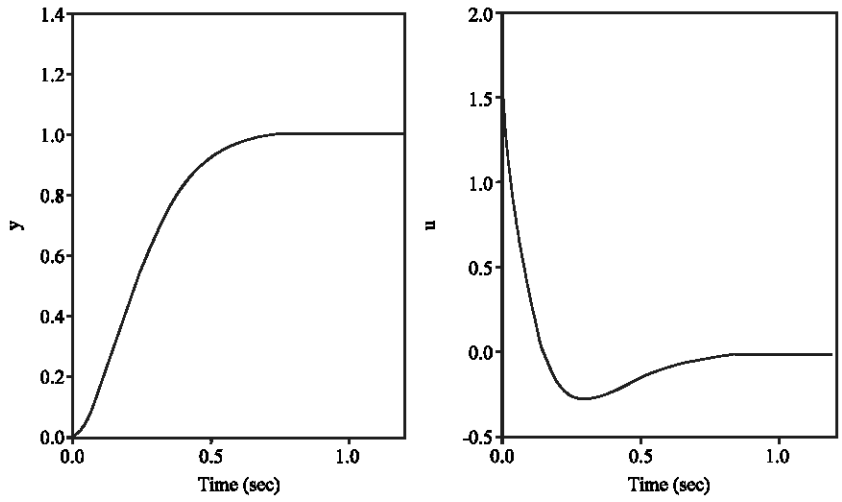

Fig. 20: Output of flexible joint (y) and control effort (u) with sliding mode control using Gaussian RBFNN (mode 2)

During 1 sec the output reaches to unit step without any overshoot. The other controller's output is consisted of overshoot and the output reaches to desired value during more than 1 sec.

After comparing all of applied controllers, we found that using sliding mode controller for flexible is more appropriate.

CONCLUSION

The designed controller consists of FNN network and conventional controller, PID. The FNN parameters are being trained by FEL technique. Also, LQR controller and sliding mode control using Gaussian radial basis function neural network are designed. The PID and FNN controllers were applied for the first two modes of vibration of the flexible arm. This controller has the ability to adapt with system dynamic and it can also produce a robust system output, against every kind of disturbance. The results of simulation show that system tracks the step input, in a good way. If we want to compare the estimated results derived in previous section with one of the main references, best of them is Tian and Collins (2005). In this study, tip deflection is controlled at about 0.0004 mm, thus, in previous section tip deflection was derived, 0.00001 mm. also, overshoot is reduced up to 28%, but in this reference, overshoot is reduced to 35%. The system steady state error is limited in about 1%. Other point is that, state feedback in PID controller designing is omitted and the controller is converted to a simple PD. All above results indicate that, the hybrid controller design in FEL figuration is one of the most practical methods in controller designing.

Sliding controller which is applied for this system is appropriate. Because, the internal dynamics in this system will be stable and also the output would not have any oscillations and there is no overshoot in the system's response and during 1.5 sec the output reaches to the desired value.

Comparing this controller with other controllers like PID, FNN, LQR, which are applied in this paper and last methods (Tian and Collins, 2005), shows that this controller has the best performance.

ACKNOWLEDGMENTS

The help of M. Naimi, V. Namazikhah and A. Vosolipour is gratefully acknowledged.

REFERENCES

Cheng, W. and J.T. Wen, 1993. A neural controller for the tracking control of flexible arms. IEEE Int. Conf. Neural Networks, 2: 749-754.

Donne, J.D. and U. Ozguner, 1994. Neural control of a flexible-link manipulator. IEEE Int. Conf. Neural Networks, 4: 2327-2332.

Korayem, M.H. and A. Basu, 1994. Formulation and numerical solution of elastic robot dynamic motion with maximum load carrying capacity. Robotica, 12: 253-261.

Lee, C.H. and C.C. Teng, 2000. Identification and control of dynamic systems using recurrent fuzzy neural networks. IEEE Trans. Fuzzy Syst., 8: 349-366.

Miyamura, A. and H. Kimura, 2002. Stability of feedback error learning scheme. Syst. Control Lett., 45: 303-316.

Nakanishi, J. and S. Schaal, 2004. Feedback error learning and nonlinear adaptive control. Neural Networks, 17: 1453-1465.

Newton, R.T. and Y. Xu, 1993a. Neural network control of a space manipulator. IEEE Control Syst. Magazine, 13: 14-22.

Newton, R.T. and Y. Xu, 1993b. Real-time implementation of Neural Network learning control in a flexible space manipulator. IEEE Int. Conf. Robotics Automation, 1: 135-141.

Önder Efe, M., O. Kaynak, X. Yu and B.M. Wilamowski, 2001. Sliding mode control of nonlinear systems using gaussian radial basis function neural networks. Int. Joint Conf. Neural Netwoks, 1: 474-479.

Slotin, J.J.E. and W. Li, 1991. Applied Nonlinear Control. 1st Edn., Prentice Hall, New York.

Talebi, H.A., K. Khorasani and R.V. Patel, 1998. Neural network based control schemes for flexible-link manipulators: Simulations and experiments. Neural Networks, 11: 1357-1377.

Tian, L. and C. Collins, 2005. Adaptive neuro-fuzzy control of a flexible manipulator. Mechatronics, 15: 1305-1320.

Topalov, A.V. *et al.,* 1998. Fuzzy-net control of non-holonomic mobile robot using evolutionary feedback-error-learning. Robotics Autonomous Syst., 23: 188-200.

Torfs, D.E., R. Vuerinckx, J. Swevers and J. Schoukens, 1998. Comparison of two feed forward design methods aiming at accurate trajectory tracking of the end point of a flexible robot arm. IEEE Trans. Control Syst. Technol., 6: 2-14.

Zhou, Y. *et al*., 2002. A new fuzzy neural network with fast learning algorithm and guaranteed stability for manufacturing process control. Fuzzy Sets Syst., 132: 201-216.