Neuro Controller For Time Varying Dynamical System.

Yousif. I. Al-Mashhidiny.
University of Anbar
Engineering College – Electrical Department.
M.SC 1999 yousif_unv@yahoo.com.

Abstract.

The feasibility of using an Artificial Neural Network (ANN) for controlling time- varying dynamical system is presented. The direct adjusting of neural controller by direct adaptive control (DAC) is available, by using the error between output of plant and desired input. The finite recurrent back propagation (FRBP) is used in the learning process, because the ability of this method to capture the nonlinearly and overcome the problem of time varying system. Hybrid controller structure used in this paper, where the parameters of classical controller are adjusted with time at specified freezing points for time varying dynamical system, and summed the outputs of two controllers and enter to the plant, identify of system by ANN to get the optimal initial condition for neuro controller.

A single channel for Spacecraft model is used as an example in this paper, satisfactory results are obtained, which explain the ability of recurrent neural network (RNN) to identify time varying dynamical system and overcome for all its problem and explain the ability of this structure of hybrid neuro controller to use with time varying dynamical system.

Keyword. Finite recurrent backpropagation network(FRBN), time varying system, direct neural model reference adaptive system (DNMRAS).

1- Introduction.

Model Reference Adaptive Systems or MRAS have been adopted by many researchers in controlling nonlinear plants. Such approach only requires the input and output measurements of the system and is, thus, congruous for plants where mathematical models are unavailable or difficult to obtain. In addition to this advantage, the stability of the system is somehow assured through the convergence of both the states and parameters of the plant and the reference model In direct MRAS, the controller parameters are directly adjusted to reduce some norm of the output error between the plant output and the desired reference trajectory.

Neural Network systems, on the other hand, are highly compelling for controlling nonlinear dynamic systems with unknown parameters. The integration of neural methods effectuates an excellent learning and flexible knowledge representational capability. The data driven neuro systems meliorate heuristic procedures or expert knowledge in designing the neuro control rules which has been a drawback in conventional neural systems. On the other hand, neural network is suitable to be used in solving nonlinear identification and control problems involving complex plants especially when forming a mathematical model of the system is tedious or not possible. Employs. The backpropagation algorithm is used for learning the parameter identification which provide the consequent part of the rules [1].

In this paper, a direct Neural Model Reference Adaptive Controller (NMRAC) is proposed for nonlinear systems with unknown parameters.. The control strategy used to define the adaptation law is based on the tracking error between the actual plant output and target output, which is the direct reference signal. Then, tuning of the parameters of neural

network is based on the standard delta rule or steepest descent algorithm to minimize the tracking error. This algorithm is preferred since the weights update is governed by the first derivative of the error, and thus produces faster rate of convergence, consistent training and not getting stuck in local minima. The system structure identification mechanism that includes the technique for input data partitioning, automatic generation of rules and tuning of its parameters based on observed input-output data of a reference signal, then using an artificial neural network as an adaptive direct hybrid controller to control a simple time varying dynamical system. Four different learning architectures based on the neural network approach have been proposed by Psaltis et. al [2] and G.W. Irwin [3]: they are indirect learning, general leaning, direct learning and specialized learning architectures. **Fig. (1)** represents direct adaptive hybrid controller.

Two main methods exist for providing a neural network with dynamic behavior: the insertion of a buffer somewhere in the network to provide an explicit memory of the past inputs, or the implementation of feedbacks. As for the first method, it builds on the structure of feedforward networks where all input signals flow in one direction, from input to output. Then, because a feedforward network does not have a dynamic memory, *tapped-delay-lines* (temporal buffers) of the inputs are used. The buffer can be applied at the network inputs only, keeping the network internally static as in the buffered multilayer perceptron (MLP) or at the input of each neuron as in the MLP with Finite Impulse Response (FIR) filter synapses (FIRMLP). The main disadvantage of the buffer approach is the limited past-history horizon, which needs to be used in order to keep the size of the network computationally manageable, thereby preventing modeling of arbitrary long time dependencies between inputs and outputs It is also difficult to set the length of the buffer given a certain application [4].

The second method, the most general example of implementation of feedbacks in a neural network is the recurrent neural network constituted by a single layer of neurons fully interconnected with each other or by several such layers Because of the required large structural complexity of this network, in recent years growing efforts have been propounded in developing methods for implementing temporal dynamic feedback connections into the widely used multi-layered feedforward neural networks. Recurrent connections can be added by using two main types of recurrence or feedback: *external* or *internal*. *External recurrence* is obtained for example by feeding back the outputs to the input of the network, while the *internal recurrence* is obtained by feeding back the outputs of neurons of a given layer in inputs to neurons of the same layer, giving rise to the so called *Locally Recurrent Neural Networks* (*LRNNs*) [5].

Recurrent neural networks (RNNs) are widely acknowledged as an effective tool that can be employed by a wide range of applications that store and process temporal sequences. The ability of RNNs to capture complex, nonlinear system dynamics has served as a driving motivation for their study. RNNs have the potential to be effectively used in modeling, system identification, and adaptive control applications. The proposed RNN learning algorithms rely on the calculation of error gradients with respect to the network weights. What distinguishes recurrent neural networks from static, or feedforward networks, is the fact that the gradients are time dependent or dynamic. This implies that the current error gradient does not only depend on the current input, output, and targets, but rather on its possibly infinite past.

2- System Identification.

Feedforward neural networks are composed of neurons in which the input layer of neurons is connected to the output layer through one or more layers of intermediate neurons،The training process of neural networks involves adjusting the weights till a desired

input/output relationship is obtained. The majority of adaptation learning algorithms are based on the fast momentum backpropagation the mathematical characterization of a multilayer feed forward network is that of a composite application of functions each of these functions represents a particular layer and may be specific to individual units in the layer, e.g. all the units in the layer are required to have same activation function. The overall mapping is thus characterized by a composite function relating feed forward network inputs to output. That is $O=f_{composite}(x)$. Using (p) mapping layers in a (p+1) layer feed forward net yield:
$O=f^{Lp}(f^{Lp-1}\ldots\ldots(f^{L1}(x).....))$. Thus the interconnection weights from unit (k) in L_1 to unit (I) in L_2 are w^{L1-L2} . If hidden units have a sigmoidal activation function, denoted f^{sig} [6].

$$O_i^{L2} = \sum_{k=1}^{Hi} w_{ik}^{L1\to L2} \left\{ f_k^{sig} \left[\sum_{j=1}^{I} w_{kj}^{Lo\to L1} i_j \right] \right\} \quad (1)$$

Where: O_i^{L2} is the o/p neuron i in $L2$. H_i is number of neurons which connect with neuron i . L_0, L_1, L_2 represents the layers of N.N.

Above equation illustrates neural network with supervision and composition of non-linear function.The learning is a process by which the free parameters of a neural network are adapted through a continuing process of simulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameters changes take place. A prescribed set of well defined rules for the solution of a learning problem is called learning algorithm. The Learning algorithms differ from each other in the way in which the adjustment kj, Δw to the synaptic weight w_{kj} is formulated. The basic approach in learning is to start with an untrained network.

The network outcomes are compared with target values that provide some error. Suppose that $\mathbf{t}_k(n)$ denote some desired outcome (response) for the K^{th} neuron at time n and let the actual response of the neuron is $O_k(n)$. Suppose the response $y_k(n)$ was produced when x(n) applied to the network. If the actual response $y_k(n)$ is not same as $\mathbf{t}_k(n)$ ، we may define an error signal as: $\mathbf{e}_k = \mathbf{t}_k(n) - \mathbf{y}_k(n)$.
The purpose of error-correction learning is to minimize a cost function based on the error signal $\mathbf{e}_k(n)$. Once a cost function is selected, error-correction learning is strictly an optimization problem. A cost function usually used in neural networks is mean square error criteria called L.M.S learning.

$$J = E\left[\sum_k e_k^2(n) \right] \quad (2)$$

Here summation runs over all neurons in the output layer of the network. This method has the task of continually search for the bottom of cost function in iterative manner Minimization of the cost function J with respect to free parameters of the network leads to so-called Method of Gradient Descent [8].

$$w_{k,(n+1)} = w_{k,(n)} - \eta \nabla J_{(n)} + m_u \Delta w_{k,(n-1)} . \quad (3)$$

The proportion of error for weight updating (correction). The learning parameter has a profound impact on the performance of convergence of learning. A plot of the cost function versus the synaptic weights characterizes the neural network consists of a multidimensional surface called error surface. The neural network consists of cross-correction learning algorithm to start from a n arbitrary point on the error surface (initial weights) and then move
towards a global minima, in step by step fashion. We can gain understanding and intuition about the algorithm by studying error surfaces themselves the function J(w), **Fig.(2**) show the error surface. Such an error surface depends upon the task in hand, but even there are some general properties of error surfaces that seem to hold over a broad range of real-world pattern recognition problems.

In case of non-linear neural network, the error surface may have troughs, valleys, canyons, and host of shapes, where as in low dimensional data, contains many minima and so many local minima plague the error landscapes, then it is unlikely that the network will find the global minimum. Another issue is the presence of plateaus regions where the error varies only slightly as a function of weights see and. Thus in presence of many plateaus، training will get slow. To overcome this situation momentum is introduced that forces the iterative process to cross saddle points and small landscapes.

Neural network training begins with small weights; the error surface in the neighborhood of w≈0will determine the general direction of descent. High dimensional space may afford more ways dimensions for the system to 'get around' a barrier or local maximum during training. For large networks, the error varies quite gradually as a single weight is changed. For networks having differentiable activation functions, there exist a powerful and computationally efficient method, called error backpropagation for finding the derivatives of an error function with respect to the weights and biases in the network.Gradient Descent algorithm is the most commonly used error backpropagation method.

Standard backpropagation is a gradient descent algorithm, in which the network weights are moved along the negative of the gradient of the performance function. There are a number of variations on the basic algorithm that are based on other standard optimization techniques, such as conjugate gradient. Backpropagation was created by generalizing the learning rule to multiple-layered networks with nonlinear differentiable transfer functions which is the gradient of the pattern error with respect to weight w_k and forms the basis of the gradient descent training algorithm. Specifically, we assign the weight correction, Δw_k، such that [7].

$$w_k^{j+1} = w_k^j - \eta \left[\left(O^p - t^p \right) \frac{df\left(net^p\right)}{dnet^p} . x_k^p \right] \quad (4)$$

Mostly, the learning algorithms involve a sequence of steps through weight space. We can consider each of these steps in two parts. First decide the direction in which to move and secondly, how far to move in that direction. This direction of search provides minimum of the error function in that direction in weight space. This procedure is referred to as line search and it forms the basis for several algorithms which are considerably more powerful than gradient descent. Suppose at step n in some algorithm the current weight vector is w_n and we wish to obtain a particular

search direction p_n through weight space. The minimum along the search direction then gives the next values for the weight vector as:

$$w_{n+1} = w_n + \lambda_n p_n \tag{5}$$

This gives us an automatic procedure for setting the step length, once we have chosen the search direction. The line search represents a one dimensional minimization problem. This minimization can be performed in a number of ways. A simple approach would be to

proceed long the search direction in small steps, evaluating the error function at each step (position), and stop when the error starts to increase. It is possible, however, to find much
more efficient approaches. This includes the issue that whether local gradient information is preferable in line search. The search direction is towards the minimum is obtained (possibly)
through proper weight adjustments, this involves search through negative direction of gradient information. To apply line search to the problems of error function minimization, we
need to choose a suitable search direction at each stage of the algorithm. Note that at minimum of the line search

$$\frac{\partial}{\partial \lambda} E(w_n + \lambda p_n) = 0 \tag{6}$$

Thus the gradient of the new minimum is orthogonal to the search direction. Choosing successive search directions to the local (negative) gradient directions can lead to the problem of slow learning. The algorithm can then take many steps to converge, even for a quadratic error function. The solution to this problem lies in choosing the successive search directions d_n to minimize cost function such that, at each step of the algorithm, the component of the gradient parallel to the previous search direction which has just been made zero, is unaltered Suppose, we have already performed a line minimization along the direction d_n , starting from the point w_n , to give the new point w_{n+1} [8].

3- Control Architecture.

In direct MRAS, the controller parameters are directly adjusted to reduce some norm of the output error between the plant output and the desired reference trajectory. The controller structure used in this paper illustrated in **Fig.(3)** where the controller represent by hybrid controller between classical controller and neural controller, the outputs of the two controllers will be summed and entered to servo[9].

The input to classical controller is the error signal and the parameter of it adjusting at freezing point because the varying in system, the adjusting of parameters in classical controller achieved firstly. Error signal and its derivative forms the input to neural controller and, can be add any state from the system. The adjusted the weights achieved by FRBP, and the structure of N.N controller illustrated in **Fig.(4)** (see ref 10). It consist of i/p Layer (I), hidden Layer (H) and output Layer (O). The weight matrix consists of wI_{mxi} and wO_{nxm}, the characteristic of the network is that the units in hidden layer are connected with themselves and each other. However, the information in such units at one time can only be transferred until being replaced by

new data after finite steps, i.e. the units in hidden layer only memorize the information for finite time which is the key difference from the fully recurrent BP. For the sake of this, the network here is called finite recurrent back propagation network. Hence at the time (t) the input to the (ith) hidden units is:

$$AHi(t)=\sum_{k=1}^{I} wT_{ik}\, I_k + \sum_{k=1}^{m} wH_{ik}\, f\,(AH_k(t-1)); \qquad i=1,2.....m \tag{7}$$

The activation function (f) at the o/p layer of controller is a tangential, while the others activation function (g) is sigmoidal that is:

$$f(x)=\frac{1}{2}+\frac{\left(1-\exp(-x)\right)}{\left(1+\exp(-x)\right)},\cdots,g(x)=\frac{1}{2}+\frac{1}{\left(1+\exp(-x)\right)} \tag{8}$$

The output of the network is a weighted sum of the hidden unit o/p's:

$$O_i(t)=\sum_{k=1}^{m} wO_{ik}\, f\left(AH_k(t)\right); \qquad i=1,2,....n \tag{9}$$

The net is trained by minimization of the total error, which is evaluated as that:

$$\mathrm{E(t)}=\sum_{\mathrm{p=1}}^{\mathrm{pp}}\left(\frac{1}{2}\sum_{\mathrm{k=1}}^{\mathrm{n}}\mathrm{e_{kp}(t)^2}\right) \quad \text{and} \quad E(t)=\sum_{p=1}^{pp}\left(\frac{1}{2}\sum_{k=1}^{n}\left(T_{kp}(t)-O_{kp}(t)^2\right)\right) \tag{10}$$

Where; (pp) is the sample length, $T_{kp}(t)$ are the target values that the output of the (k_{th}) unit in output layer while inputting the (pth) sample match at time (t). This can be fulfilled by a gradient descent procedure adjusting (w) a long the negative of $\nabla w\, E(t)$. the weight change for network by FRBP algorithm which shown in [10].

4- Case Study.

The system using here represents single channel in spacecraft, with transfer function [11]:

$$\mathrm{F_{\delta}^{\alpha}(S)}=\frac{\mathrm{A_1S^2+A_2S+A_3}}{\mathrm{B_1S^2+B_2S+B_3}}; \tag{11}$$

Where, the coefficient (A1, A2…..B3) are calculated as function of time and the input (δ) to dynamic system is the deflection of elevator, we used the lookup table for these parameters and can be get at freezing point numeric transfer function is:

$$F_{\delta}^{\alpha}(S)=\frac{-310.\mathrm{S^2}-2870\mathrm{S}-227640}{\mathrm{S^4}+120\mathrm{S^3}+1950\mathrm{S^2}} \tag{12}$$

the transfer function for servo used here is:

$$F_{\delta e}^{\delta}(S)=\frac{k}{CS^2+DS+1} \tag{13}$$

Where k=2 , C= 0.01 and D = 0.075

The classical controllers used for this system with period of time (0 to 140 sec) adapted at specified freezing points on all the time are given by the following transfer function:

$$\left.\begin{aligned} F_{\alpha e}^{\delta c}(S) &= \frac{30S^2+6S+70}{0.5S^2+10S} && (\text{time } 0\rightarrow 30\,\text{sec.}).\\ &= \frac{80S^2+0.7S+10}{0.7S^2+1.5S+0.15} && (\text{time } 30\rightarrow 90\,\text{sec.}).\\ &= \frac{80S^2+0.7S+10}{0.5S^2+0.15S} && (\text{time } 90\rightarrow 140\,\text{sec.}). \end{aligned}\right\} \tag{14}$$

The input to neural controller represents by vector $Xp=[\alpha_e \;\; \dot{\alpha}_e]$; where it represents by error signal and its derivative (see **Fig.(3)**), and the N.N is multi-layer network (2-20-1) with learning rate (α_r) equal to (0.009) and momentum term (mu) equal to (0.0025). The two outputs (neural and classical controller O/PS) are summed and entered to servo:

$$\delta e = F(\alpha p)+\delta c \tag{15}$$

The complete nonlinear single channel controller is shown in **Fig.(5).** The input to the system is taken as unit step, the simulation results by using Matlab software package ver.8a,[12]; it's can be summarized as:

* identification part which shown in **Fig.(6)** ,from it can be illustrated the ability of FRBP network to identify the response of time varying . nonlinear dynamic system and shown the ability to adjustment and tuning its parameters to get high accuracy identify results and overcome for main problem in identification of this type of system (i.e: the local and global minimum that shown in **Fig.(2)** , where we get error approach to zero after 100 iteration for identification and used this results as initial condition for running controller in the next step.

* the limitation of deflection of elevator of plant (delta) about (± 20º , i.e about 0.35 rad) from **Fig.(7)** it can de seen that the o/p of elevator is fulfill this constrain and this done without using the limiting part in classical approach this results get with N.N controller with tangential activation function in the o/p layer.

* In the hybrid controller with DNMRAC the affect of high accuracy of identify appear at the choose of initial condition of neural controller and the values of parameters of classical controller .From **Fig's.(8 9 & 10)** it can be seen the efficiency of this type of controller for dealing with the time varying nonlinear dynamical system and get the desired response with very accepted rising time ,steady state error and minimum overshot, where we check the system with many types of inputs and it

overcome on the problem of choose the reference model where the choose of desired o/p is very , where we used the input signal is the desired signal.

5- Conclusions.

From the results it can be illustrated that the ability of direct hybrid controller with FRBP network to treat the time varying dynamic system and forced it to followed the desired input. The FRBP Network has high speed learning to get the desired error, this property offer for the neural controller to treatment many types of system and get desired response, where the main problem of time varying dynamic system the variation of transfer function with time , this constrain need to solving the problem by special N.N, where it offers high response with any types of input signal and depend the instantaneous state of system as initial condition to next state all these requirements gets in FRBP network.

6- References.

[1] M.S. Lan, "Adaptive Control of Unknown Dynamical System Via Neural Network Approach", Rockwell International science center thousand Daks. Ca 91360.2004., pp. 910-
914

[2] D. psaltis, A. sideris and A.A. Yamamura, "A Mult-Layered Neural Network Controller",
IEEE control system magazine, April 1998 pp. 17-21.

[3] G. Lightbody and G.W. Irwin, "Direct Neural Model Reference Adaptive control", IEEE
proce, control theory and its application vol. 142 No. 1 Jan 1995, pp 32-43.

[4] L.Xiaoou , Y,Wen, " Dynamic System Identification Via Recurrent Multilayer Perceptrons "
, Information Sciences 147 , 2002 , 45-63.

[5] F.Cadini, E. Zio, N. Pedroni - "Recurrent Neural Network For Dynamic Rellability
Analysis".R&RATA # 2 (Vol.1) 2008, June.

[6] S.M.Buney, T.A.Jilani, G.Ardil, "A Comparison of First and Second Order Training
Algorithms of Artifical Neural Networs ", vol.1 Jan.2005 ISSN 1307-6884.

[7] L Wu, P Baldi. "Learning to play Go using recursive neural networks" science direct neural
network 21 February 2008

[8] Z. Liu and I. Elhanany, "A Fast and Scalable Recurrent Neural Network Based on Stochastic
Meta Descent". IEEE transaction on neural network, vol. 19, no. 9, september 2008

[9] K.J. Astrom, "Adaptive Feed back Control" proceedings of the IEEE vol. 75- No. 2. February
1987 pp. 185-217.

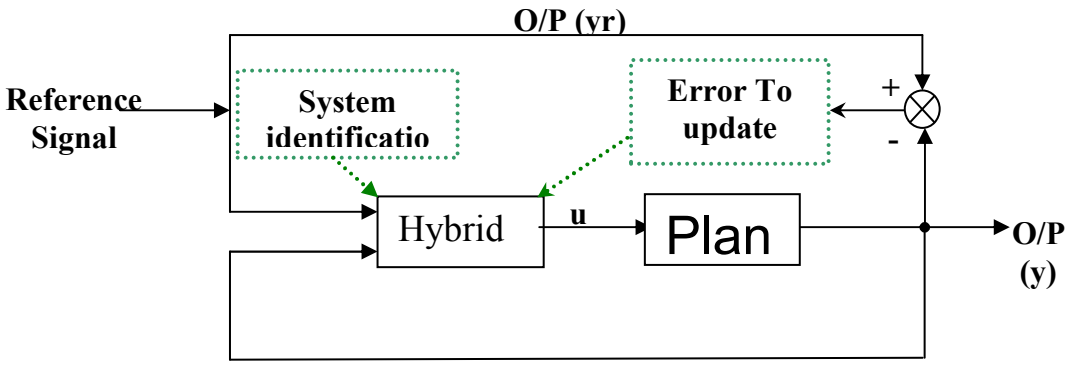
[10] F. Tong, X. Zheng, Z.Ling and S. Lian, "On Finite Recurrent Neural Network", IEEE in
ternational symposium on speech, Image. Processing and neural Network, April 1994, pp.
13-16 Hong Kong.

[11] L. Jack, " Spacecraft Aerodynamics". First Ed. Newyork 1960.
[12] Matlab for windows Var.8a- 2008 Programming Language.

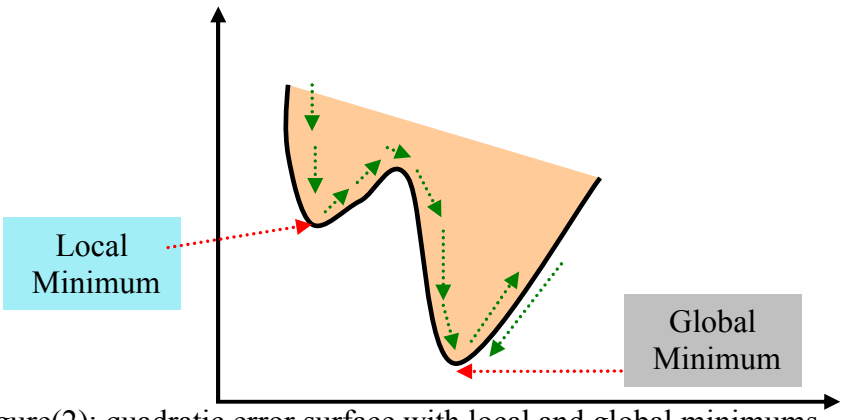
7- Nomenclature.

(α) : input signal to classical controller.
(α_d) : desired input signal.
(α_e) : error signal between o/p plant and reference signal.
(δ) : input signal to plant.
(δ_c) : o/p for classical controller.
(δ_e) : summation signals of classical and neural controllers.
(η) : learning rate in identifier network.

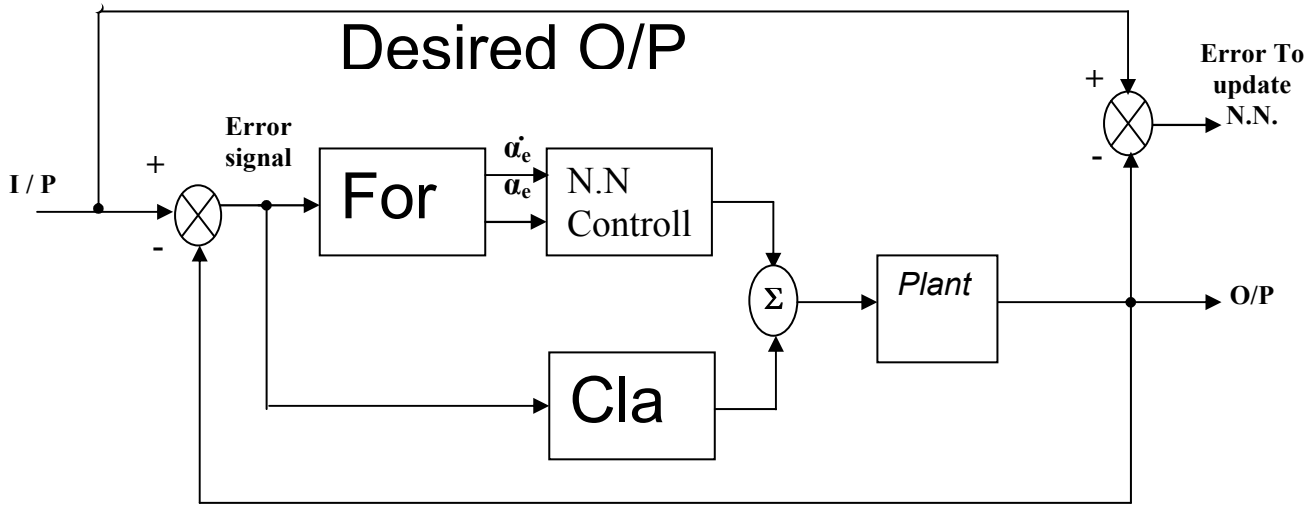
(e_k) : error signal in neuron.
(mu) : momentum term in learning of identifier network.
(t_k) : target signal for neural network.
(pp) : No. of sample in training network.
(O_k) : o/p of neurons in network.
(w_{ik}) : weight between input layer to hidden layer
(w_{ko}) : weight between hidden layer to output layer



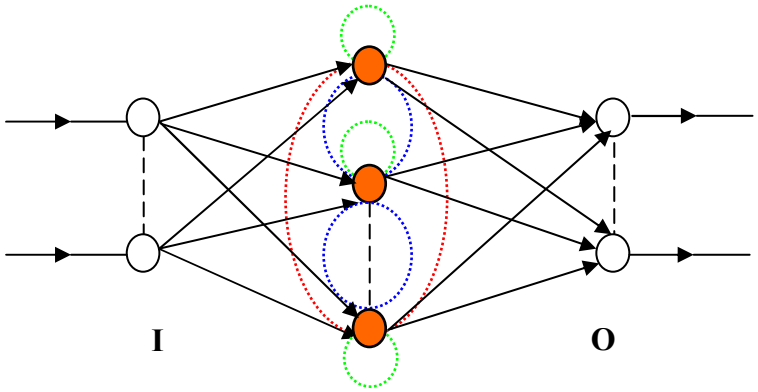
Figure(1): direct adaptive hybrid controller.



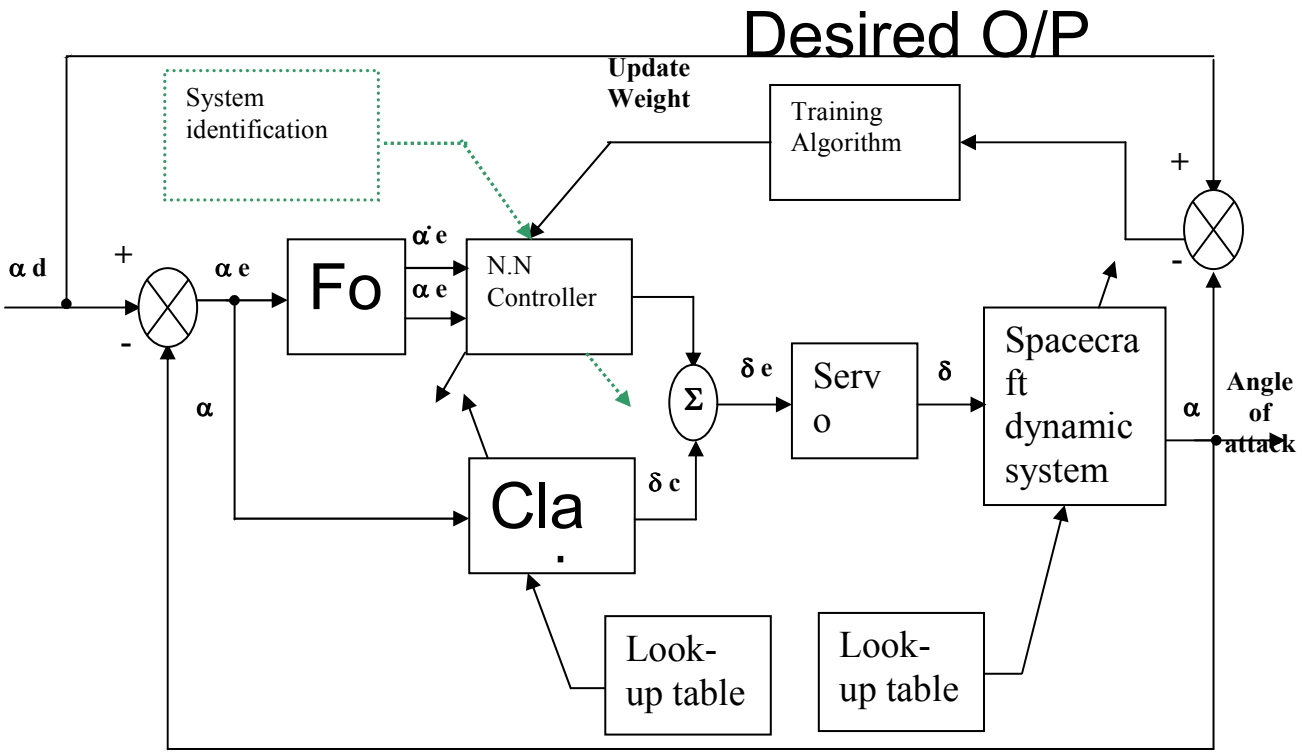
Figure(2): quadratic error surface with local and global minimums.



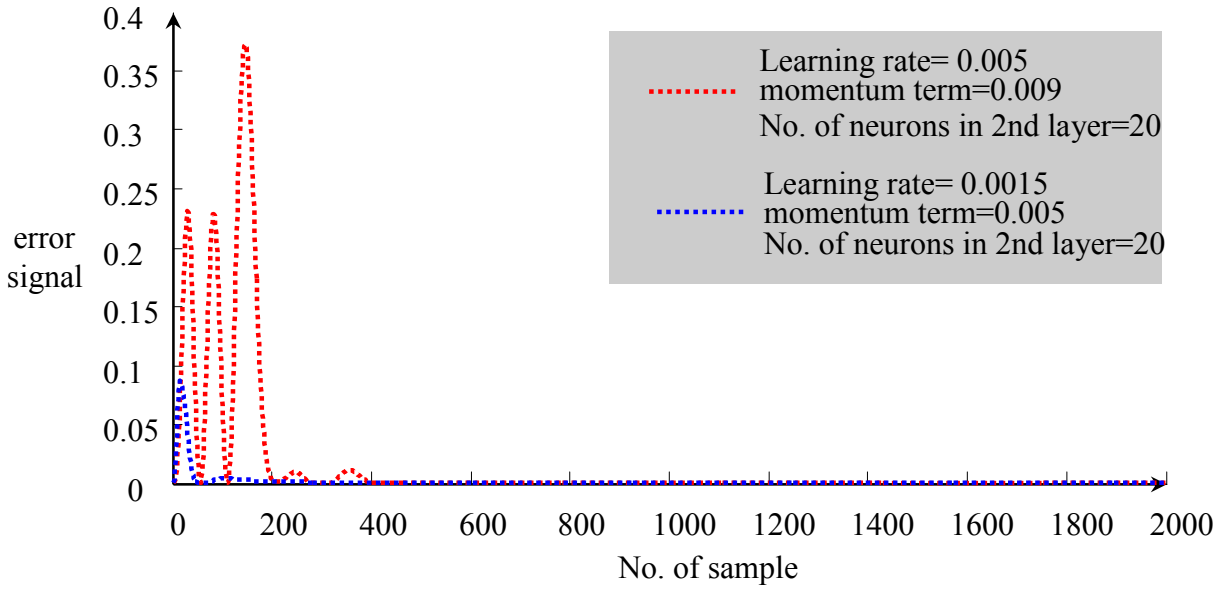
Figure(3): hybrid controller structure.



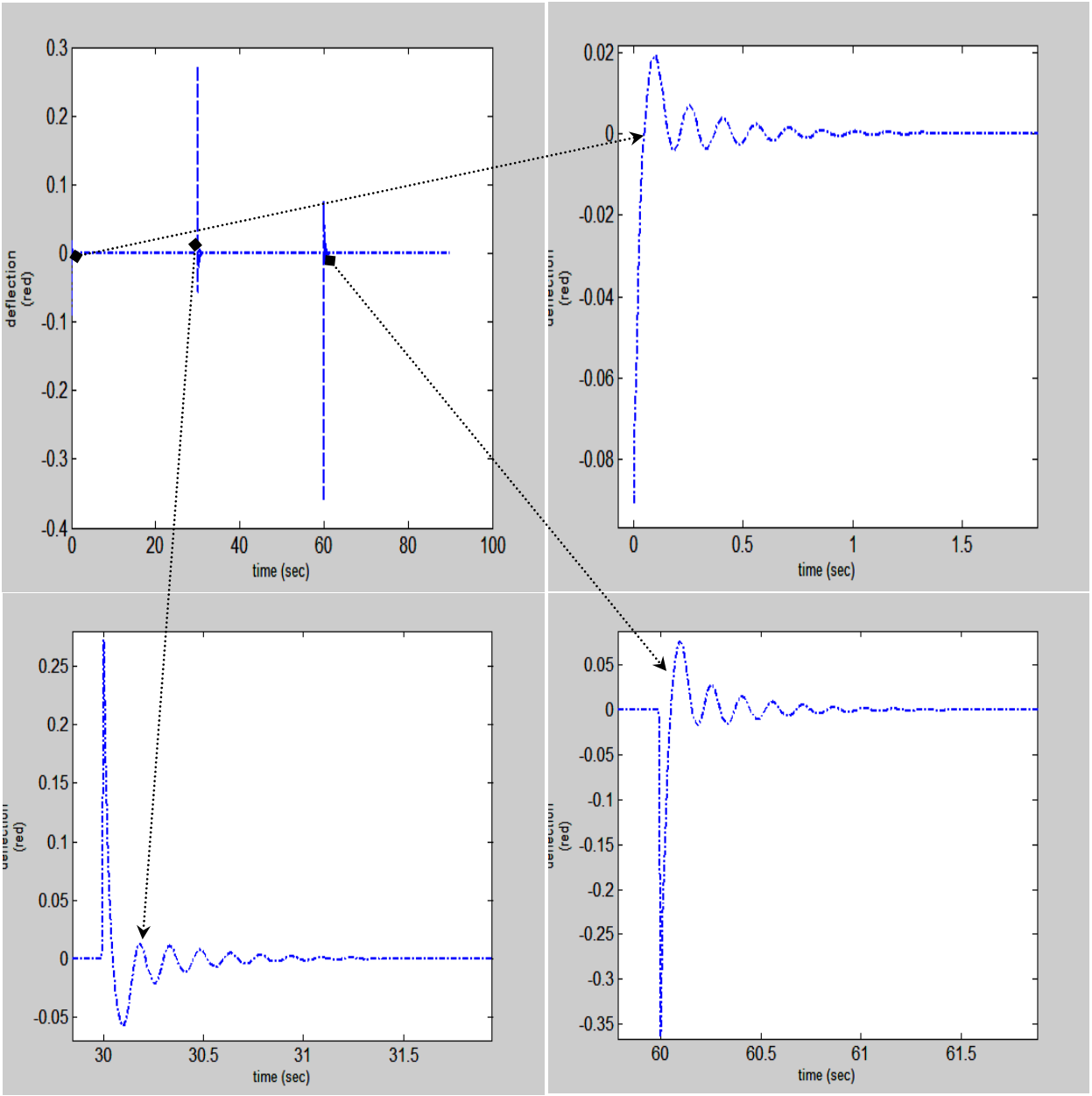
Figure(4): FRBP Network with (I) Input, one Hidden Layer of (m) Units and (n) Outputs.
(The Dotted Curve Lines Denote the Finite Recurrent Connection).



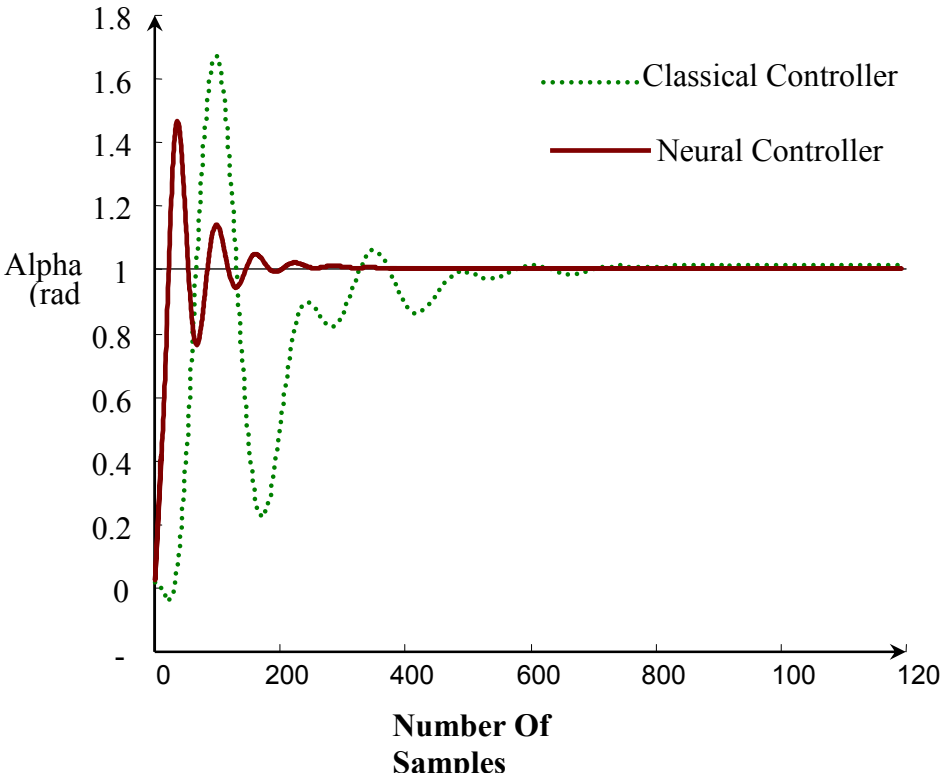
Figure(5): Complete neural controller for simple channel for spacecraft.



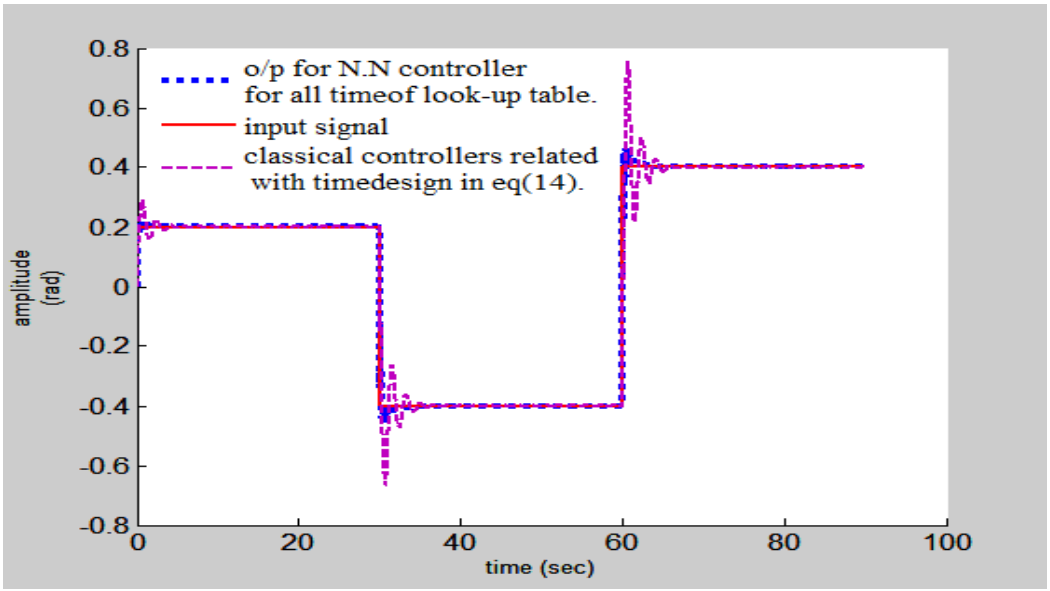
Figure(6): Identification system and adjusting its parameters .



Figure(7): The deflection of elevator (δ) with respect to time.



Figure(8): comparison between classical controller and neural controller by FRBP network.



Figure(9): The o/p of Neural controller and classical controller for all time.

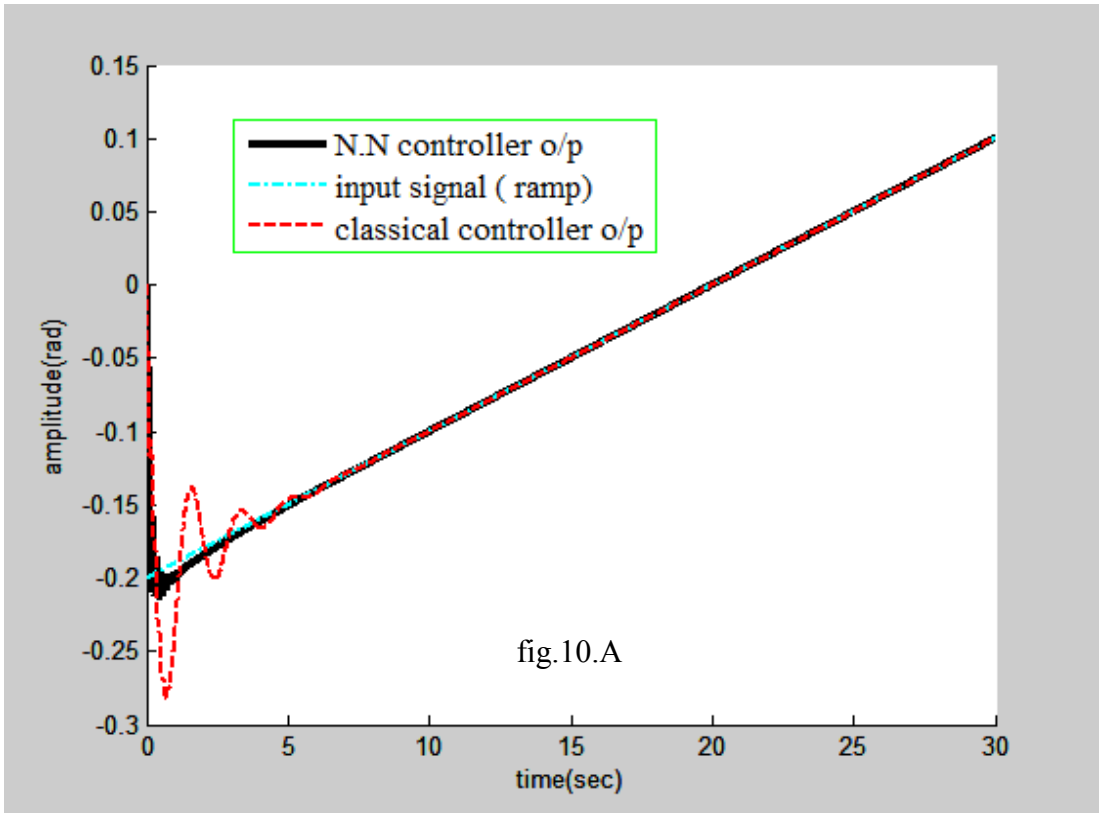


fig.10.A

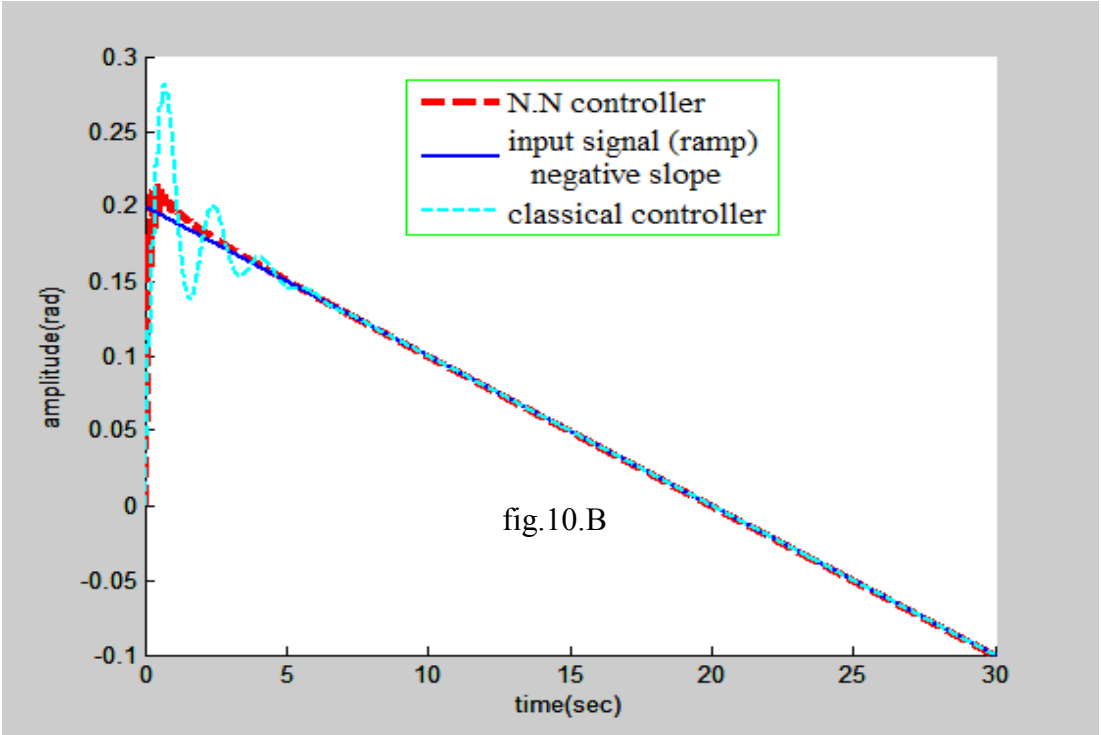


fig.10.B

Figure(10): the response of two controller with ramp input signal (A) positive slope,(B)negative slope.

مسيطر عصبي للأنظمة الديناميكية المتغيرة مع الزمن

المدرس المساعد
يوسف إسماعيل محمد المشهداني
جامعة الانبار – كلية الهندسة – قسم الهندسة الكهربائية.

الخلاصة.

تعرض المقالة إمكانية الشبكات العصبية ذات التعليم العكسي بخاصية الإرجاع العكسي (R.P.N.N) في السيطرة على الأنظمة الديناميكية المتغيرة مع الزمن، وذلك من خلال استخدام أنظمة السيطرة المتكيفة المباشرة (Direct Adaptive Control) بواسطة استخدام مسيطر هجين (Hybrid Controller) يتكون من المسيطر التقليدي ومسيطر عصبي يكون بناءه من الشبكات ذات التعليم العكسي المحدد (FRBP Network) وذلك لإمكانية هذه الشبكات في العمل كمسيطر لهذه الأنظمة وذلك من خلال أعادة ضبط الأوزان بواسطة عملية تعليم الشبكة للحصول على درجة التقريب المطلوبة مبينة من خلالها إمكانية هذة الشبكات على التغلب على أهم مشاكل تعريف المنظومات اللاخطية الديناميكية المتغيرة مع الزمن في عدم وجود دالة انتقالية تعرف سلوك النظام مع الزمن.

المثال المستخدم في هذه المقالة هو أحد قنوات الحركة في المركبة الفضائية، تم الحصول على نتائج تؤكد أمكانية استخدام هذا الأنموذج في السيطرة على الأنظمة الديناميكية المتغيرة مع الزمن حيث تم عرض نتائج التعريف للمنظومة ونتائج المسيطر العصبي الهجين مقارنة مع نتائج المسيطر التقليدي.