DESIGN OF ADAPTIVE FUZZY-NEURAL PID-LIKE CONTROLLER FOR NONLINEAR MIMO SYSTEMS

Dr. Mohammed Yousif Hassan
Lecturer

Mr. Qussay F. Ad'Doory
Asst. Lecturer

Received on: 12 / 6 / 2006
Accepted on: 16 / 1/2007

Abstract

A combination of fuzzy logic and neural network can generate a fuzzy neural controller which in association with a neural network emulator can improve the output response of the controlled system. This combination uses the neural network training ability to adjust the membership functions of a PID like fuzzy neural controller. Such controller can be used to adaptively control nonlinear MIMO systems.

The goal of the controller is to force the controlled system to follow a reference model with required transient specifications of minimum overshoot, minimum rise time and minimum steady state error. The fuzzy membership functions were tuned using the propagated error between the plant outputs and the desired ones.

To propagate the error from the plant outputs to the controller, a neural network is used as a channel to the error. This neural network uses the back propagation algorithm as a learning technique.

The controller was tested using two inputs / two outputs nonlinear time invariant model. Different reference (set-point) inputs were applied to the closed loop system. Also, different values of loads and disturbances were applied to the closed loop system. Simulation results show that the controller achieves the design requirements.

الخلاصة

من الممكن إستخدام خليط من المنطق المضبب والشبكات العصبيه لتكوين مسيطر عصبي مضبب قادر بالأشتراك مع مشابه عصبي على تحسين إستجابة تصرف الخرج لمنظومة السيطره . يستخدم هذا الخليط قدرة الشبكه العصبيه على التعلم لكي يعّدل في الدوال العضويه التابعه للمسيطر العصبي المضبب المشابه للمسيطر المتناسب-المتكامل-التفاضلي . ومن الممكن إستخدام مثل هذا المسيطر للسيطره المتكيفه على نظام لا خطي متعدد المداخل والمخارج . إن هدف هذا المسيطر هو إرغام منظومة السيطره على تتبع نموذج مرجعي مع مواصفات إنتقاليه مثل : أقل تجاوز للهدف وأقل زمن صعود وأقل خطأ مستقر ، لذلك تم تنغيم الدوال العضويه المضببه بأستخدام الخطأ الممتد الأتي من الفرق بين خرج النظام والخرج المرغوب به .

تم أستخدام شبكه مشابه عصبي كقناة لتجهيز الخطأ المنتشر من خرج النظام الى المسيطر العصبي المضبب وبأستخدام طريقة الأمتداد العكسي كتقنيه تعليم للشبكه . كذلك تم إختبار المسيطر بربطه مع نظام لا خطي ثابت مع الزمن ذو مدخلين ومخرجين بمسار مغلق . وقد تم إستخدام أكثر من إدخال مرجعي كإشاره دخل للمنظومه المغلقه . وقد تمت تجربة المنظومه المغلقه بوجود حمل وضوضاء وقد أظهرت نتائج الأختبارات أن المسيطر قد حقق متطلبات التصميم .

Keywords: **Neural networks, Fuzzy logic, PID controller, Nonlinear systems, MIMO systems.**

*Control and Systems Engineering Department University of Technology Baghdad, Iraq

1-Introduction

Proportional-Integral-Derivative (PID) is the most used controller type in industry. Its use is so diversified that the control engineer must tune the PID values according to specific needs [8,9]. However, developments in intelligent control had been made in the past years through the development of fuzzy logic control (FLC) and Neural Networks (NN) [9]. Many complex systems were controlled using these techniques such as robotic manipulators, motor drives … etc.

FLC techniques have been widely used in industrial processes, particularly in situations where conventional control design techniques have been difficult to apply. The main advantage of the Fuzzy Logic Controller (FLC) is that it can be applied to plants that are difficult to be described as a mathematical model, and the controller can be designed to apply heuristic rules that reflect the experience of human experts. PID FLCs have been successfully applied to a variety of practical problems. In spite of its practical success, there is no standard procedure for tuning PID FLCs. The design of most PID FLCs is a very time consuming activity involving, knowledge acquisition, definition of the control structure, definition of rules, and tuning a variety of gains and other parameters [10].

Artificial Neural Network (ANN) is a combination of processing elements that perform certain tasks through learning weights. The weights of the ANN are adjusted using certain algorithms. This ability of adjusting weights, is used to simulate the human learning ability so that a NN can learn to model or to control many systems efficiently [9]. However, Neural Networks provide a different approach to problem solving from linguistic or algorithmic systems such as FLC. They have two main features which are noise-resistant parallel distributed structure and their ability to learn from examples. NNs have a wide spectrum of actual and potential applications ranging from prediction to dynamic system modeling and control [10].

By combining both algorithms of NNs and FLCs together a robust controller may be achieved, which can give precise actions and learn to enhance its performance [9]. A FLC can represent human reasoning while NN can simulate human learning. Thus, a Fuzzy-Neural Network (FNN) can be defined as the network that consists of many simple fuzzy neurons, which perform some kinds of fuzzy operations, that in general combination, perform the reacquired tasks, as a controller, or emulator [14].

Many researchers worked with the design of neural fuzzy controllers. The method of fuzzy neurons was first studied in 1970 by Lee and Lee, but the 90^{th} was the start of fuzzy neural applications [8, 14]. In 1991, Hayashi et al. Introduced a NN that functions as a FLC to control a DC servo motor. They used the back-propagation algorithm to achieve learning of the controller [14] . In 1994, Chen and Gill built a Fuzzy Neural Controller (FNC) and a neural emulator to provide the controller with the information needed fro learning. They used the controller to control an inverted pendulum as an unstable system [10]. Furthermore, A neural-fuzzy controller was designed in 1997 by Bose et al. to control an induction motor. They proposed this controller to a stator flux-oriented electric vehicle

induction motor via a back-propagation learning algorithm [8]. In 1997, Cerruto et al. used the neural-fuzzy algorithm with the Model Reference Adaptive Control (MRAC) to control an indirect field oriented induction motor [3]. Moreover, Yu et al. introduced in 1998 a FNN that tunes a PID controller [13]. Samhouri et al. addressed in 2005 an adaptive neuro-fuzzy approach to PID tuning. They used the approach to online tune a PID gains. They tested the controller with a pneumatic gantry robot [13].

2- Fuzzy Neural Network Design

In order to design a FNC a NN should be designed first. This NN represents the structure of FLC. Its called Fuzzy Neural Network (FNN) Structure .In this work, it assumed that a Mamdani type FLC with two inputs of error and rate of error and one output is used. The memberships used for the inputs and output are bill shaped type with 7 memberships for each. These rules are reduced from 7*7 to 7 only since any input has some contribution in all of the fuzzy sets and will circle around the main diagonal of the fuzzy rule table and settle in the center of this table (4). Hence, fuzzy rule table will be as shown in table (1)

where the abbreviations of the table represent: PB as Positive Big, PM as a Positive Medium, PS as Positive Small, Z as Zero, NS as Negative Small, NM as Negative Medium and NB as Negative Big. Moreover, the rules are implied using product for AND operation. Furthermore, the defuzzification used in this controller is a center of gravity type.

2-1 FNN Structure

The structure of FNN is shown in Fig. (1). This structure consists of five layers which are:

a- Input layer: in the input layer, each node transmits the corresponding input to the antecedent layer, thus:-

$$X_i^1 = I_i \quad (1)$$

and

$$O_i^1 = X_i^1 \quad (2)$$

Where
i_i is the i^{th} network input,
X_i^1 is the node input, and
O_i^1 is the node output.
The subscript refers to the node number while the superscript refers to the layer number.

b- Antecedent layer: this layer will transmits each value of input to the corresponding linguistic set, hence:-

$$X_i^2 = -\frac{1}{2}.\left(\frac{O_j^1 - m_{ij}}{s_{ij}}\right)^2 \quad (3)$$

and

$$O_i^2 = e^{\left(X_i^2\right)} \quad (4)$$

Where i =1,2…14, j= 1,2,..
mij is the center of bill shaped fuzzy membership function,
sij is the standard deviation,
i refers to the antecedent node while j refers to the input node.

c- Rule layer: this layer performs the implication of AND operation. The rule is implied using product operation. Since only 7 rules will contribute the rules which represent the diagonal of the fuzzy rule table mentioned in table (1), then:

$$X_i^3 = O_i^2.O_{i+7}^2 \quad (5)$$

and

$$O_i^3 = X_i^3 \quad (6)$$

Where i = 1,2….7.

d- Consequent layer: only two nodes are available in this layer which performs the center of gravity defuzzification algorithm. The first node has a weighted input and the

second is of the strength of unity, thus:

$$X_1^4 = \sum_{i=1}^{n} O_i^3 . y_i \quad (7)$$

$$X_2^4 = \sum_{i=1}^{n} O_i^3 \quad (8)$$

and

$$O_i^4 = X_i^4 \quad (9)$$

Where i =1,2. n is the number of rules
yi is the weight between the rule and the consequent layers.

e- Action layer: the completion of center of gravity defuzzificztion algorithm is done in this layer, so the input and output of each node is given by:

$$X^5 = \frac{O_1^4}{O_2^4} \quad (10)$$

and

$$O^5 = X^5 \quad (11)$$

2-2 FNN Learning Algorithm

FNN structure is expressed analytically in the previous section, such that the optimization method (Steepest Descent) can be applied on such structure. Hence, it can be implemented in the layers as follows:

a. Consequent layer (layer four):

$$y_i(k+1) = y_i(k) + \frac{\eta . O_i^3 . (O_d(k) - O^5(k))}{O_2^4} \quad (12)$$

Where η is the learning rate,
O_d is the desired output

b. Antecedent layer (layer two):In this layer, the learning equation of back propagation is:

$$\mathbf{m_{ij} = (k+1) = m(k) + \eta . O_i^3 . (O_d(k) - O^5(k)) ... \cdot (y_i(k) - O^5(k)) . \frac{(O_j^1 - m_{ij})}{O_2^4 . (s_{ij})^2}} \quad (13)$$

and

$$\mathbf{s_{ij}(k+1) = s_{ij}(k) + \eta . (O_d(k) - O^5(k)) ... \cdot (y_i(k) - O^5(K)) . \frac{(O_j^1 - m_{ij})^2}{O_4^2 . (s_{ij})^3}} \quad (14)$$

Equations (12), (13), and (14) will perform the back propagation procedure. Initializing the parameters in FNN is very important since it may reduce the learning time of the network, thus these parameters have been chosen at the same bases of choosing them in an ordinary fuzzy logic controller. That is m_{ij} will divide the universe of discourse to 7 equal intervals, while s_{ij} will give the bill shaped functions a reasonable width. Finally, y_i is scattered along the output universe of discourse in an equal intervals.

3-PID Like Fuzzy Neural Controller (PID-FNC)

The equation of PID controller in time domain is:

$$u(t) = K_P . e(t) + K_D . \delta e(t) + K_I . \int e(t) . dt \quad (15)$$

where K_P, K_I and K_D are the proportional, integral and derivative gains of the PID respectively.

Thus, in the discrete case of a PID like fuzzy controller one has an additional process state variable, namely sum-of-error to dente the integral part. Unfortunately, if any input is described with (m) linguistic value, then since PID controller has three inputs and since any rule has three conditions, then there is a need of $m*m*m=m^3$ rules. So, it is too much work to write m^3 rules. The PID-like fuzzy controller can be constructed as a parallel structure of a PD-like fuzzy controller and a PI-like fuzzy controller and the output can be approximated as [11]:

$$\mathbf{u(t) = u_a + u_b = (K_{P1} . e + K_D . \delta e(t)) + ... \cdot (K_{P2} . e + K_I . \int e . dt)} \quad (16)$$

However, the first part of equ. (16) represents PD controller. PD controller for any pair of the values of e and δe, calculates the control signal (u_a).

$$u_a(t) = K_{P1}.e(t) + K_D.\delta e(t) \qquad (17)$$

The fuzzy controller should do the same thing. For any pair of error (e) and change of error (δe), it should work out the control signal though rules. In fuzzy rules, the sampling time will be omitted since such a rule expresses a causal relationship between the process state and control output variables, which holds for any sampling time.

Moreover, the second part of equ. (16) represents PI controller, which can represent PI like fuzzy controller. The fuzzy controller and the rules table have other inputs; error and sum of error. It means that, the rules themselves should be reformulated. Since only the diagonal of the rule table will be used, it is found that the rules of PI-controller part are the same rules mentioned in table (1). However, the equation of PI controller is:

$$u_b(t) = K_{P2}.e(t) + K_I.\int e(t).dt \qquad (18)$$

The proposed PID-FNC will consist of two FNNs. The first one is for PD like controller action mentioned in equ. (17), while the second FNN is for PI like controller action mentioned in equ. (18). The general block diagram of PID-FNC is shown in Fig. (2). The parameters of K_{ua} and K_{ub} are the output scaling factors of PD like and PI like fuzzy controllers respectively. It will be assumed in this work that there is no need to any rule definition, since the rule layer is fixed and take the optimal rules of the fuzzy logic controller. However, the general Block Diagram of the Adaptive PID-FNC Controlled System is shown in Fig. (3). In this figure, Neural Network Emulator (NNE) is used to emulate the plant model. Hence, it is used to generate the required propagated error signal to PID-FNC (FNCe). This NNE uses the back propagation algorithm as a learning technique and uses the error between the plant and NNE, (NNEe), as the learning signal to adjust its weights. PID-FNC uses the error between the plant output and the reference model output to generate both error, sum of error and change of error internally. Then after generating the controller action, it updates its weights using FNCe; which is the error between the reference model output and the plant output propagated through NNE.

4-Simulation Results

In order to simulate the closed loop system, a nonlinear model with two inputs / two outputs is selected to be controlled with the following system of difference equations:

$$x_1(k+1) = \frac{x_1(k)}{1+x_2^2(k)} + u_1(k) \qquad (19)$$

and

$$x_2(k+1) = \frac{x_1(k).x_2(k)}{1+x_2^2(k)} + u_2(k) \qquad (20)$$

while the reference model is chosen to be a 2nd order linear type with the following system of difference equations:

$$x_1(k+1) = 0.06.x_1(k) + 0.02.x_2(k) + r_1(k) \qquad (21)$$

and

$$x_2(k+1) = 0.01.x_1(k) + 0.08.x_2(k) + r_2(k) \qquad (22)$$

An input consists of a step response, sine wave and cosine wave with amplitude of .5 were applied to the inputs of the closed loop system at intervals of 0, 50 and 150 seconds respectively. However, the response of reference model outputs for the specified inputs is shown in Fig. (4).

The NNE that represents the model of the plant is trained first. A feed-forward parallel-series neural

network with 12 hidden layers will be used. The inputs to the NNE consist of 8 values, they are: $u_1(k)$, $u_2(k)$, $u_1(k-1)$, $u_2(k-1)$, $x_1(k-1)$, $x_2(k-1)$, $x_1(k-2)$ and $x_2(k-2)$ and the outputs of the NNE are $\hat{X}_1(k+1)$ and $\hat{X}_2(k+1)$. Furthermore, the activation functions of the hidden layer are selected to be hyperbolic tangent sigmoid type, while the activation functions of the output layer are of linear type. By training the NNE using Gradient Descent method with learning rate of 0.01, momentum term of 0.01 and minimum accepted training error of 10^{-6}, the NNE will be trained after 110 epochs.

Two controllers were added for each input of the model. Each controller consists of the PID-FNC structure shown in Fig. (2). The learning rate of each controller is 0.0001 with a momentum term of 0.01. It is assumed the load and disturbance will be added within the simulation time interval of (67 to 134 seconds).

As a first step, the controller was tuned to follow a reference model with transient specifications of minimum overshoot, minimum rise time and minimum steady state error. Using trial and error, it was found that the best controller gains with the output scaling factors are shown in table (2).

By applying the specified input signal mentioned above to the two reference inputs, $R_1(k)$ and $R_2(k)$, of the closed loop system with no load and no effect of disturbances, the responses of the outputs of the model, $x_1(k)$ and $x_2(k)$, are shown in Fig. (5-a) and Fig. (5-b) respectively. The two controllers will try to compensate any change in the reference input to achieve the transient response specifications. The responses of the two controller outputs are shown in Fig. (5-c). Moreover, to test the closed loop system with the effect of load, a constant load was applied at the outputs of the model. Its value was chosen to be 50% of the maximum amplitude of the reference inputs. However, the responses of the outputs of the model with the effect of load are shown in Fig. (6-a) and Fig. (6-b) respectively. Also, the responses of the two controllers are shown in Fig. (6-c). Furthermore, to test the closed loop system with the effect of disturbances, a random noise of uniform distribution type with amplitude of 10% from the maximum reference inputs was applied at the inputs of the model. The responses of the outputs of the model with the effect of disturbances are shown in Fig. (7-a) and Fig. (7-b) respectively. Also, the responses of the two controllers are shown in Fig. (7-c). Finally, with the addition of the same constant load and external disturbances to the closed loop system, the responses of the outputs of the model are shown in Fig. (8-a) and Fig. (8-b) respectively. Also, the responses of the two controllers are shown in Fig. (8-c).

5-Conclusions and Future Work

It can be seen from results that the controller was able to meet the design goals, minimum overshoot, minimum rise time and minimum steady state error. The controller was built using simple fuzzy-neural algorithm with reduced number of fuzzy rules compared with ordinary FLC. This simple structure can reduce the calculation time of the control action, hence improving the reliability of system and make it possible to be used in real time applications. The PID technique improves the behavior of the system and the learning ability improves the robustness of the controller. However, the problem of implementing three dimensional fuzzy rule base table to produce the PID like FNC was solved in a simple way by

separating the PID-FNC controller into PD like FNC and PI like FNC. The outputs of the controller were added to produce the control action. However, the closed loop response acts smoothly with a robust behavior even with the existence of load effect and disturbance. The results showed that the closed loop time responses behaves with accepted specifications as an overshoot less than 4%, rise time less than 0.2 seconds and steady state error less than 0.005.

However, the need to train the NNE off-line before using the closed loop controller is a major drawback in this design, since a lot of time will be spent to train the NN.

As a future work, the PID-FNC can be used to control a time varying models by training the NNE on-line.

References

[1] Bose, B.K. Patel, N.R. and Rajashekara, K., April, 1997 "A Neuro-Fuzzy Based online Efficiency Optimization Control of a Stator Flux-Oriented Direct Vector-Controlled Induction Motor Drive", IEEE Trans. On Industrial Electronics, Vol. 44, No. 2, PP. 270-273.

[2] Brown, M. and Harris, C., 1994, "Neuro-Fuzzy Adaptive Modeling and Control", Prentice-Hall Inc., Englewood Cliffs, N.J. USA,

[3] Cerruto, E., Consoli, A., Raciti, A. and Testa, A., Nov. 1997, "Fuzzy Adaptive Vector Control of Induction Motor Drive", IEEE Trans. On Power Electronics, Vol. 12, No. 6, PP. 1028-1040.

[4] Chen, Y.M. and Gill, K.F., April 1994, "Applications of Fuzzy Neural Network to the Control of an Unstable System", IMACS, International Symposium on Signal Processing, Robotics and Neural Networks, France, PP. 454-457,.

[5] Gerry, J.P., March 1987, "A Comparison of PID Control Algorithms", Control Engineering Magazine.

[6] Golob, M., 2001,"Decomposed Fuzzy Proportional-Integral Derivative Controllers", Applied Soft Computing, Vol. 18, PP. 1-14.

[7] Hayashi, H., Nasu, J., Strafezza, M. and Dote, Y., 1991,"Neuro-Fuzzy Transmission Control for Automobile", Intelligent Engineering Systems Through Artificial Neural Network, ASME Press, USA, PP. 283-288.

[8] Kaya, A, and Schelb, T.J., July 1988, "Tuning PID Control of Different Structures", Control Engineering Magazine, Vol. 35, No. 7.

[9] Kosko, B., 1992, "Neural Networks and Fuzzy Systems", Prentice Hall, Englewood Cliffs, USA.

[10] Pham, D. T. and Xing, L., 1995, "Neural Networks for Identification, Prediction and Control", Springer-Verlag Ltd., Great Britain.

[11] Reznik, L., 1997, "Fuzzy Controllers", Biddles Ltd., Britain.

[12] Samhouri, M., Raoufi, A.and Surgenor, B., Aug. 2005,"Control of a Pneumatic Robot for Grinding: A Neuro-Fuzzy Approach to PID Tuning", Proceedings 2005 of IEEE Conference on Control Applications (CCA), Toronto, Ontario, 28-31.

[13] Yu, K. W., Hwang, R. C and Hsien, J. G., 1998, " Fuzzy PID Controller Gain Scheduling by using Neural Network Back-Propagation Algorithm", International Conference on Brain

and Neural Network.

[14] Zang, Y.Q., and Kandel, A., Jan. 1998, "Compensatory Neuro-Fuzzy Systems with Fast Learning Algorithm", IEEE Trans. On Neural Networks, Vol. 9, No.1, , PP. 83-105,

Table (1) Rules of Fuzzy Logic Controller.

Change Of Error (ce) \ Error (e)	NB	NM	NS	Z	PS	PM	PB
NB	PB						
NM		PM					
NS			PS				
Z				Z			
PS					NS		
PM						NM	
PB							NB

Table (2) Best parameter values of the two PID-FNC

Controller of $u_1(k)$					
K_{P1}	K_{P2}	K_1	K_D	K_{ua}	K_{ub}
0.5	2.5	2	1	0.25	2
Controller of $u_2(k)$					
K_{P1}	K_{P2}	K_1	K_D	K_{ua}	K_{ub}
1.5	2	1.5	1.5	0.25	1.25

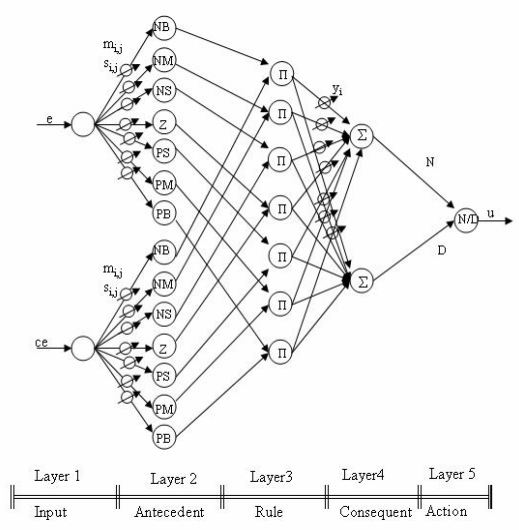


Fig. (1) FNN Structure.

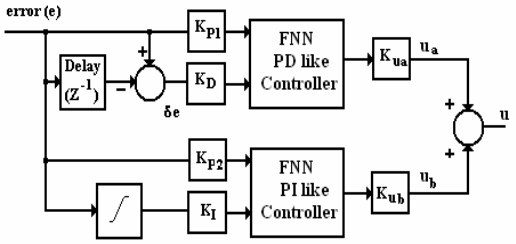


Fig. (2) General block diagram of PID-FNC.

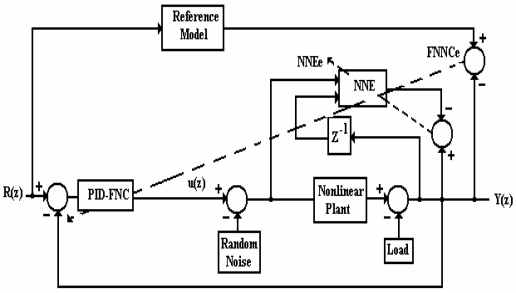


Fig. (3) General block diagram of the adaptive PID-FNC controlled system.

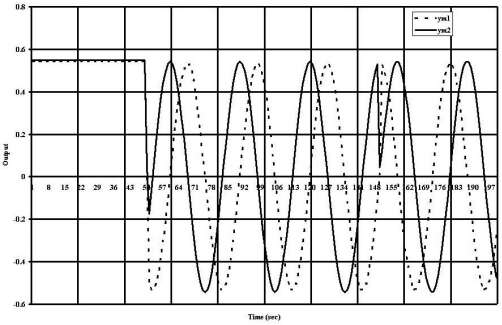
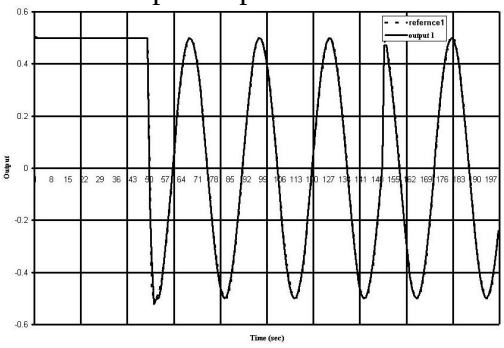
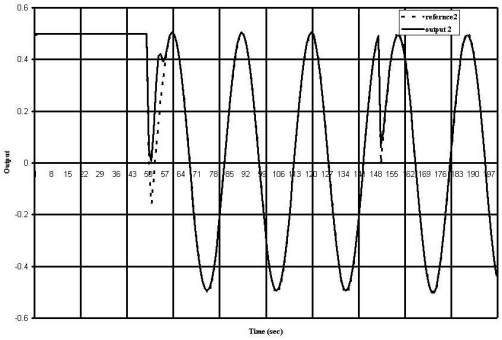


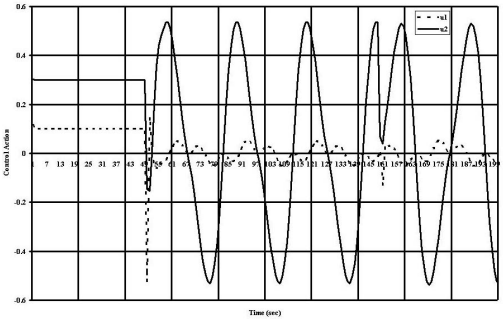
Fig. (4) The reference model input and output responses.



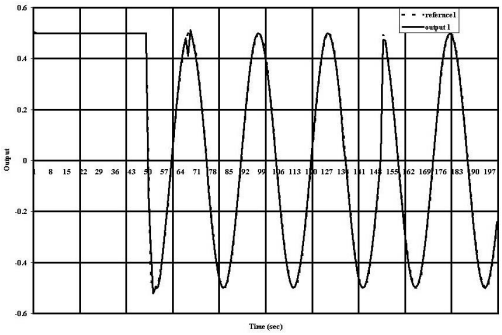
Fig(5-a) Closed loop response of x_1 and reference input R_1 with no load and no disturbances.



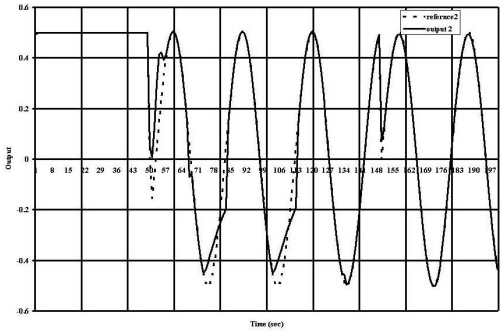
Fig(5-b) Closed loop response of $x_2(k)$ And reference input $R_2(k)$ with no load and no disturbances.



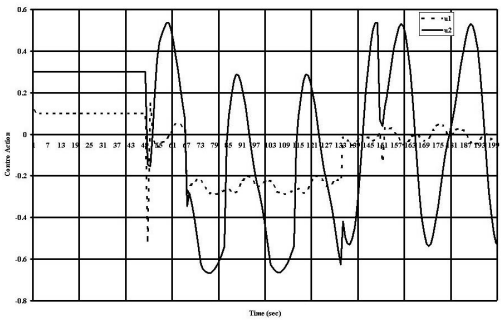
Fig(5-c) Time response of controllers outputs, (U_a and U_b), with no load and no disturbances.



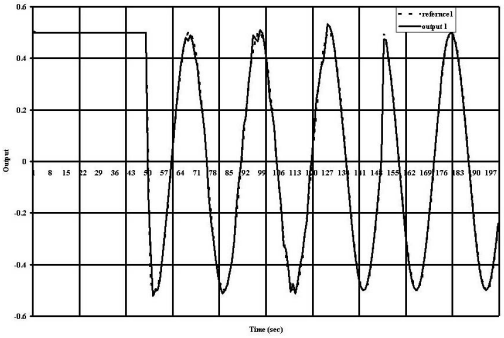
Fig(6-a) Closed loop response of x_1 and reference input R_1 with 50 % load and no disturbances.



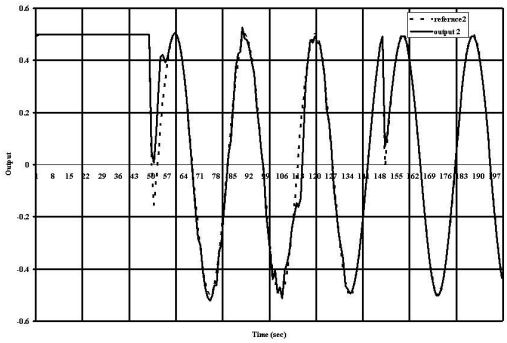
Fig(6-b) Closed loop response of $x_2(k)$ And reference input $R_2(k)$ with 50% load and no disturbances.



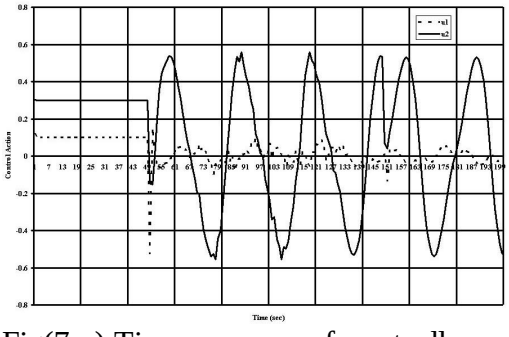
Fig(6-c) Time response of controllers outputs, (U_a and U_b), with 50% load and no disturbances.



Fig(7-a) Closed loop response of x_1 and reference input R_1 with no load and 10% disturbances.

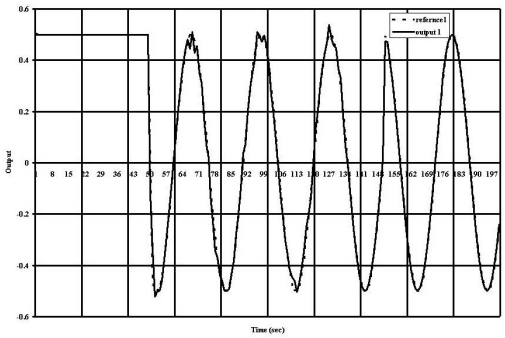


Fig(7-b) Closed loop response of $x_2(k)$ and reference input $R_2(k)$ with no load and 10% disturbances.

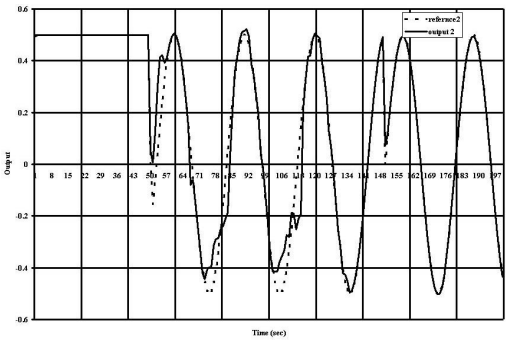


Fig(7-c) Time response of controllers outputs, (U_a and U_b), with no

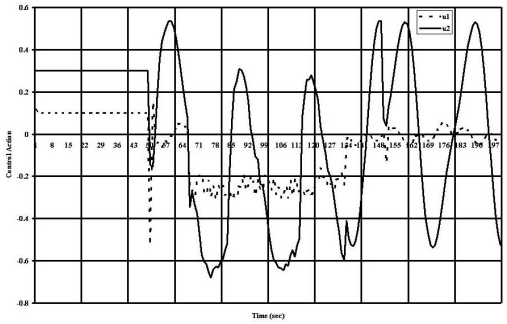
load and 10% disturbances.



Fig(8-a) Closed loop response of x_1 and reference input R_1 with 50 % load and 10% disturbances.



Fig(8-b) Closed loop response of $x_2(k)$ and reference input $R_2(k)$ with 50% load and 10% disturbances.



Fig(8-c) Time response of controllers outputs, (U_a and U_b), with 50% load and 10% disturbances.