國立臺灣師範大學工業教育研究所

碩士論文

指導教授：呂藝光 博士

遺傳演算模糊類神經與其在直流伺服馬達上之應用

Genetic Algorithms Fuzzy-Neural Controller and Its Application in DC Servo Motors

研究生：林建佑 撰

中 華 民 國 九十八年七月

國立台灣師範大學工業教育學系　　研究生：林建佑

碩士論文：遺傳演算模糊類神經與其在直流伺服馬達上之應用

經審查合格，特予證明

論文口試委員

鄭錦聰　博士
國立虎尾科技大學資訊工程學系主任

莊鎮嘉　博士
國立宜蘭大學電機工程學系副教授

洪欽銘　博士
國立台灣師範大學應用電子科技學系主任

曾煥雯　博士
國立台灣師範大學應用電子科技學系副教授

呂藝光　博士
國立台灣師範大學應用電子科技學系副教授
本論文指導教授

系主任：

中 華 民 國　98　年 7　月　15　日

Genetic Algorithms Fuzzy-Neural Controller and Its Application in DC Servo Motors

Student：Jian-You Lin　　Adviser：Dr. Yih-Guang Leu

Department of Industrial Education
National Taiwan Normal University

ABSTRACT

In this thesis, a compact genetic algorithm used to tune the parameters of fuzzy neural networks is proposed for function approximation and adaptive control of nonlinear systems. For the design of adaptive controller, the compact genetic algorithm does not require the procedure of off-line learning and the complicated mathematical computation. Compared with traditional adaptive controllers, computation loading can be effectively alleviated. In adaptive control procedure for nonlinear systems, the weights of the fuzzy neural controller are online adjusted by the compact genetic algorithm in order to generate appropriate control input. For the purpose of on-line evaluating the stability of the closed-loop systems, an energy fitness function derived from Lyapunov function is involved in the compact genetic algorithm. In addition, the system states may go into the unsafe region if the compact genetic algorithm can not instantaneously generate the appropriate weights. In order to guarantee the stability of the closed-loop nonlinear system, a safe controller is incorporated into the fuzzy neural controller.

Finally, some computer simulation examples and a servo motor experiment are provided to demonstrate the feasibility and effectiveness of the proposed method.

Keywords: genetic algorithm, fuzzy neural networks, adaptive control, nonlinear systems

遺傳演算模糊類神經與其在直流伺服馬達上之應用

學生:林建佑　　指導教授:呂藝光 博士

國立臺灣師範大學工業教育學系碩士班

摘　　要

本文提出一個使用小型的遺傳演算法來調整模糊類神經網路的參數，並將其應用於函數近似與非線性系統之適應控制器設計。此小型的遺傳演算法應用於適應控制器設計，不需要事先離線學習的程序和複雜的數學運算。相較於傳統非線性系統的適應控制器，可有效減少適應控制器所需複雜的數學運算。在非線性系統之適應控制過程中，模糊類神經控制器的權重值是經由遺傳演算法來即時調整，以產生適當的控制輸入。為了即時評估閉迴路系統穩定的趨勢，本文從里亞布諾夫（Lyapunov）函數的穩定性分析推導過程中，提出一個能量適應函數於小型的遺傳最佳演算法中，藉此獲得較佳的閉迴路系統的穩定度。此外，由於小型的遺傳演算法可能在即時控制過程中使系統狀態進入不安全的區域。因此，加入安全控制器以限制閉迴路系統的狀態進入不安全的區域。

本文藉由電腦模擬結果驗證所提出方法的可行性與效能。最後，將此模糊類神經控制器應用在直流伺服馬達追蹤控制實驗。

關鍵字: 遺傳演算法、模糊類神經、適應控制、非線性控制

誌　　謝

由衷地感謝指導教授呂藝光博士在研究所兩年悉心的指導與教誨，且無私奉獻的訓練筆者在學術知識、語言表達與獨立思考之能力，使本論文得以順利完成，謹此致上最誠摯的謝意。

其次感謝口試委員臺灣師範大學洪欽銘主任、曾煥雯教授、虎尾科技大學鄭錦聰主任及宜蘭大學莊鎮嘉教授對論文的指正與建議，使本論文的內容更臻完善。再者感謝實驗室學長宏見和同學建宏、建豪、俊堯、伯楷、正皓、名峰以及學弟皓勇、皓程、嘉良、弼廣、小建宏、暉翔在研究上的指導切磋和精神上的相互鼓勵，讓我每天的生活充滿動力與歡樂。

最後，將此論文獻給敬愛的雙親、家人及女友，深深感謝您們在筆者的成長過程中，給予關心體恤與支持鼓勵，使筆者在求學過程中可以專心完成學業而無後顧之憂，願與您們分享此份榮耀。

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

1.1 Background

In the past decade, the design of fuzzy logic systems and/or neural networks for adaptive controllers [1-6] has been widely developed because of the universal approximation feature [7-8], and the stability analysis of the adaptive fuzzy logic and/or neural network controllers for nonlinear systems is generally provided by Lyapunov stability theory. In addition, to absolutely guarantee the stability of the closed-loop systems, a supervisory controller for fuzzy systems has been proposed in [4]. In general, the universal approximators are trained via gradient-based methods, which may only find a local minimum solution during the learning process. To search for global optimal solutions, many heuristic learning algorithms have been proposed, such as genetic algorithm (GA), simulated annealing (SA), particle swarm optimization (PSO), etc. Genetic algorithms utilize the concept of “survival of the fittest” of Darwin’s theory. GA is an exploratory procedure which can find an optimal solution for complex engineering problems, such as optimization problems and machine learning. Genetic algorithms have been incorporated into the design of fuzzy logic systems and/or neural networks systematically [9-29] because they are based on natural selection and natural genetics, and possess the simple implement ability and the capability of escaping from local optima. In [9], to guarantee the system stability for the nonlinear systems, the solution of the Lyapunov condition is solved by the genetic algorithms. An optimal fuzzy

controller design with index function obtained by genetic algorithms has been proposed in [10]. Vehicle system control involved in genetic algorithms has been proposed in [11-12]. Motor servo control via genetic algorithms has been proposed in [13-14]. In [25-28], for the fuzzy-neural networks, the learning process utilizes genetic algorithms rather than the conventional learning methods. In [29], to reduce the computation loading, a reduced-form genetic algorithm has been proposed for function approximation.

Some adaptive fuzzy-neural controllers for nonlinear systems have been proposed to incorporate the expert information systematically, and their stability can be guaranteed by the universal approximation theorem [30]. In general, the adaptive controller can be divided into two different approaches [31]. One is indirect adaptive control, and the other is direct adaptive control. In the direct adaptive control [4], [6], the controlled system model is parameterized in terms of the controller parameters estimated directly without computing the controlled system parameters. In the indirect adaptive control, [32-34], the controlled system parameters are estimated and used to compute the controller parameters. In [3], an indirect adaptive fuzzy-neural controller for nonlinear systems with H_∞ tracking performance has been developed. On the other hand, a direct adaptive fuzzy-neural control with supervisory controller for nonlinear systems with only output measurement has been proposed in [6]. Since the 1990s, backstepping technique has become one of the most popular control methods for a large class of nonlinear systems [35-38]. Compared with feedback linearization methods, the backstepping technique has the advantage of avoiding the cancellation of useful nonlinearities in the design process. Thus, in the past decade, the backstepping technique has been widely

used for nonlinear control systems. Its main design procedure is that an appropriate state and virtual control are selected for each smaller subsystem, then the state equation is rewritten in terms of them, and finally Lyapunov functions are chosen for these subsystems so that the true controller integrating the individual controls of these subsystems guarantees the stability of the overall system. Recently, many intelligent backstepping methods [39-41] have been proposed to control nonlinear systems with unknown system dynamics by combining the intelligent control methods with an adaptive backstepping design. Thus, in this thesis, the adaptive control and backstepping techniques are used.

The motion control design of robot manipulators has attracted considerable attention by many researchers [42-45]. The system dynamics of the robotic manipulators is often unknown and time-varying, and they also have to suffer from various uncertainties in their dynamics, such as nonlinear friction, payload variation, and external disturbance. Therefore, in order to control the robot manipulator, it is necessary that a control algorithm has robustness. Traditionally, the traditional proportional and derivative (PD) controller is very simple and does not require any knowledge of the robot dynamics. It, however, requires very large actuation to achieve precise control. This results in a difficulty in implementation [42]. Recently, some adaptive controllers integrated with intelligent control methods, such as fuzzy systems and neural networks, have been designed for robot manipulators [43-45].

1.2 Motivation and Purpose

Since the complicated mathematical form for fuzzy logic systems and/or

neural networks [1-6], such as the update laws and the Lyapunov condition for the system stability, must be solved. For this reason, it is difficult to implement the control algorithms into real controllers. Moreover, the design of adaptive controllers incorporated into genetic algorithms generally requires the procedure of off-line learning [10-14] before they on-line control a plant. Thus, in this thesis, to avoid solving complicated mathematical equations, a genetic algorithm controller without the procedure of off-line learning is developed for nonlinear systems, and the stability of the closed-loop system is guaranteed.

More specifically, the objective of this thesis is to develop the control scheme incorporated a compact genetic algorithms and fuzzy-neural networks into backstepping design. The weighting factors of the adaptive fuzzy-neural controller are tuned on-line via the compact genetic algorithm, instead of solving complicated mathematical equations. For the purpose of on-line tuning these parameters and evaluating the stability of the closed-loop system, an energy fitness function is included in the compact genetic algorithms. According to the energy fitness function, the adaptive backstepping fuzzy-neural controller can instantaneously evaluate the stability of the closed-loop system and generate the appropriate control action. In addition, in order to guarantee that the system states are confined to the safe region, the concept of the safe control is incorporated into the adaptive backstepping fuzzy-neural controller.

1.3 Organization of the Thesis

The remaining chapters of this thesis are organized as follows. Chapter 2 describes the fuzzy-neural networks and evolutionary processes of the compact

genetic algorithms. A novel approach to adjust the weightings of fuzzy-neural networks using a compact genetic algorithms optimization algorithm is introduced in this chapter. In Chapter 3, a genetic adaptive backstepping fuzzy-neural controller for uncertain single-input single-output (SISO) nonlinear systems is proposed by using a compact genetic algorithm. To illustrate the feasibility and effectiveness of the proposed method, two examples of nonlinear systems controlled by the proposed method are demonstrated. The purpose of Chapter 4 is to design an adaptive backstepping fuzzy-neural controller using the compact genetic algorithm for a class of multiple-input multiple-output (MIMO) nonlinear systems. An illustrative example is in this chapter to show the feasibility and applicability of the proposed method. Then in Chapter 5, an adaptive backstepping fuzzy-neural controller using the compact genetic algorithm for *n*-link planar manipulator nonlinear systems is proposed. The applicability of the proposed control scheme is demonstrated by the simulation results. Chapter 6, the compact genetic algorithm fuzzy-neural controller for a DC servomotor is performed through experiment. The DC servomotor is introduced in this chapter, and the computer simulations and experimental results are shown. Finally, Chapter 7 summaries this thesis and given some suggestions for future research.

Chapter 2

The Learning of Fuzzy-neural Networks Using Compact Genetic Algorithms

2.1 Overview

A novel method of adjusting the weightings of fuzzy-neural networks (FNN) using a compact genetic algorithm (CGA) optimization algorithm is introduced in this chapter. This approach can be used to search for the optimal parameters of fuzzy-neural networks, because of their capabilities of random search for global optimization. Compared with traditional genetic algorithms, the compact genetic algorithm has three simplified processes, (1) keep a small population size, (2) replace the string codes with real-value representation, and (3) perform compact mutation and crossover on the chromosomes by the energy fitness function. In addition, the structure of the fuzzy-neural networks is presented in this chapter. The organization of this chapter is summarized as follows. The construction of the fuzzy-neural networks is given in section 2.2. Section 2.3 briefly is reviewed the genetic algorithms. Section 2.4 gives the details of the proposed compact genetic algorithms. Conclusions are drawn in Section 2.5.

2.2 Fuzzy-Neural Networks

Fuzzy-neural network is generally a fuzzy inference system constructed from structure of neural networks. Learning algorithm is used to adjust the weightings of the fuzzy inference system [8]. Fig. 2-1 shows the configuration

of a typical fuzzy-neural network. It has the total of four layers. Nodes at layer I are input nodes that represent input linguistic variables. Nodes at layer II represent the values of the membership function of total linguistic variables. At layer III, nodes are the values of the fuzzy basis vector ξ. Each node at layer III is a fuzzy rule. The links between layer III and layer IV are full connected by the weighting factors. Layer IV is the output layer.

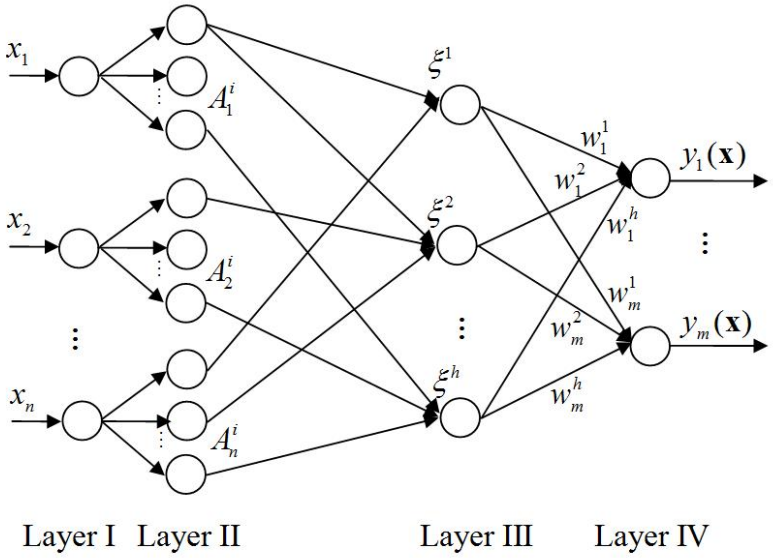

Fig. 2-1. Configuration of the fuzzy-neural network.

2.2.1 Fuzzy Inference Method

The fuzzy inference engine uses fuzzy IF-THEN rules to perform a mapping from training input data x_q, $q = 1,2,\cdots,n$, and the output data y_p, $p = 1,2,\ldots,m$. The *i*th fuzzy rule has the following form:

$$\begin{aligned} R^i &: \text{IF } x_1 \text{ is } A_1^i \text{ and } \cdots\ x_n \text{ is } A_n^i \\ &\quad \text{THEN } y_1 \text{ is } w_1^i \text{ and } \cdots\ y_m \text{ is } w_m^i \end{aligned} \tag{2-1}$$

where i is a rule number, and A_q^i is the fuzzy sets of the antecedent part, and

w_p^i is real numbers of the consequent part. When the input vector $\mathbf{x} = [x_1, x_2, ...x_n]^T$ is given, By using product inference, center-averaging and singleton fuzzification, the output of the fuzzy-neural network y_p can be expressed as:

$$y_p(\mathbf{x} \mid w_p) = \frac{\sum_{i=1}^{h} w_p^i \left(\prod_{q=1}^{n} \mu_{A_q^i}(x_q) \right)}{\sum_{i=1}^{h} \left(\prod_{q=1}^{n} \mu_{A_q^i}(x_q) \right)} = w_p^T \xi \tag{2-2}$$

where $\mu_{A_q^i}(x_q)$ is the membership function of A_q^i, h is the total number of IF-THEN rules, w_p^i is the point at which $\mu_{B^i}(w_p^i) = 1$, $w_p = [w_p^1 \; w_p^2 \cdots w_p^h]^T$ is a weighting vector related to the *p*th output $y_p(\mathbf{x})$, and $\xi^T = [\xi^1 \; \xi^2 \cdots \xi^h]$ is a set of fuzzy basis vector. ξ^i is defined as

$$\xi^i(\mathbf{x}) = \frac{\prod_{q=1}^{n} \mu_{A_q^i}(x_q)}{\sum_{i=1}^{h} \left(\prod_{q=1}^{n} \mu_{A_q^i}(x_q) \right)}, \quad i = 1, 2, \ldots, h \tag{2-3}$$

By adjusting weighting w_p^i values of the fuzzy-neural network, the learning algorithm can be derived to minimize the error function as follows:

$$e_p(w_p) = (y_p - y_p^*)^2, \quad p = 1, 2, \ldots, m \tag{2-4}$$

and for multioutputs

$$E(w) = \left\| Y - Y^* \right\|^2 \tag{2-5}$$

where $Y = [y_1 \; y_2 \cdots y_m]$ is an *m*-dimensional vector of the actual outputs of the fuzzy-neural network, $Y^* = [y_1^* \; y_2^* \cdots y_m^*]$ is an *m*-dimensional vector of the desired outputs, $w = [w_1^T \; w_2^T \cdots w_m^T]^T$ is a weighting vector of the fuzzy-neural network for *m* outputs.

2.3 Overview of Genetic Algorithms

Genetic algorithms (GAs), were first proposed by John Holland at the University of Michigan in 1975. GAs is optimization search algorithms that use principles inspired by natural population genetics to evolve solutions to problems. They are useful approaches to problems requiring effective and efficient searching, and their use is universal in applications to commerce, scientific, and engineering fields. GAs can be used to obtain an approximate solution for single variable or multivariable optimal problems. In GA, a set of variables for a given problem is encoded into a string (or other coding structure), analogous to a chromosome in nature. However, genetic algorithms are probabilistic algorithms, which maintain a population of individuals (often called chromosomes) for iteration. Each chromosome represents a potential solution to the problem, and is evaluated through some measure of its fitness. Then, more fit individuals selected form a new population. Some members of the new population undergo transformations by means of genetic operators to form new solutions. After some number of generations, it is hoped that the system converges with a near-optimal solution.

The objective of a GA is to mimic the nature evolution operation process contains reproduction, recombination (crossover) and mutation. Reproduction is the operator carrying old strings through into a new population, depending on the fitness value. Strings with high fitness values obtain a larger number of copies in the next generation. Crossover combines the trait of two parent chromosomes to form two similar offspring by swapping the corresponding segments of the parents. The instinct behind the applicability of the crossover operator is the exchange of information between potential solutions. Mutation,

arbitrarily alters one or more genes of a selected chromosome, by a random change with a probability equal to the mutation rate. The instinct behind the mutation operator is the injection of some extra variability into the population. The principal structure of genetic algorithms is shown in Fig. 2-2.

The GA described above is a conventional one. In this chapter, we propose a compact genetic algorithm, which is characterized by three simplified processes. The details will be discussed in the next section.

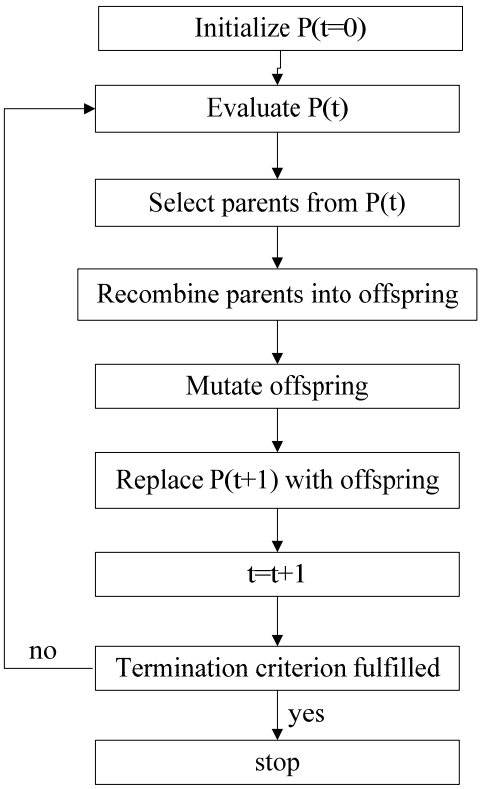

Fig. 2-2. A basic structure of genetic algorithms.

2.4 Evolutionary Processes of the Compact Genetic Algorithms (CGA)

To solve the problems encountered by conventional genetic algorithms, we propose a compact genetic algorithm to deal with the complicated status

where a huge number of adjustable parameters are to be searched in the fuzzy-neural networks. For the purpose of speeding up the computation of the genetic operation, the mechanism of the compact genetic algorithm has three simplified parts: (1) keep a small population size, (2) replace the string codes with real-value representation, and (3) perform compact mutation and crossover on the chromosomes by the energy fitness function. The details are discussed in the following.

2.4.1 Population Initialization

To evolutionarily obtain the adjustable parameters of the fuzzy-neural network shown in section 2.2, we define the chromosome as

$$^{\ell}\mathbf{w} = [\,^{\ell}w_1 \;\; ^{\ell}w_2 \cdots ^{\ell}w_{\beta}\,] \in R^{\beta} \tag{2-6}$$

where w is a set of weighting factors in the interval $D = [d, -d] \subseteq R$. Initially, a genetic algorithm requires a population of potential solutions to be initialized and then maintained. Here, a population consisting of *k* chromosomes is used to avoid the unlimited growth of population. Each chromosome consists of β genes, and each gene represents an adjustable parameter of the fuzzy-neural networks. Real number representation for potential solutions is also adopted to simplify the genetic operator definitions and obtain a better performance of the genetic algorithms. The initial chromosomes are randomly generated within the feasible ranges *D*. The initial population with *k* chromosomes defined in (2-6) is randomly generated as follows

$$\Xi = \begin{bmatrix} {}^{1}w \\ {}^{2}w \\ \vdots \\ {}^{k}w \end{bmatrix} = \begin{bmatrix} {}^{1}w_1 & {}^{1}w_2 & \cdots & {}^{1}w_\beta \\ {}^{2}w_1 & {}^{2}w_2 & \cdots & {}^{2}w_\beta \\ \vdots & \vdots & \ddots & \vdots \\ {}^{k}w_1 & {}^{k}w_2 & \cdots & {}^{k}w_\beta \end{bmatrix} \tag{2-7}$$

where ${}^{\ell}w = [{}^{\ell}w_1 \ {}^{\ell}w_2 \cdots {}^{\ell}w_\beta]$ is the ℓth chromosome for $\ell = 1, 2, \ldots, k$. Each chromosome has β elements. It is expected that one of the candidate solutions, ${}^{\ell}w$, can be evolutionarily obtained as a set of near-optimal parameters for the fuzzy-neural network. Note that the population is sorted by ranking the fitness of chromosomes, that is, the first chromosome denotes the best chromosome in the population in terms of fitness.

2.4.2 Energy Fitness function

The performance of each chromosome is evaluated according to the defined energy fitness function denoted as F. After generations of evolution, it is expected that the genetic algorithm converges, and a best chromosome with largest fitness representing the optimal solution to the problem is obtained. An energy fitness function for online tuning the weighting vector of the fuzzy-neural controller is established by the Lyapunov design approach. The detail explanation of the energy fitness function is given next chapter.

2.4.3 Single Gene Crossover Operation

In order to deal with a huge number of adjustable parameters, the single gene crossover is introduced in this section. Fig. 2-3(a) and (b) show the traditional crossover operation adopting one crossover point and two crossover points, respectively.

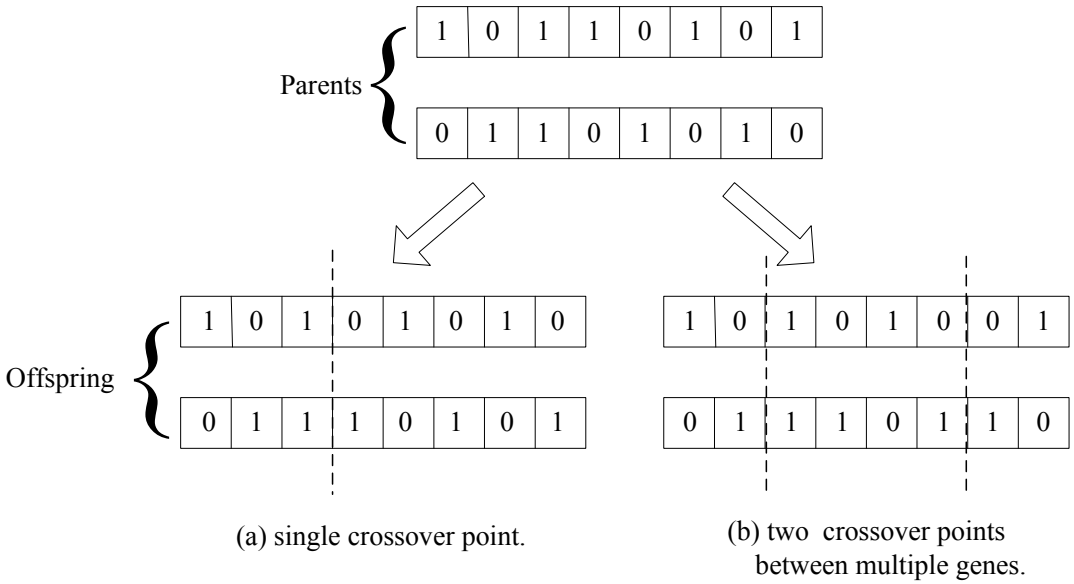

Fig. 2-3. Traditional crossover operation with encoded string.

Though the proposed single gene crossover shown in Fig. 2-4 has two crossover points, the distance between these points is only one gene (parameter). To avoid inappropriate crossover, the distance between the two crossover points is reduced to only one gene.

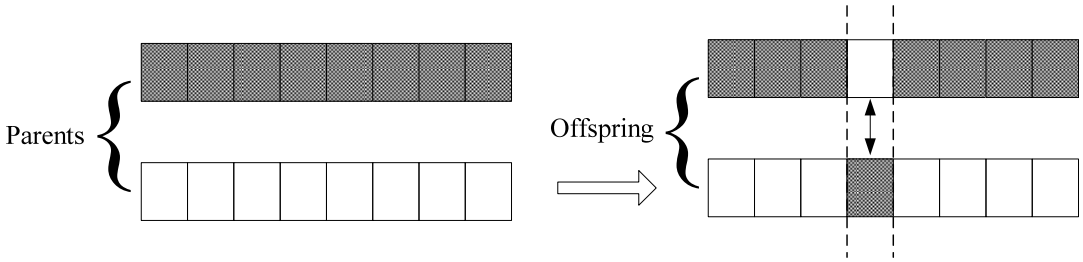

Fig. 2-4. The single gene crossover operation with real-value representation.

The single gene crossover operator is defined as

$$\hat{\Xi} = \begin{bmatrix} & {}^{1}\hat{w}_j & \\ \Lambda & {}^{2}\hat{w}_j & \Lambda \\ & \vdots & \\ & {}^{k}\hat{w}_j & \end{bmatrix} = \begin{bmatrix} {}^{1}\hat{w} \\ {}^{2}\hat{w} \\ \vdots \\ {}^{k}\hat{w} \end{bmatrix} \tag{2-8}$$

where j is the crossover point determined by randomly selected crossover point. Λ denotes the elements of offspring which remain the same as those of their parents, Then, the j-th gene of all chromosomes performs a linear combination operation and is updated as

$${}^{i}\hat{w}_j = \begin{cases} {}^{i}w_j * {}^{i}\zeta + {}^{i+(k/2)}w_j * (1 - {}^{i}\zeta) & \text{for } i = 1, 2, ..., k/2 \\ {}^{i}w_j * {}^{i}\kappa + {}^{i-(k/2)}w_j * (1 - {}^{i}\kappa) & \text{for } i = (k/2)+1, (k/2)+2, ..., k \end{cases} \tag{2-9}$$

where the crossover factors ${}^{i}\zeta$ and ${}^{i}\kappa$ are the linear combination weights and defined as

$${}^{i}\zeta = \frac{{}^{i}F - {}^{k}F}{[{}^{i}F - {}^{k}F] + [{}^{i+(k/2)}F - {}^{k}F]} \tag{2-10}$$

$${}^{i}\kappa = \frac{{}^{i}F - {}^{k}F}{[{}^{i}F - {}^{k}F] + [{}^{i-(k/2)}F - {}^{k}F]} \tag{2-11}$$

respectively. The single gene crossover operator generates new genes, ${}^{i}\hat{w}_j$, only at the position *j*-th for all chromosomes with a linear combination, and $\hat{\Xi}$ is a new population.

2.4.4 Sorting Operation

After crossover, the newly generated population is sorted by ranking the fitness F of chromosomes within the population, that is, $F({}^{1}\hat{w}) \geq F({}^{2}\hat{w}) \cdots \geq F({}^{k}\hat{w})$. The first chromosome ${}^{1}\hat{w}$ of the sorted population $\hat{\Xi} = [{}^{1}\hat{w}\ {}^{2}\hat{w} \ldots {}^{k}\hat{w}]^{T}$ has the highest fitness value.

2.4.5 Compact Mutation Operation

After sorting, the first chromosome is the best one in the population in terms of fitness. As to the operation procedure of the mutation operator, the (k/2+1)-th chromosome is replaced by the first chromosome according to the mutation rate p_m. Then, the (k/2+1)-th chromosome is updated as

$$ {}^{(k/2+1)}w_j = \begin{cases} {}^{1}w_j + \gamma^{\beta} * \eta, & \text{if } \delta > 0.5 \\ {}^{1}w_j \quad\quad\quad , & \text{if } \delta \le 0.5 \end{cases} \tag{2-12} $$

where

$$ \eta = \begin{cases} d - {}^{1}w_j \ , \text{if } \ \lambda > 0.5 \\ -d - {}^{1}w_j , \text{if } \ \lambda \le 0.5 \end{cases} \tag{2-13} $$

and $\delta, \gamma, \lambda \in [0,1]$ are random numbers, and $\beta > 0$ is a design parameter. The compact genetic flowchart is shown in Fig. 2-5.

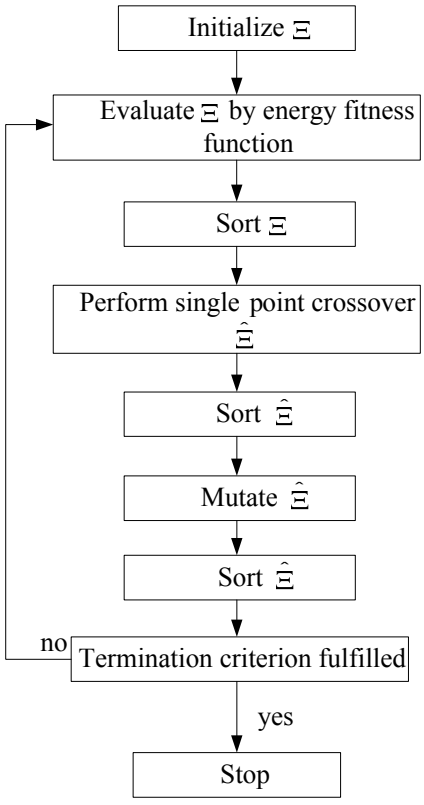

Fig. 2-5. The compact genetic flowchart.

2.5 Conclusions

A novel method of adjusting the weightings of fuzzy-neural networks using a compact genetic algorithm has been introduced in this chapter. This method can be used to search for the optimal parameters of fuzzy-neural networks. The compact genetic algorithm offers an advantageous position over the conventional gradient-based methods during the learning process of fuzzy-neural networks. To start this, chromosomes consisting of adjustable parameters of the fuzzy-neural networks are coded as a vector with real number components. The energy fitness function is obtained by the Lyapunov design approach. Thus, all the best adjustable parameters of the fuzzy-neural network can be obtained by repeating the compact genetic operations, including the single gene crossover and the compact mutation, so that an optimal fuzzy neural network satisfying a desired condition can be obtained.

Chapter 3

CGA-based On-Line Tuning of Fuzzy-neural Networks for Uncertain Nonlinear Systems

3.1 Overview

In this chapter, a genetic adaptive backstepping fuzzy-neural controller for uncertain nonlinear systems is proposed by using a compact genetic algorithm. The control scheme incorporates a compact genetic algorithm and fuzzy-neural networks into backstepping design. The weighting factors of the adaptive fuzzy-neural controller are tuned on-line via the compact genetic algorithm approach. Each gene represents an adjustable parameter of the fuzzy-neural networks with real number components. For the purpose of on-line tuning these parameters and evaluating the stability of the closed-loop system, an energy fitness function is included in the compact genetic algorithm approach. In addition, in order to guarantee that the system states are confined to the safe region, a safe control term is incorporated into the control scheme. To illustrate the feasibility and effectiveness of the proposed method, two examples of nonlinear systems controlled by the proposed method are demonstrated. The organization of this chapter is summarized as follows. In section 3.2, presents the problem formulation and the design of backstepping controller. Development of genetic adaptive fuzzy-neural control scheme is shown in section 3.3. Two examples are illustrated in section 3.4 to show the feasibility and effectiveness of the proposed method. Conclusions are drawn in section 3.5.

3.2 Backstepping Control Systems

In this section, we describe the control problem for a class of nonlinear systems, and then design the backstepping controller.

3.2.1 Problem Formulation

Consider the *n*th-order nonlinear systems as

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_3 \\ &\vdots \\ \dot{x}_n &= f(\mathbf{x}) + u \\ y &= x_1 \end{aligned} \tag{3-1}$$

where $u \in \Re$ and $y \in \Re$ are input and output of the system, respectively, and $\mathbf{x} = [x_1, x_2, ...x_n]^T \in \Re^n$ is the state vector of the system, which is assumed to be available for measurement. $f(\mathbf{x})$ is an unknown continuous function with known upper bound, *i.e.*$\left| f(\mathbf{x}) \right| \le f^U(\mathbf{x})$. Our control objective is to develop the backstepping controller so that the system output y follows a given bounded reference y_d.

3.2.2 The Design of Backstepping Controller

The detail design procedure of the backstepping controller is described as follows.

Step 1) Define a tracking error as

$$z_1 = x_1 - y_d \tag{3-2}$$

Then, differentiating z_1 can be expressed as

$$\dot{z}_1 = \dot{x}_1 - \dot{y}_d \tag{3-3}$$

Define a virtual control as

$$\alpha_1 = \dot{y}_d - c_1 z_1 \tag{3-4}$$

where $c_1 > 0$ is a design parameter. From (3-3) and (3-4), if $\alpha_1 = \dot{x}_1$, then $\lim_{t\to\infty} z_1 \to 0$, that is, the state trajectory x_1 can asymptotically track the bounded reference y_d. Thus, define an error state as $z_2 = \dot{x}_1 - \alpha_1 = x_2 - \alpha_1$. Then, our next goal is to force the error state z_2 to decay to zero. By using (3-4) and the fact that $\dot{x}_1 = z_2 + \alpha_1$, equation (3-3) can be rewritten as

$$\dot{z}_1 = z_2 - c_1 z_1 \tag{3-5}$$

Step 2) Differentiating z_2 can be expressed as

$$\dot{z}_2 = \dot{x}_2 - \dot{\alpha}_1 = x_3 - (-c_1 \dot{z}_1 + \ddot{y}_d) \tag{3-6}$$

Similarly, define a virtual control as

$$\alpha_2 = \ddot{y}_d - c_1 \dot{z}_1 - c_2 z_2 - z_1 \tag{3-7}$$

where $c_2 > 0$ is a design parameter. Moreover, define an error state as $z_3 = x_3 - \alpha_2$. Then, by using (3-7) and the fact that $\dot{x}_2 = z_3 + \alpha_2$, equation (3-6) can be rewritten as

$$\dot{z}_2 = z_3 - c_2 z_2 - z_1 \tag{3-8}$$

Step 3) Let *k* be a positive integer. Define an error state as $z_k = x_k - \alpha_{k-1}$. Then, differentiating z_k, where $3 \le k \le n-1$, can be expressed as

$$\dot{z}_k = \dot{x}_k - \dot{\alpha}_{k-1} \tag{3-9}$$

Define a virtual control as

$$\alpha_k = y_d^{(k)} - \sum_{i=1}^{k} c_i z_i^{(k-i)} - \sum_{j=1}^{k-1} z_j^{(k-1-j)} \tag{3-10}$$

where $c_i > 0$ is a design parameter. Moreover, define an error state as $z_{k+1} = x_{k+1} - \alpha_k$. Then, by using (3-10) and the fact that $\dot{x}_k = z_{k+1} + \alpha_k$, equation (3-9) can be rewritten as

$$\dot{z}_k = z_{k+1} - c_k z_k - z_{k-1} \tag{3-11}$$

Step 4) Differentiating z_n can be expressed as

$$\dot{z}_n = \dot{x}_n - \dot{\alpha}_{n-1} = f(\mathbf{x}) + u - (y_d^{(n)} - \sum_{i=1}^{n-1} c_i z_i^{(n-i)} - \sum_{j=1}^{n-2} z_j^{(n-1-j)}) \tag{3-12}$$

Define a control law as

$$u = y_d^{(n)} - \sum_{i=1}^{n} c_i z_i^{(n-i)} - \sum_{j=1}^{n-1} z_j^{(n-1-j)} - f(\mathbf{x}) \tag{3-13}$$

where $c_n > 0$ is a design parameter. Then, from (3-13), equation (3-12) can be rewritten as

$$\dot{z}_n = -c_n z_n - z_{n-1} \tag{3-14}$$

Step 5) Consider the Lyapunov function as follows

$$V = \frac{1}{2} \sum_{i=1}^{n} z_i^2 \tag{3-15}$$

By differentiating (3-15) and using (3-5), (3-8), (3-11) and (3-14), we have

$$\begin{aligned} \dot{V} &= \sum_{i=1}^{n} z_i \dot{z}_i \\ &= z_1(z_2 - c_1 z_1) + \sum_{i=2}^{n-1} z_i(z_{i+1} - c_i z_i - z_{i-1}) + z_n(-c_n z_n - z_{n-1}) \\ &= -\sum_{i=1}^{n} c_i z_i^2 \\ &\le -c_1 z_1^2 \end{aligned} \tag{3-16}$$

From (3-15) and (3-16), we can conclude that z_i is bounded. Moreover, from (3-5), $\dot{z}_1$ is also bounded. Integrating (3-16) can be expressed as

$$\int_0^\infty z_1(\tau)d\tau = -(V(\infty)-V(0))/c_1 \tag{3-17}$$

Because the fact that the right side of (3-17) is bounded, we have $z_1 \in L_2$. According to the Barbalat's Lemma [46], $\lim_{t\to\infty} z_1 = 0$, that is, the state trajectory x_1 can asymptotically track the bounded reference y_d.

On the basis of the aforementioned description, the backstepping controller of the *n*th order nonlinear system can be summarized as the following lemma.

Lemma 3.1: Consider the *n*th-order nonlinear systems (3-1). Let $z_1 = x_1 - y_d$ and $z_i = x_i - \alpha_{i-1}$ for $2 \le i \le n$, where $\alpha_1 = \dot{y}_d - c_1 z_1$, and $\alpha_k = y_d^{(k)} - \sum_{i=1}^{k} c_i z_i^{(k-i)} - \sum_{j=1}^{k-1} z_j^{(k-1-j)}$ for $2 \le k \le n-1$. Suppose that the control law is given as $u = y_d^{(n)} - \sum_{i=1}^{n} c_i z_i^{(n-i)} - \sum_{j=1}^{n-1} z_j^{(n-1-j)} - f(\mathbf{x})$, where $c_i > 0$. Then, the state trajectory x_1 can asymptotically track the bounded reference y_d.

3.3 Development of Genetic Adaptive Backsteping Fuzzy-neural Control Scheme

In this section, based on the compact genetic algorithm as mentioned in Chapter 2, a method for developing on-line backstepping adaptive fuzzy-neural controllers is proposed. In addition, in order to instantaneously generate the appropriate control strategy. The compact genetic algorithm has a simplified procedure with an energy fitness function which is used to evaluate the real-time stability of the closed-loop systems.

3.3.1 Designing the Genetic Adaptive Backstepping Fuzzy-neural Controller

Since $f(\mathbf{x})$ is uncertain, the optimal control law (3-13) cannot be obtained. To solve this problem, the fuzzy-neural system is used to approximate the uncertain continuous function $f(\mathbf{x})$. First, the uncertain continuous function $f(\mathbf{x})$ in (3-13) is replaced by fuzzy-neural networks (2-2), i.e., $\hat{f}(\mathbf{x}\,|\,\mathbf{w})$. The resulting control law

$$u_c = y_d^{(n)} - \sum_{i=1}^{n} c_i z_i^{(n-i)} - \sum_{j=1}^{n-1} z_j^{(n-1-j)} - \hat{f}(\mathbf{x}\,|\,\mathbf{w}) \tag{3-18}$$

Suppose that $u = u_c$. Then, substituting (3-18) in (3-1) and after some manipulations, we obtain the error equation

$$\dot{z}_n = f(\mathbf{x}) - \hat{f}(\mathbf{x}\,|\,\mathbf{w}) - c_n z_n - z_{n-1} \tag{3-19}$$

Let $V = \frac{1}{2}\sum_{i=1}^{n} z_i^2$. Using (3-19) we have

$$\begin{aligned}
\dot{V} &= \sum_{i=1}^{n} z_i \dot{z}_i \\
&= z_1(z_2 - c_1 z_1) + \sum_{i=2}^{n-1} z_i(z_{i+1} - c_i z_i - z_{i-1}) \\
&\quad + z_n(f(\mathbf{x}) - \hat{f}(\mathbf{x}\,|\,\mathbf{w}) - c_n z_n - z_{n-1}) \\
&= -\sum_{i=1}^{n} c_i z_i^2 + z_n[f(\mathbf{x}) - \hat{f}(\mathbf{x}\,|\,\mathbf{w})] \\
&\le -\sum_{i=1}^{n} c_i z_i^2 + |z_n|\, f^U(\mathbf{x}) - z_n \hat{f}(\mathbf{x}\,|\,\mathbf{w})
\end{aligned} \tag{3-20}$$

where $|f(\mathbf{x})| \le f^U(\mathbf{x}) < \infty$. In order to instantaneously evaluate the stability of the closed-loop system, we define an energy fitness function as

$$\mathrm{F} = z_n \hat{f}(\mathbf{x}\,|\,\mathbf{w}) \tag{3-21}$$

A chromosome with the largest energy fitness function denotes the optimal

solution. So, a better chromosome can be obtained according to the energy fitness function (3-21).

3.3.2 Safe Controller

The system states may go into the unsafe region if the genetic operations can not instantaneously generate the appropriate weightings of the fuzzy-neural networks in some time interval. Thus, the concept of the safe controller [4] is incorporated into the genetic adaptive backstepping fuzzy-neural controller in order to guarantee that the system states are confined to the safe region. By incorporating a safe control term u_s into u_c, the control law becomes

$$u = u_c + u_s \tag{3-22}$$

The safe control term u_s is added when the function V is greater than a positive limit V^u. If $V \le V^u$, then the safe control term u_s is canceled. That is, if the system tends to enter the unsafe region ($V > V^u$), then u_s forces the system to return to the safe region

Substituting (3-22) into (3-1), the error equation becomes

$$\dot{z}_n = f(\mathbf{x}) - \hat{f}(\mathbf{x} \mid \mathbf{w}) - c_n z_n - z_{n-1} + u_s \tag{3-23}$$

Using (3-22) and (3-23), we have

$$\begin{aligned} \dot{V} &= -\sum_{i=1}^{n} c_i z_i^2 + z_n [f(\mathbf{x}) - \hat{f}(\mathbf{x} \mid \mathbf{w}) + u_s] \\ &\le -\sum_{i=1}^{n} c_i z_i^2 + |z_n| [|f(\mathbf{x})| + \left|\hat{f}(\mathbf{x} \mid \mathbf{w})\right|] + z_n u_s \end{aligned} \tag{3-24}$$

Suppose that the safe control term u_s is given as

$$u_s = \begin{cases} -\text{sgn}(z_n)[f^U(\mathbf{x}) + \left|\hat{f}(\mathbf{x} \mid \mathbf{w})\right|] & \text{if } V > V^u \\ 0 & \text{if } V \le V^u \end{cases} \tag{3-25}$$

where V^u is a design parameter and $\text{sgn}(z_n) = 1(-1)$ if $z_n \ge 0 (< 0)$. Suppose

that $V > V^u$. Then, substituting (3-25) into (3-24), we have

$$\begin{aligned}\dot{V} &\le -\sum_{i=1}^{n} c_i z_i^2 + |z_n|[|f(\mathbf{x})| + |\hat{f}(\mathbf{x}\,|\,\mathbf{w})| - f^U(\mathbf{x}) - |\hat{f}(\mathbf{x}\,|\,\mathbf{w})|] \\ &\le -\sum_{i=1}^{n} c_i z_i^2 \le 0\end{aligned} \tag{3-26}$$

From (3-26), the bounded stability of the closed-loop system for the nonlinear system in (3-1) can be guaranteed.

The overall scheme of the proposed scheme is shown in Fig. 3-1. The design algorithm of proposed method is as follows.

Design algorithm:

Step 1: Construct the fuzzy-neural networks for $\hat{f}(\mathbf{x}\,|\,\mathbf{w})$ including fuzzy sets for $\mathbf{x}$, and the weighting vectors $\mathbf{w}$.

Step 2: Adjust the weighting vectors by using the compact genetic algorithm (2-7)-(2-13) with the energy fitness function (3-21).

Step 3: Compute $\hat{f}(\mathbf{x}\,|\,\mathbf{w})$ according to (2-2). Then, obtain the control law (3-18).

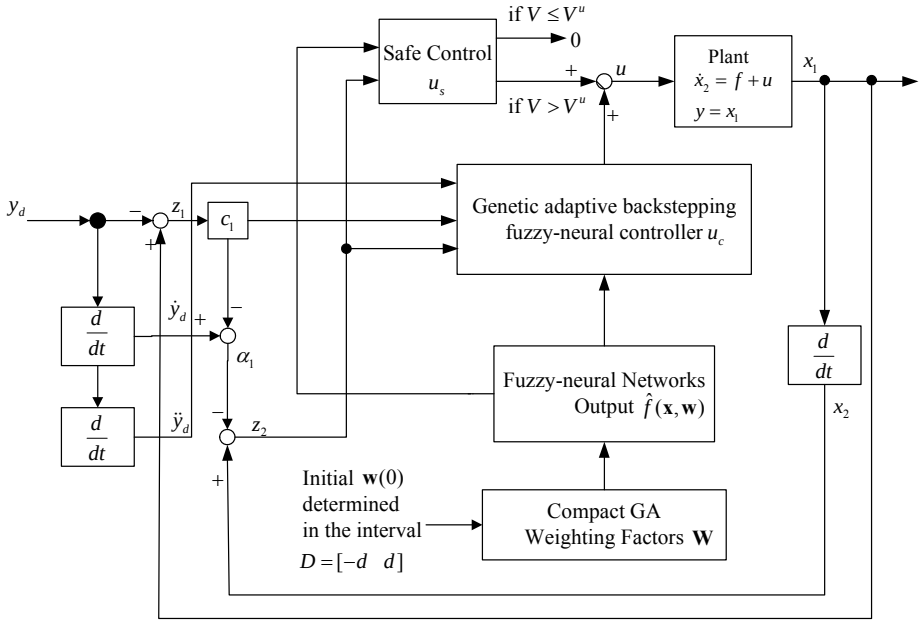

Fig. 3-1. The illustration of the proposed scheme for the 2th order nonlinear system.

3.4 Simulation Examples

This section presents the simulation results of the proposed method for a class of uncertain nonlinear systems to illustrate that the stability of the closed-loop system is guaranteed, and all signal involved are bounded.

Example 3.1: Consider the 2th order nonlinear system described as

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1-e^{-x_1}}{1+e^{-x_1}} + \sin x_1 - x_2 + u = f(x_1, x_2) + u \\ y &= x_1 \end{aligned} \tag{3-27}$$

Our objective is to control the system output y to track the reference trajectory y_d. Clearly, (3-27) is in the form of (3-1). Thus, the proposed method is suitable to control the system. The design parameters of the genetic algorithm are given as *k=4*, p_m=0.06, and $\beta = 4$. The adjustable parameters $\mathbf{w}$ of $\hat{f}(x_1, x_2)$ are in the intervals D=[-0.3,0.3]. The bounded reference signal is given as $y_d(t) = \sin(t)$ in the following simulations. The membership functions for x_i, i=1,2 are given as $\mu_{A_i^1}(x_i) = \frac{1}{(1+\exp(5\times(x_i+3)))}$, $\mu_{A_i^2}(x_i) = \exp(-(x_i+2)^2)$, $\mu_{A_i^3}(x_i) = \exp(-(x_i+1)^2)$, $\mu_{A_i^4}(x_i) = \exp(-(x_i)^2)$, $\mu_{A_i^5}(x_i) = \exp(-(x_i-1)^2)$, $\mu_{A_i^6}(x_i) = \exp(-(x_i-2)^2)$ and $\mu_{A_i^7}(x_i) = \frac{1}{(1+\exp(-5\times(x_i-3)))}$.

To apply the proposed method to the system, the bounds f^U should be obtained as:

$$\left| f(x_1, x_2) \right| = \left| \frac{1-e^{-x_1}}{1+e^{-x_1}} + \sin x_1 - x_2 \right| \le 2 + \left| x_2 \right| \equiv f^U(x_1, x_2) \tag{3-28}$$

The initial states are set as $\mathbf{x}(0) = [1, 0]$. The design parameters are set as $c_1 = 5$, $c_2 = 8$ and $V^u = 0.05$ (case 1) and $V^u = 0.02$ (case 2). The simulation results are shown in Figs. 3-2, 3-3, 3-4, 3-5, 3-6 and 3-7. In the case 1, Figs.

3-2, 3-3 and 3-4 show that the proposed method can control the uncertain nonlinear systems to follow the desired trajectories, and the tracking error does not reach the safe limit ($V \le V^u = 0.05$). In the case 2, Figs. 3-5, 3-6 and 3-7 show that the system output y can track the reference signals $y_d(t)$ very well. Compared with the results of the case 1, the case 2 can achieve the better tracking performance at the expense of the chattering effect of the control input in some time intervals as shown in Fig. 3-7. The chattering effect is due to adding the safe control term when the tracking error exceeds the safe limit ($V > V^u = 0.02$).

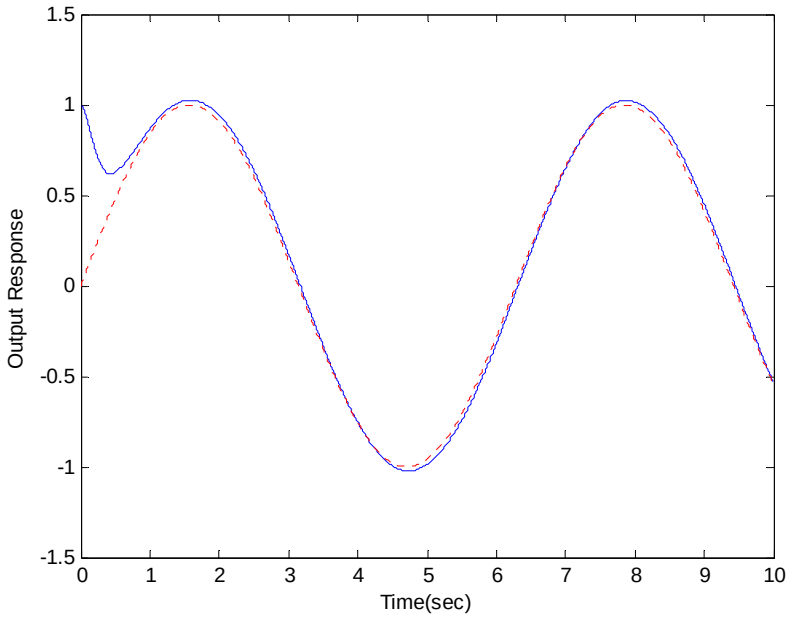

Fig. 3-2. The system output $y(t)= x_1$ (solid line) and bounded reference $y_d(t)$ (dashed line) (case 1).

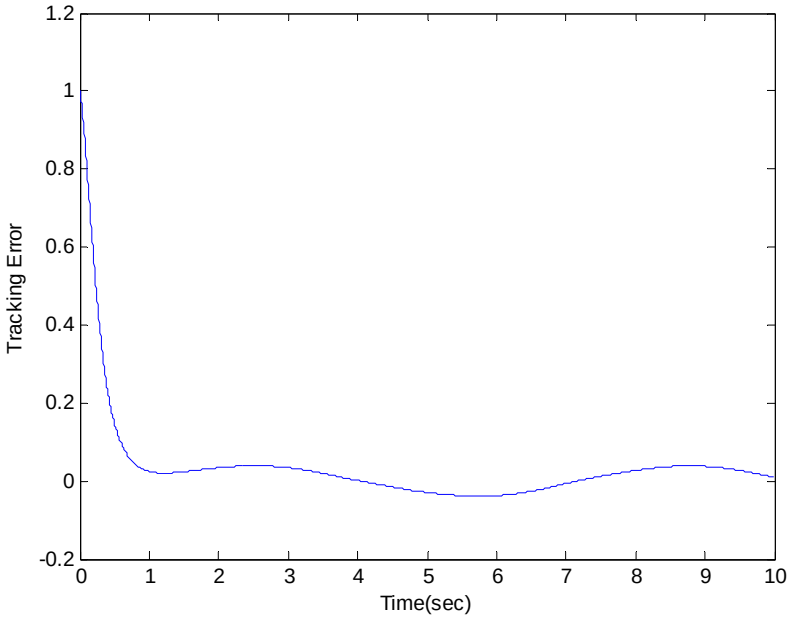

Fig. 3-3. The tracking error $z_1(t)$ (case 1).

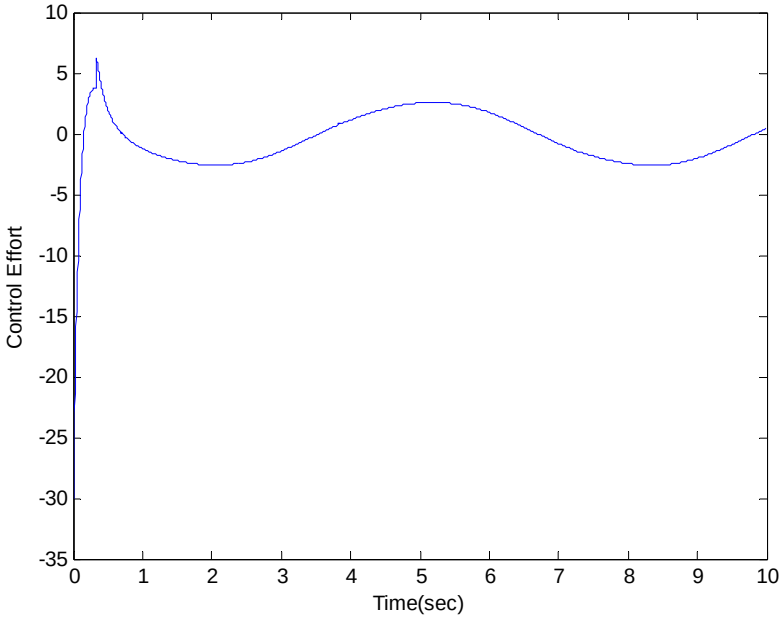

Fig. 3-4. The control effort $u(t)$ (case 1).

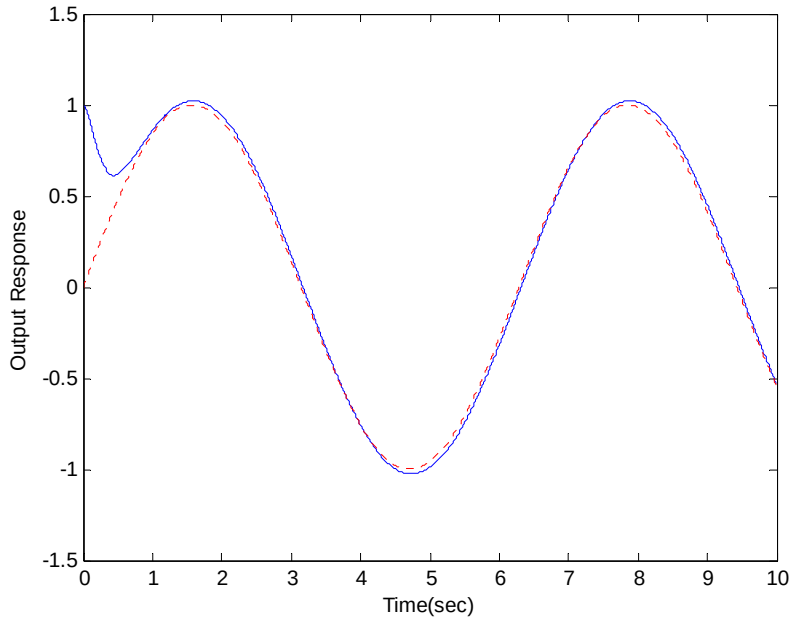

Fig. 3-5. The system output $y(t)= x_1$ (solid line) and bounded reference $y_d(t)$ (dashed line) (case 2).

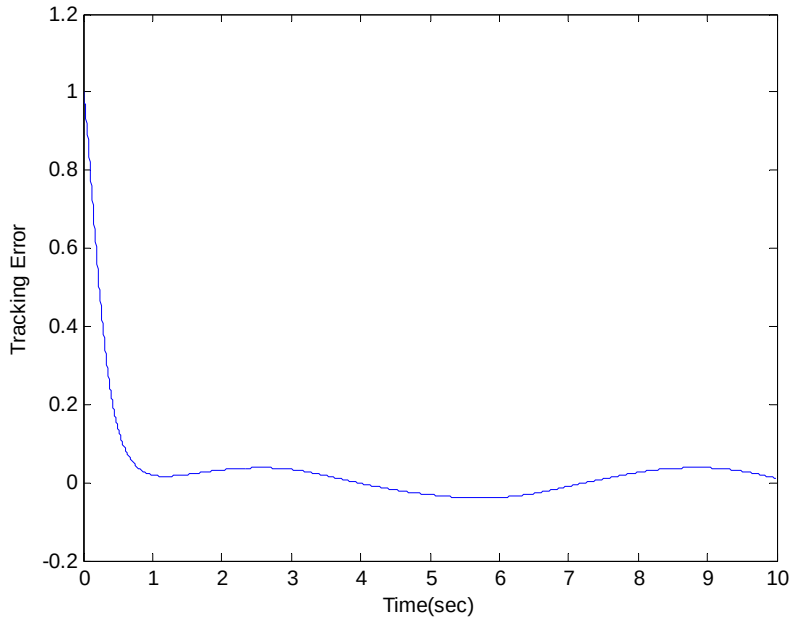

Fig. 3-6. The tracking error $z_1(t)$ (case 2).

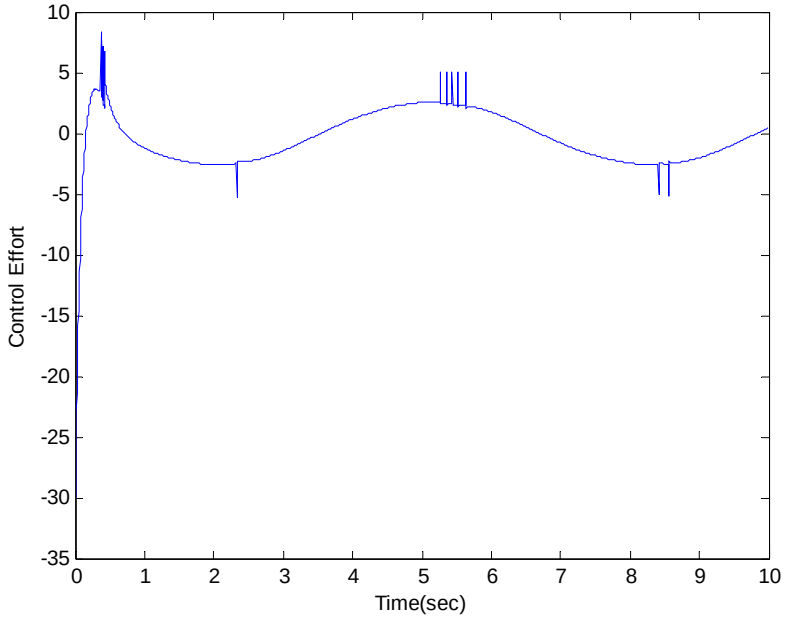

Fig. 3-7. The control effort $u(t)$ (case 2).

Example 3.2: Consider the 3th order nonlinear system described as

$$\begin{aligned}\dot{x}_1 &= x_2\\ \dot{x}_2 &= x_3\\ \dot{x}_3 &= -x_1^3 - 0.1x_2 + 0.2x_3 + 12\cos(t) + u = f(x_1, x_2, x_3) + u\\ y &= x_1\end{aligned} \tag{3-29}$$

The objective is to control the system output y to track the reference trajectory y_d. Clearly, (3-29) is in the form of (3-1). Thus, the proposed method is suitable to control the system. The design parameters of the genetic algorithm are given as k=4, p_m=0.06, and $\beta = 4$. The adjustable parameters $\mathbf{w}$ of $\hat{f}(x_1, x_2, x_3)$ are in the intervals D=[-1,1]. The bounded reference is set as $y_d = \sin(0.5t) + \cos(t)$ in the following simulations. The membership functions for x_i, i=1,2 are the same as those in Example 3-1. The function bounds is obtained as $f^U(x_1, x_2, x_3) = 12 + |x_1|^3 + 0.1|x_2| + 0.2|x_3|$.

The initial states are set as $\mathbf{x}(0) = [0.5, 1, 0.5]$. The design parameters are set as $c_1 = 5$, $c_2 = c_3 = 8$ and $V^u = 1.5$ (case 3) and $V^u = 0.8$ (case 4). The

simulation results are shown in Figs. 3-8, 3-9, 3-10, 3-11, 3-12 and 3-13. In the case 1, Figs. 3-8, 3-9 and 3-10 show that the proposed method can control the uncertain nonlinear systems to follow the desired trajectories, and the tracking error do not reach the safe limit ($V \le V^u = 1.5$). In the case 3, Figs. 3-9, 3-10 and 3-11 show that the system output y can track the reference signals $y_d(t)$ very well. Compared with the results of the case 3, the case 4 can achieve the better tracking performance at the expense of the chattering effect of the control input in some time intervals as shown in Fig. 3-11. The chattering effect is due to adding the safe control term when the tracking error exceeds the safe limit ($V > V^u = 0.8$).

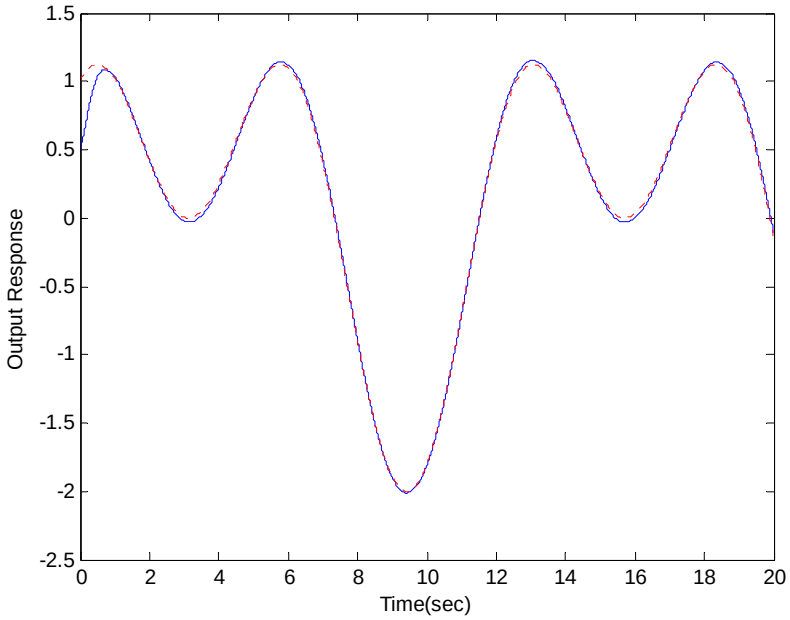

Fig. 3-8. The system output $y(t) = x_1$ (solid line) and bounded reference $y_d(t)$ (dashed line) (case 3).

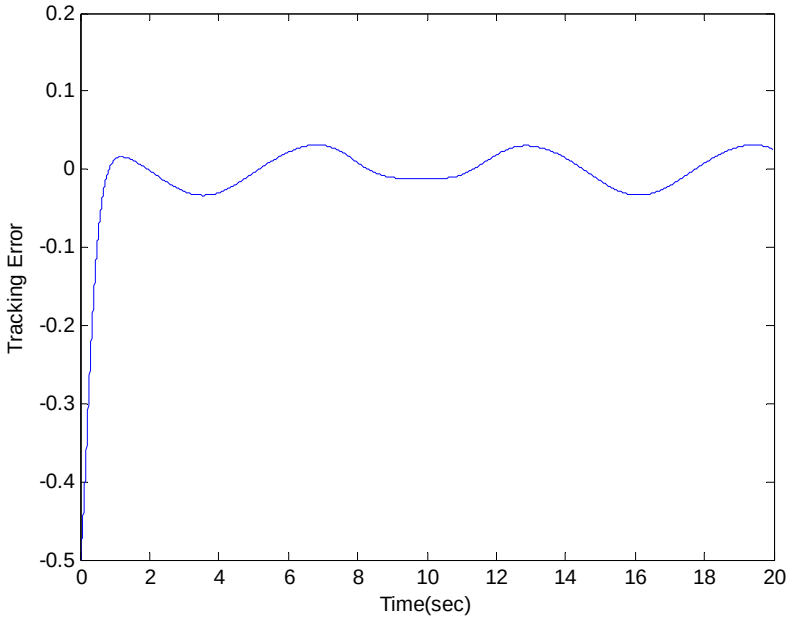

Fig. 3-9. The tracking error $z_1(t)$ (case 3).

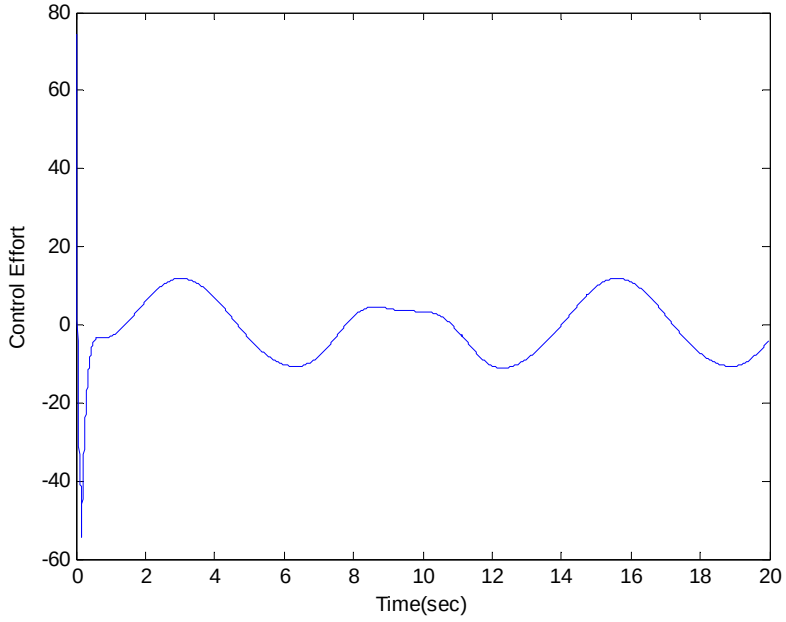

Fig. 3-10. The control effort $u(t)$ (case 3).

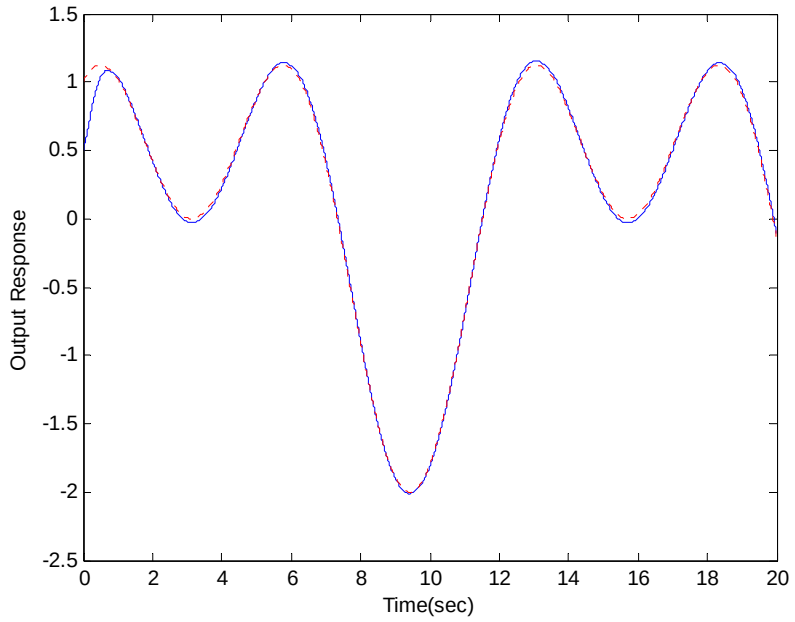

Fig. 3-11. The system output $y(t)= x_1$ (solid line) and bounded reference $y_d(t)$ (dashed line) (case 4).

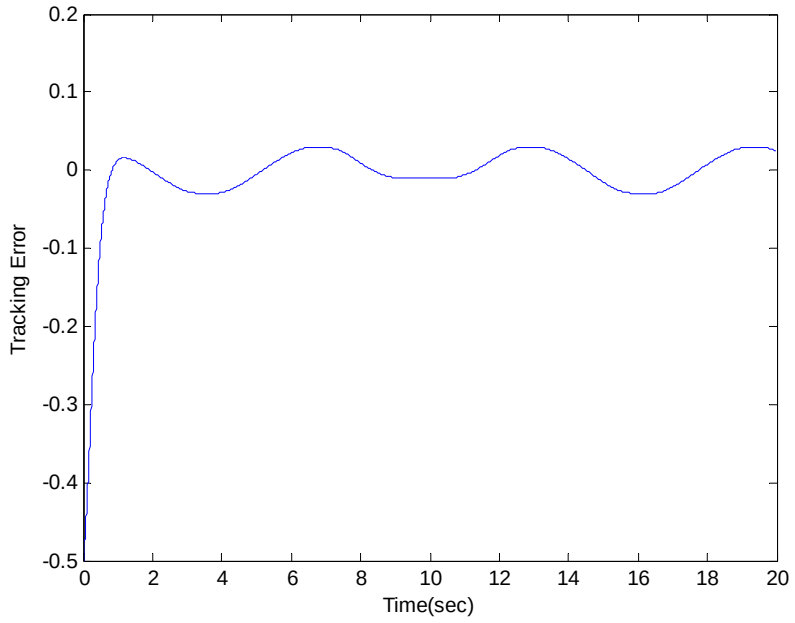

Fig. 3-12. The tracking error $z_1(t)$ (case 4).

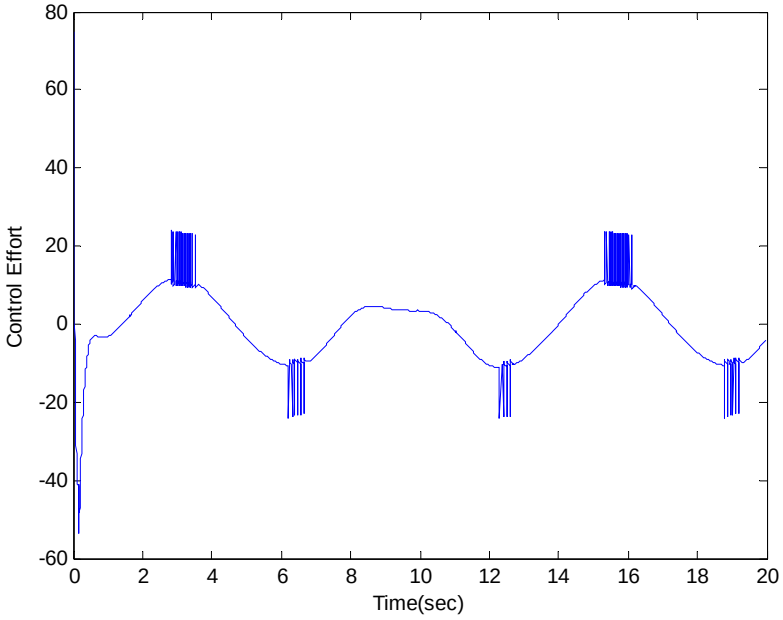

Fig. 3-13. The control effort $u(t)$ (case 4).

3.5 Conclusions

In this chapter, a genetic adaptive fuzzy-neural control scheme has proposed for uncertain nonlinear systems. The free parameters of the genetic adaptive backstepping fuzzy-neural controller can be successfully tuned on-line via the compact genetic algorithm with an energy evaluation mechanism. Using the energy evaluation mechanism can evaluate the real-time stability of the closed-loop systems in order to generate the appropriate control strategy. The simulation results have shown that the genetic adaptive backstepping fuzzy-neural control scheme performs on-line tracking successfully.

Chapter 4

Design of Fuzzy-neural Controller Using Compact Genetic Algorithms for MIMO Nonlinear Systems

4.1 Overview

In this chapter, a genetic adaptive fuzzy-neural control scheme is proposed for a class of multiple-input multiple-output (MIMO) nonlinear systems. The compact genetic algorithm is used to adjust the parameters of the fuzzy-neural networks in order to instantaneously generate the appropriate control strategy. The compact genetic algorithm has a simplified procedure with an energy fitness function which is used to evaluate the real-time stability of the closed-loop systems. In addition, in order to guarantee that the system states are confined to the safe region, a safe control term is incorporated into the control scheme. To illustrate the feasibility and applicability of the proposed method, an example of the double inverted pendulums connected by a spring controlled by the proposed method is provided. The organization of this chapter is summarized as follows. In section 4.2, presents the problem formulation and the construction of the fuzzy-neural networks. Section 4.3 describes the MIMO compact genetic algorithm. Designing the adaptive backstepping fuzzy-neural controller is shown in section 4.4. An illustrative example is given in section 4.5 to show the feasibility and applicability of the proposed method. Conclusions are outlined in section 4.6.

4.2 Problem Formulation and MIMO Fuzzy-Neural Networks

In this section, we describe the control problem for a class of MIMO nonlinear systems, and then design the backstepping controller. In addition, the structure of the MIMO fuzzy-neural networks is briefly reviewed in this section.

4.2.1 Problem Formulation and Backstepping Control Design

First, consider the MIMO nonlinear systems as

$$\begin{aligned} \dot{x}_{p1} &= x_{p2} \\ \dot{x}_{p2} &= x_{p3} \\ &\vdots \\ \dot{x}_{pn_p} &= f_p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_h) + b_p u_p \end{aligned} \quad , p = 1, 2, \ldots, m \qquad (4\text{-}1)$$

where f_p is the unknown system dynamics of the *p*-th subsystem with known upper bound, *i.e.* $\left| f_p(\mathbf{x}) \right| \le f_p^U(\mathbf{x})$, u_p is the input of the *p*-th subsystem, b_p is a positive unknown constant, $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_m]^T$ is the state vector, and $\mathbf{x}_p = [x_{p1}, x_{p2}, ...x_{pn_p}]^T$ is the state vector of the *p*-th subsystem. The reference vector define as $\mathbf{y}_{pd} = [y_{pd}, \dot{y}_{pd}, \ldots, y_{pd}^{(n_p-1)}]$. Our control objective is to develop the backstepping controller so that the state trajectory x_{p1} can asymptotically track a bounded command y_{pd}.

Next, the detail design procedure of the backstepping controller under the

assumption of the known system dynamics f_p is described as follows.

Step 1) Define a tracking error as

$$z_{p1} = x_{p1} - y_{pd} \tag{4-2}$$

Then, differentiating z_{p1} can be expressed as

$$\dot{z}_{p1} = \dot{x}_{p1} - \dot{y}_{pd} \tag{4-3}$$

Define a virtual control as

$$\alpha_{p1} = \dot{y}_{pd} - c_{p1} z_{p1} \tag{4-4}$$

where $c_{p1} > 0$ is a design parameter. From (4-3) and (4-4), if $\alpha_{p1} = \dot{x}_{p1}$, then $\lim_{t\to\infty} z_{p1} \to 0$, that is, the state trajectory x_{p1} can asymptotically track the bounded command y_{pd}. Thus, define an error state as $z_{p2} = \dot{x}_{p1} - \alpha_{p1} = x_{p2} - \alpha_{p1}$. Then, our next goal is to force the error state z_{p2} to decay to zero. By using (4-4) and the fact that $\dot{x}_{p1} = z_{p2} + \alpha_{p1}$, equation (4-3) can be rewritten as

$$\dot{z}_{p1} = z_{p2} - c_{p1} z_{p1} \tag{4-5}$$

Step 2) Differentiating z_{p2} can be expressed as

$$\dot{z}_{p2} = \dot{x}_{p2} - \dot{\alpha}_{p1} = x_{p3} - (-c_{p1}\dot{z}_{p1} + \ddot{y}_{pd}) \tag{4-6}$$

Similarly, define a virtual control as

$$\alpha_{p2} = \ddot{y}_{pd} - c_{p1}\dot{z}_{p1} - c_{p2} z_{p2} - z_{p1} \tag{4-7}$$

where $c_{p2} > 0$ is a design parameter. Moreover, define an error state as $z_{p3} = x_{p3} - \alpha_{p2}$. Then, by using (4-7) and the fact that $\dot{x}_{p2} = z_{p3} + \alpha_{p2}$, equation (4-6) can be rewritten as

$$\dot{z}_{p2} = z_{p3} - c_{p2} z_{p2} - z_{p1} \tag{4-8}$$

Step 3) Let k be a positive integer. Define an error state as $z_{pk} = x_{pk} - \alpha_{p(k-1)}$. Then, differentiating z_{pk}, where $3 \le k \le n_p - 1$, can be expressed as

$$\dot{z}_{pk} = \dot{x}_{pk} - \dot{\alpha}_{p(k-1)} \tag{4-9}$$

Define a virtual control as

$$\alpha_{pk} = y_{pd}^{(k)} - \sum_{i=1}^{k} c_{pi} z_{pi}^{(k-i)} - \sum_{j=1}^{k-1} z_{pj}^{(k-1-j)} \tag{4-10}$$

where $c_{pi} > 0$ is a design parameter. Moreover, define an error state as $z_{p(k+1)} = x_{p(k+1)} - \alpha_{pk}$. Then, by using (4-10) and the fact that $\dot{x}_{pk} = z_{p(k+1)} + \alpha_{pk}$, equation (4-9) can be rewritten as

$$\dot{z}_{pk} = z_{p(k+1)} - c_{pk} z_{pk} - z_{p(k-1)} \tag{4-11}$$

Step 4) Differentiating z_{pn_p} can be expressed as

$$\dot{z}_{pn_p} = \dot{x}_{pn_p} - \dot{\alpha}_{p(n_p-1)} = f_p + b_p u_p - \dot{\alpha}_{p(n_p-1)} \tag{4-12}$$

Define a control law as

$$u_p = -\frac{1}{b_p}(f_p - \dot{\alpha}_{p(n_p-1)}) - c_{pn_p} z_{pn_p} - z_{p(n_p-1)} \tag{4-13}$$

where $c_{pn_p} > 0$ is a design parameter. Let $\overline{f}_p = \frac{1}{b_p}(f_p - \dot{\alpha}_{p(n_p-1)})$,then (4-40) can be expressed as.

$$u_p = -\overline{f}_p - c_{pn_p} z_{pn_p} - z_{p(n_p-1)} \tag{4-14}$$

Then, from (4-13), equation (4-12) can be rewritten as

$$\dot{z}_{pn_p} = b_p(-c_{pn_p} z_{pn_p} - z_{p(n_p-1)}) \tag{4-15}$$

Step 5) Consider the Lyapunov function as follows

$$V = \frac{1}{2}\sum_{p=1}^{m}\sum_{i=1}^{n_p-1} z_{pi}^2 + \sum_{p=1}^{m}\frac{1}{2b_p} z_{pn_p}^2 \tag{4-16}$$

By differentiating (4-15) and using (4-5), (4-8), (4-11) and (4-15), we have

$$\begin{aligned}\dot{V} &= \sum_{p=1}^{m}\sum_{i=1}^{n_p-1} z_{pi}\dot{z}_{pi} + \sum_{p=1}^{m}\frac{1}{b_p} z_{pn_p}\dot{z}_{pn_p} = -\sum_{p=1}^{m}\sum_{i=1}^{n_p} c_{pi} z_{pi}^2 \\ &\le -\sum_{p=1}^{m} c_{p1} z_{p1}^2\end{aligned} \tag{4-17}$$

From (4-16) and (4-17), we can conclude that z_{pi} is bounded. Moreover, from (4-5), $\dot{z}_{p1}$ is also bounded. Integrating (4-17) can be expressed as

$$\sum_{p=1}^{m} c_{p1}\int_0^{\infty} z_{p1}^2(\tau)d\tau \le -V(\infty) + V(0) \tag{4-18}$$

Because the fact that the right side of (4-17) is bounded, we have $z_{p1} \in L_2$. According to the Barbalat's Lemma, $\lim_{t\to\infty} z_{p1} = 0$, that is, the state trajectory x_{p1} can asymptotically track the bounded command y_{pd}.

On the basis of the aforementioned description, the backstepping controller of the *n*th order nonlinear system can be summarized as the following lemma.

Lemma 4.1: Consider the *n*th-order MIMO nonlinear systems (1). Let $z_{p1} = x_{p1} - y_{pd}$ and $z_{pi} = x_{pi} - \alpha_{p(i-1)}$ for $2 \le i \le n_p$, where $\alpha_{p1} = \dot{y}_{pd} - c_{p1}z_{p1}$, and $\alpha_{pk} = y_{pd}^{(k)} - \sum_{i=1}^{k} c_{pi} z_{pi}^{(k-i)} - \sum_{j=1}^{k-1} z_{pj}^{(k-1-j)}$ for $2 \le k \le n_p - 1$. Suppose that the control law is given as $u_p = -\frac{1}{b_p}(f_p - \dot{\alpha}_{p(n_p-1)}) - c_{pn_p} z_{pn_p} - z_{p(n_p-1)}$, where

$c_{pi} > 0$. Then, the state trajectory x_{p1} can asymptotically track the bounded command y_{pd}.

4.2.2 Description of MIMO Fuzzy-neural Networks

The fuzzy inference engine uses the fuzzy IF-THEN rules to perform a mapping from an training input data x_{pq}, $p = 1, 2, \cdots, m$, $q = 1, 2, \cdots, n_p$, and the output data y_p, $p = 1, 2, \cdots, m$, the ith fuzzy rule has the following form:

$$\begin{aligned} R^i : IF \quad & x_{11} \quad is \quad A_{11}^i \quad and \cdots x_{mn_m} \quad is \quad A_{mn_m}^i \\ THEN \quad & y_1 \quad is \quad B_1^i \quad and \cdots y_m \quad is \quad B_m^i \end{aligned} \tag{4-19}$$

where i is a rule number, A_{pq}^i and B_p^i are the fuzzy sets. By using product inference, center-average and singleton fuzzifier, the output of the fuzzy-neural networks can be expressed as:

$$y_p(\mathbf{x}|w_p) = \frac{\sum_{i=1}^{N} w_p^i \left(\prod_{s=1}^{m}\prod_{q=1}^{n_s} \mu_{A_{sq}^i}(x_{sq})\right)}{\sum_{i=1}^{N} \left(\prod_{s=1}^{m}\prod_{q=1}^{n_s} \mu_{A_{sq}^i}(x_{sq})\right)} = \mathbf{w}_p^T \varphi(\mathbf{x}) \tag{4-20}$$

where $\mu_{A_{sq}^i}(x_{sq})$ is the membership function of A_{sq}^i, N is the total number of the IF-THEN rules, w_p^i is the point at which $\mu_{B^i}\left(w_p^i\right) = 1$, $\mathbf{w}_p = [w_p^1 \; w_p^2 \cdots w_p^N]^T$ is a weighting vector, $\varphi^T = [\varphi^1 \; \varphi^2 \cdots \varphi^N]$ is the fuzzy basis vector, where φ^i is defined as

$$\varphi^i(\mathbf{x}) = \frac{\prod_{s=1}^{m}\prod_{q=1}^{n_s} \mu_{A_{sq}^i}(x_{sq})}{\sum_{i=1}^{N} \left(\prod_{s=1}^{m}\prod_{q=1}^{n_s} \mu_{A_{sq}^i}(x_{sq})\right)} \tag{4-21}$$

The typical fuzzy-neural networks have a total of four layers. Nodes at

layer I are input nodes that represent input linguistic variables. Nodes at layer II represent the values of the membership function of total linguistic variables. At layer III, nodes are the values of the fuzzy basis vector φ. Each node at layer III is a fuzzy rule. The links between layer III and layer IV are full connected by the weighting factors. Layer IV is the output layer.

4.3 Description of MIMO Compact Genetic Algorithm

For the purpose of speeding up the computation of the genetic operation, the mechanism of the compact genetic algorithm has three simplified parts: (1) keep a small population size, (2) replace the string codes with real-value representation, and (3) perform compact mutation and crossover on the chromosomes by the energy fitness function. The details are discussed in the following.

First, for the adjustable parameters of the *p*-th output of the fuzzy-neural networks, define a population of solutions with *k* chromosomes as

$$\Xi_p = \begin{bmatrix} {}^1\mathbf{w}_p^T \\ {}^2\mathbf{w}_p^T \\ \vdots \\ {}^k\mathbf{w}_p^T \end{bmatrix} \tag{4-22}$$

where ${}^i\mathbf{w}_p$ denotes the *i*th chromosome and is a set of weighting factors in the interval $\mathbf{D}_p = [-d_p, d_p]$, $d_p > 0$. Each gene represents the adjustable parameter of the fuzzy-neural networks. Note that the population is sorted by ranking the fitness of chromosomes, that is, the first chromosome denotes the best chromosome in the population in terms of fitness.

Next, to instantaneously evaluate the stability of the closed-loop system,

define the *p*-th energy fitness function for the *p*-th subsystem as

$$F_p = z_{pn_p} \hat{\bar{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) \tag{4-23}$$

where $\hat{f}_p(\mathbf{x} \mid \mathbf{w}_p)$ is the estimation of the unknown dynamic $f_p(\mathbf{x} \mid \mathbf{w}_p)$. A chromosome with largest fitness denotes the optimal solution. The detail explanation of the energy fitness function is given later.

Then, according to the energy fitness function, mutation and crossover operators are performed. The operation procedure of the crossover operator is as follows. Suppose that the j-th gene is selected as the crossover point. Then, the j-th gene of all chromosomes performs a linear combination operation and is updated as

$${}^{i}w_p^j = \begin{cases} {}^{i}w_p^j * {}^{i}\zeta_p + {}^{i+(k/2)}w_p^j * (1 - {}^{i}\zeta_p) & \text{for } i = 1, 2, ..., k/2 \\ {}^{i}w_p^j * {}^{i}\kappa_p + {}^{i-(k/2)}w_p^j * (1 - {}^{i}\kappa_p) & \text{for } i = (k/2)+1, (k/2)+2, ..., k \end{cases} \tag{4-24}$$

where the crossover factors ${}^{i}\zeta_p$ and ${}^{i}\kappa_p$ are the linear combination weights and defined as

$${}^{i}\zeta_p = \frac{{}^{i}F_p - {}^{k}F_p}{[{}^{i}F_p - {}^{k}F_p] + [{}^{i+(k/2)}F_p - {}^{k}F_p]} \tag{4-25}$$

$${}^{i}\kappa_p = \frac{{}^{i}F_p - {}^{k}F_p}{[{}^{i}F_p - {}^{k}F_p] + [{}^{i-(k/2)}F_p - {}^{k}F_p]} \tag{4-26}$$

respectively.

As for the operation procedure of the mutation operator, the (k/2+1)-th chromosome is replaced by the first chromosome according to the mutation rate p_m. Then, the (k/2+1)-th chromosome is updated as

$${}^{(k/2+1)}w_p^j = \begin{cases} {}^{1}w_p^j + \gamma^{\beta} * \eta_p, & \text{if } \delta > 0.5 \\ {}^{1}w_p^j, & \text{if } \delta \le 0.5 \end{cases} \tag{4-27}$$

where

$$\eta_p = \begin{cases} d_p - {}^1w_p^j & \text{,if } \lambda > 0.5 \\ -d_p - {}^1w_p^j & \text{,if } \lambda \le 0.5 \end{cases} \tag{4-28}$$

and $\delta, \gamma, \lambda \in [0,1]$ are random numbers, and $\beta > 0$ is a design parameter.

4.4 Development of Genetic Adaptive Fuzzy-neural Control Scheme

In this section, based on the compact genetic algorithm as mentioned above, a method for developing on-line backstepping adaptive fuzzy-neural controllers is proposed.

Since f_p and b_p are uncertain, the optimal control law (4-13) cannot be obtained. To solve this problem, the fuzzy-neural system is used to approximate the uncertain continuous function $\overline{f}_p$. First, the uncertain continuous function $\overline{f}_p$ in (4-13) is replaced by fuzzy-neural networks (4-20), i.e., $\hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p)$. The resulting control law

$$u_{pc} = -\hat{\overline{f}}_p - c_{pn_p} z_{pn_p} - z_{p(n_p-1)} \tag{4-29}$$

Suppose that $u_p = u_{pc}$. Then, substituting (4-29) in (4-1) and after some manipulations, we obtain the error equation

$$\dot{z}_{pn_p} = b_p(\overline{f}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd}) - \hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) - c_{pn_p} z_{pn_p} - z_{p(n_p-1)}) \tag{4-30}$$

Let $V = \frac{1}{2}\sum_{p=1}^{m}\sum_{i=1}^{n_p-1} z_{pi}^2 + \sum_{p=1}^{m}\frac{1}{2b_p} z_{pn_p}^2$. By using (4-30), we have

$$
\begin{aligned}
\dot{V} &= \sum_{p=1}^{m}\sum_{i=1}^{n_p-1} z_{pi}\dot{z}_{pi} + \sum_{p=1}^{m}\frac{1}{b_p} z_{pn_p}\dot{z}_{pn_p} \\
&= \sum_{p=1}^{m}[z_{p1}(z_{p2} - c_{p1}z_{p1}) + \sum_{i=2}^{n_p-1} z_{pi}(z_{p(i+1)} - c_{pi}z_{pi} - z_{p(i-1)}) \\
&\quad + z_{pn_p}(\overline{f}_p(\mathbf{x},\dot{\mathbf{y}}_{pd}) - \hat{\overline{f}}_p(\mathbf{x},\dot{\mathbf{y}}_{pd}\,|\,\mathbf{w}_p) - c_{pn_p}z_{pn_p} - z_{p(n_p-1)})] \qquad (4\text{-}31) \\
&= \sum_{p=1}^{m}[\sum_{i=1}^{n_p} -c_{pi}z_{pi}^2 + z_{pn_p}(\overline{f}_p(\mathbf{x},\dot{\mathbf{y}}_{pd}) - \hat{\overline{f}}_p(\mathbf{x},\dot{\mathbf{y}}_{pd}\,|\,\mathbf{w}_p))] \\
&\le \sum_{p=1}^{m}[-\sum_{i=1}^{n_p} c_{pi}z_{pi}^2 + \left|z_{pn_p}\right| \overline{f}_p{}^{U}(\mathbf{x},\dot{\mathbf{y}}_{pd}) - z_{pn_p}\hat{\overline{f}}_p(\mathbf{x},\dot{\mathbf{y}}_{pd}\,|\,\mathbf{w}_p)]
\end{aligned}
$$

where $\left|\overline{f}_p(\mathbf{x},\dot{\mathbf{y}}_{pd})\right| \le \overline{f}_p{}^{U}(\mathbf{x},\dot{\mathbf{y}}_{pd}) < \infty$. By using the fact that the states must move in the direction of smaller values of V if $\dot{V} < 0$, the *p*-th energy fitness function $z_{pn_p}\hat{\overline{f}}_p(\mathbf{x},\dot{\mathbf{y}}_{pd}\,|\,\mathbf{w}_p)$ in (4-23) for the *p*-th subsystem is defined in order to instantaneously evaluate the stability of the closed-loop system. A chromosome with the largest energy fitness function denotes the optimal solution. So, a better chromosome can be obtained according to the energy fitness function (4-23).

Since the system states may go into the unsafe region if the genetic operations can not instantaneously generate the appropriate weightings of the fuzzy-neural networks in some time interval. Thus, the concept of the safe controller is incorporated into the genetic adaptive backstepping fuzzy-neural controller in order to guarantee that the system states are confined to the safe region. By incorporating a safe control term u_{ps} into u_{pc}, the control law becomes

$$u_p = u_{pc} + u_{ps} \qquad (4\text{-}32)$$

The safe control term u_{ps} is added when the function V is greater than a positive limit V^u. If $V \le V^u$, then the safe control term u_{ps} is canceled. That

is, if the system tends to enter the unsafe region ($V > V^u$), then u_{ps} forces the system to return to the safe region.

Substituting (4-32) into (4-1), the error equation becomes

$$\dot{z}_{pn_p} = b_p(\overline{f}_p(\mathbf{x}) - \hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) - c_{pn_p} z_{pn_p} - z_{p(n_p-1)}) + b_p u_{ps} \tag{4-33}$$

Using (4-32) and (4-33), we have

$$\begin{aligned} \dot{V} &= \sum_{p=1}^{m}[-\sum_{i=1}^{n_p} c_{pi} z_{pi}^2 + z_{pn_p}(\overline{f}_p(\mathbf{x}) - \hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) + u_{ps})] \\ &\le \sum_{p=1}^{m}[-\sum_{i=1}^{n_p} c_{pi} z_{pi}^2 + \left| z_{pn_p} \right| (\left| \overline{f}_p(\mathbf{x}) \right| + \left| \hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) \right|) + z_{pn_p} u_{ps}] \end{aligned} \tag{4-34}$$

Suppose that the safe control term u_{ps} is given as

$$u_{ps} = \begin{cases} -\text{sgn}(z_{pn})[\overline{f}_p^{\,U}(\mathbf{x}) + \left| \hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) \right|] & \text{if } V > V^u \\ 0 & \text{if } V \le V^u \end{cases} \tag{4-35}$$

where V^u is a design parameter and $\text{sgn}(z_{pn}) = 1(-1)$ if $z_{pn} \ge 0(< 0)$.

Suppose that $V > V^u$. Then, substituting (4-35) into (4-34), we have

$$\begin{aligned} \dot{V} &\le \sum_{p=1}^{m}[-\sum_{i=1}^{n_p} c_{pi} z_{pi}^2 + \left| z_{pn_p} \right| [\left| \overline{f}_p(\mathbf{x}) \right| + \left| \hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) \right| - (\overline{f}_p^{\,U}(\mathbf{x}) + \left| \hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p) \right|)] \\ &\le -\sum_{p=1}^{m}\sum_{i=1}^{n_p} c_{pi} z_{pi}^2 \le 0 \end{aligned} \tag{4-36}$$

From (4-36), the bounded stability of the closed-loop system for the nonlinear system in (4-1) can be guaranteed.

The design algorithm of proposed method is as follows.

Design algorithm:

Step 1: Construct the fuzzy-neural networks for $\hat{\overline{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p)$ including fuzzy sets for $\mathbf{x}$, $\dot{\mathbf{y}}_{pd}$ and the weighting vectors $\mathbf{w}_p$.

Step 2: Adjust the weighting vectors by using the compact genetic algorithm (4-22)-(4-28) with the energy fitness function (4-23).

Step 3: Compute $\hat{\bar{f}}_p(\mathbf{x}, \dot{\mathbf{y}}_{pd} \mid \mathbf{w}_p)$ according to (4-20). Then, obtain the control law (4-29).

4.5 Simulation Results

This section presents the simulation results of the proposed method for a class of MIMO nonlinear systems, and then illustrates that the stability of the closed-loop system is guaranteed.

Example 4.1: Consider the following problem of balancing double inverted pendulums connected by a spring [47]

$$\begin{aligned}
\dot{x}_{11} &= x_{12} \\
\dot{x}_{12} &= \frac{m_1 gr}{J_1}\sin(x_{11}) - \frac{k}{J_1}x_{11} + \frac{u_1}{J_1} + \frac{k}{J_1}x_{21} \\
\dot{x}_{21} &= x_{22} \\
\dot{x}_{22} &= \frac{m_2 gr}{J_2}\sin(x_{21}) - \frac{k}{J_2}x_{21} + \frac{u_2}{J_2} + \frac{k}{J_2}x_{11}
\end{aligned} \tag{4-37}$$

where x_{i1} is the angular position of the ith pendulum from the vertical reference, and u_i is a torque input. It is assumed that both x_{i1} and $\dot{x}_{i1}$ are available for measurement. The parameters of the double inverted pendulums are chosen as $m_1 = 0.5$ kg $m_2 = 0.5$ kg, $J_1 = 0.5$ kg and $J_2 = 0.5$ kg, $k = 2\,\mathrm{N}\cdot\mathrm{m/rad}$, and $r = 1\,\mathrm{m}$.

Our objective is to control the system state x_{p1} to track the reference trajectory y_{pd}. It is clear that from (4-37) and (4-1), the proposed method is suitable to control the system. The design parameters of the genetic algorithm

are given as k=4, $p_m = 0.06$, and β=4. The adjustable parameters w_i of $\hat{\bar{f}}_i(x_{11}, x_{12}, x_{21}, x_{22}, \dot{y}_{id}, \ddot{y}_{id})$ are in the intervals D_i=[-1,1]. The reference signals are set as $y_{1d}(t) = (2.5\pi/12)\sin(t)$ and $y_{2d}(t) = (3.75\pi/12)\cos(t)$. The membership functions for x_{pi}, i=1,2 are given as

$$\begin{aligned}
\mu_{A_i^1}(x_{pi}) &= \exp[-(x_i+1)^2] \\
\mu_{A_i^2}(x_{pi}) &= \exp[-(x_i+0.5)^2] \\
\mu_{A_i^3}(x_{pi}) &= \exp[-x_i^2] \\
\mu_{A_i^4}(x_{pi}) &= \exp[-(x_i-0.5)^2] \\
\mu_{A_i^5}(x_{pi}) &= \exp[-(x_i-1)^2]
\end{aligned} \tag{4-38}$$

To apply the proposed method to the system, the bounds $\bar{f}_p^U$ for p=1,2 should be obtained as

$$\begin{aligned}
|f_1(x_1,x_2,x_3,x_4)| &= \left|\frac{m_1 gr}{J_1}\sin(x_{11}) - \frac{k}{J_1}x_{11} + \frac{k}{J_1}x_{21}\right| \le 9.8 + 4|x_{11}| + 4|x_{21}| \\
&\equiv f_1^U(x_1,x_2,x_3,x_4) \qquad (4\text{-}39) \\
\bar{f}_1^U &= \frac{1}{b_1^L}(f_1^U + |\dot{\alpha}_{11}|) = \frac{1}{b_1^L}(f_1^U + |c_{11}\dot{z}_{11} + \ddot{y}_{1d}|)
\end{aligned}$$

$$\begin{aligned}
|f_2(x_1,x_2,x_3,x_4)| &= \left|\frac{m_2 gr}{J_2}\sin(x_{21}) - \frac{k}{J_2}x_{21} + \frac{k}{J_2}x_{11}\right| \le 9.8 + 4|x_{21}| + 4|x_{11}| \\
&\equiv f_2^U(x_1,x_2,x_3,x_4) \qquad (4\text{-}40) \\
\bar{f}_2^U &= \frac{1}{b_2^L}(f_2^U + |\dot{\alpha}_{21}|) = \frac{1}{b_2^L}(f_2^U + |c_{21}\dot{z}_{21} + \ddot{y}_{2d}|)
\end{aligned}$$

where $0 < b_1^L \le b_1$, $0 < b_2^L \le b_2$ and given as $b_{1L} = b_{2L} = 1.5$.

The initial states are set as $x(0) = [0.5, 0, 0.3, 0]$. The design parameters are selected as c_{11}=10, c_{12}=10, c_{21}=8, c_{22}=8, $V^u = 0.3$ (case 1) and $V^u = 0.02$ (case 2). The simulation results are shown in Figs. 4-1, 4-2, 4-3, 4-4, 4-5, 4-6, 4-7 and 4-8. In the case 1, Figs. 4-1, 4-2, 4-3 and 4-4 show that the

proposed control algorithm can control the MIMO uncertain nonlinear systems to follow the desired trajectories, and the tracking error do not reach the safe limit ($V \le V^u = 0.3$). In the case 2, Figs. 4-5, 4-6, 4-7 and 4-8 show that the system outputs x_{11} and x_{21} can track the reference signals $y_{1d}(t)$ and $y_{2d}(t)$ very well, respectively. Compared with the results of the case 1, the case 2 can achieve the better tracking performance at the expense of the chattering effect of the control input in some time intervals as shown in Fig. 4-7. The chattering effect is due to adding the safe control term when the tracking error exceeds the safe limit ($V > V^u = 0.02$).

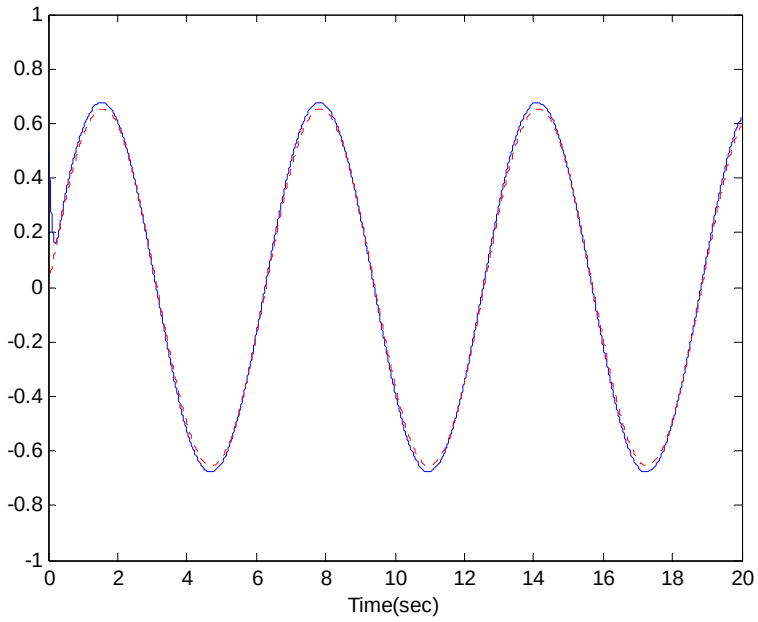

Fig. 4-1. The system output $x_{11}(t)$ (solid line) and bounded reference $y_{1d}(t)$ (dashed line) (case 1).

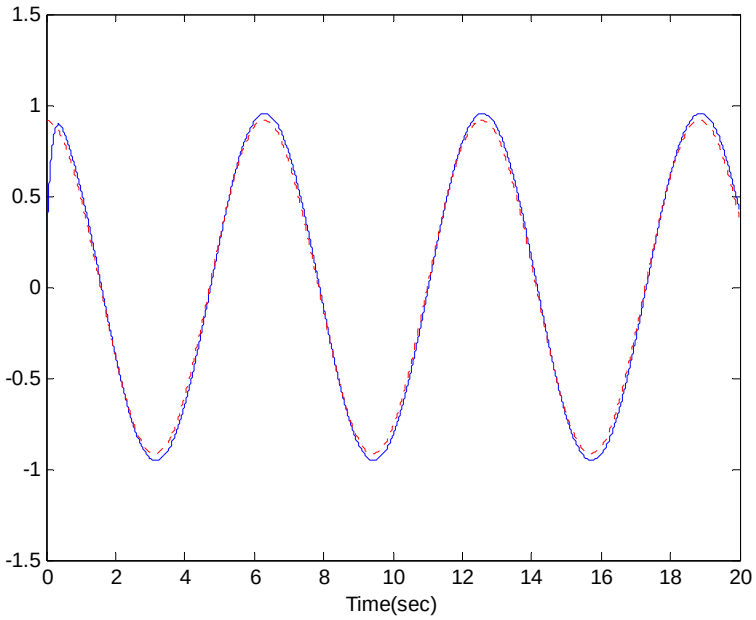

Fig. 4-2. The system output $x_{21}(t)$ (solid line) and bounded reference $y_{2d}(t)$ (dashed line) (case 1).

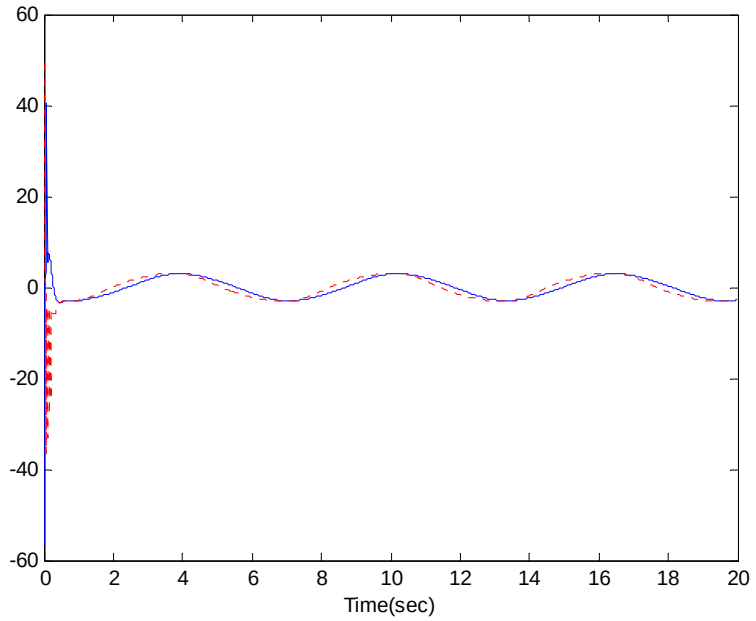

Fig. 4-3. The control effort u_1 (solid line) and u_2 (dashed line) (case 1).

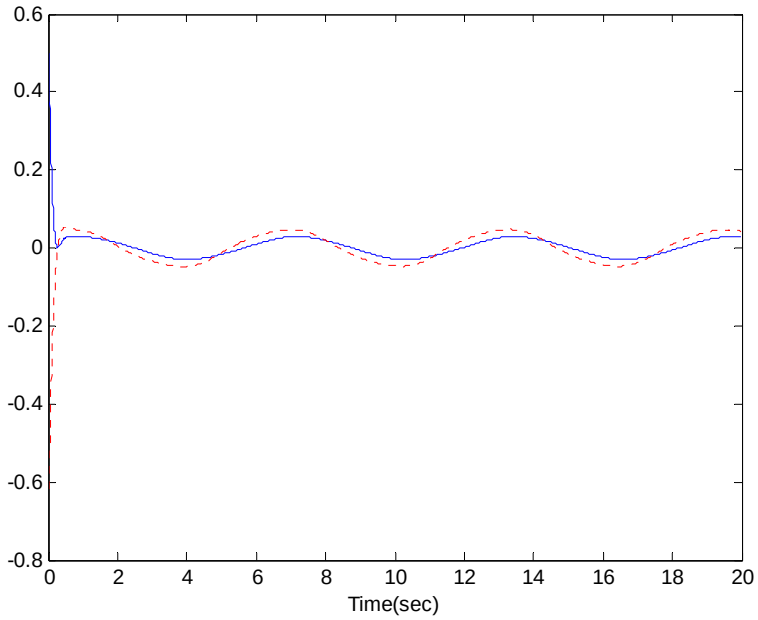

Fig. 4-4. The tracking errors $z_{11}(t)$ (solid line) and $z_{21}(t)$ (dashed line) (case 1).

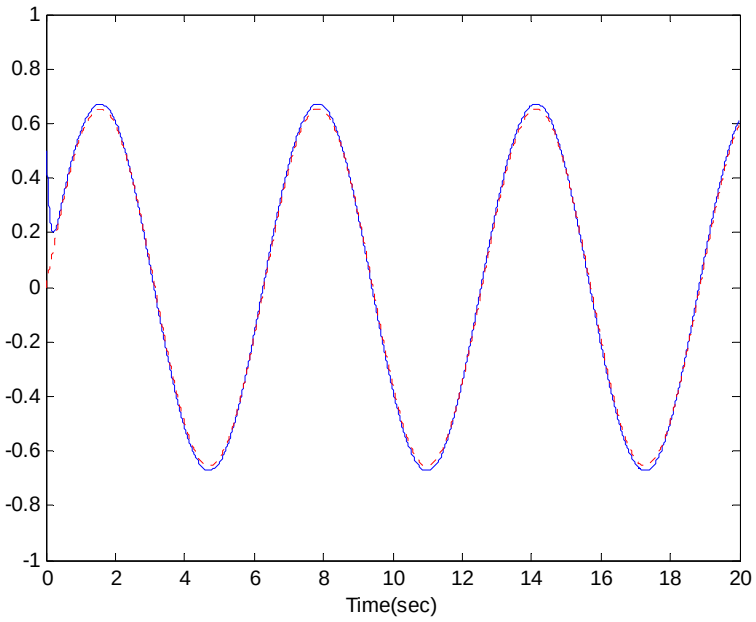

Fig. 4-5. The system output $x_{11}(t)$ (solid line) and bounded reference $y_{1d}(t)$ (dashed line) (case 2).

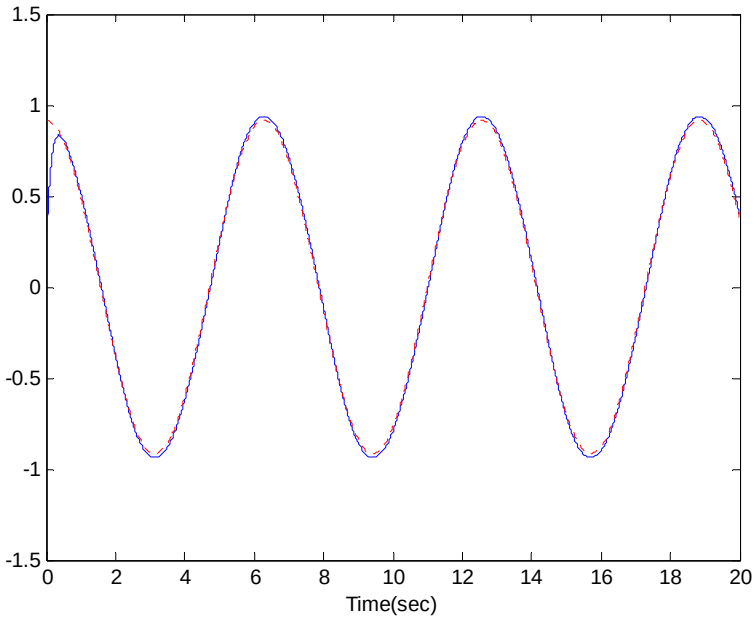

Fig. 4-6. The system output $y_{21}(t)$ (solid line) and bounded reference $y_{2d}(t)$ (dashed line) (case 2).

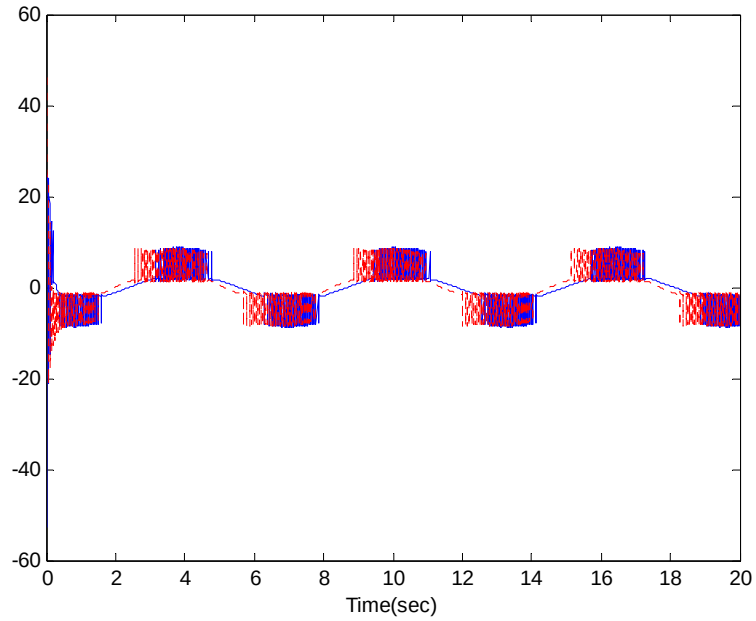

Fig. 4-7. The control effort u_1 (solid line) and u_2 (dashed line) (case 2).

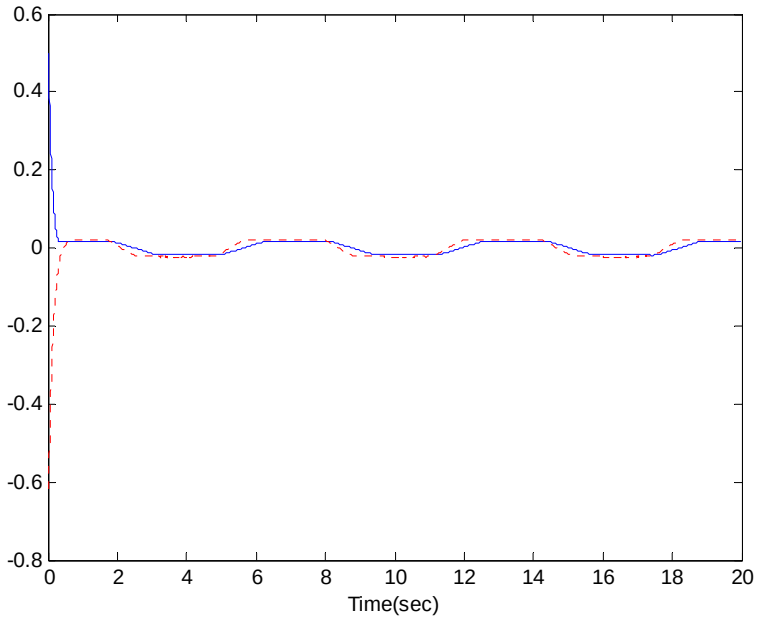

Fig. 4-8. The tracking errors $z_{11}(t)$ (solid line) and $z_{21}(t)$ (dashed line) (case 2).

4.6 Conclusions

In this chapter, a genetic adaptive fuzzy-neural control scheme has proposed for a class of MIMO nonlinear systems. The weighting parameters of the fuzzy-neural controller can be successfully tuned instantaneously via the compact genetic algorithm with an energy evaluation mechanism. Using the energy evaluation mechanism can evaluate the real-time stability of the closed-loop systems in order to generate the appropriate control strategy. Moreover, the proposed design algorithm has been successfully applied to control double inverted pendulums connected by a spring. The simulation results have shown that the genetic adaptive backstepping fuzzy-neural control scheme performs on-line tracking successfully.

Chapter 5

Design of Fuzzy-neural Controller Using Compact Genetic Algorithms for Robot Manipulators

5.1 Overview

A robot manipulator is an uncertain nonlinear dynamic system which suffers from structured and unstructured uncertainties such as load variation, friction, and external disturbances. Therefore, in order to control the robot manipulator, it is necessary that a control algorithm has robustness to uncertainties. To solve the tracking control of uncertain MIMO robot system, a genetic adaptive fuzzy-neural control scheme is proposed in this chapter. Fuzzy-neural networks are used to approximate complicated nonlinear functions. A compact genetic algorithm is used to adjust the parameters of the fuzzy-neural networks in order to instantaneously generate the appropriate control strategy. The compact genetic algorithm has a simplified procedure with an energy fitness function which is used to evaluate the real-time stability of the closed-loop systems. Finally, simulation results are provided to demonstrate the applicability of the proposed method. The organization of this chapter is summarized as follows. In section 5.2, the formulation of system model is presented. Designing the backstepping controller is shown in section 5.3. The development of genetic adaptive fuzzy-neural control scheme is shown in section 5.4. An illustrative example is given in section 5.5 to show the feasibility and applicability of the proposed method. Conclusions are drawn in section 5.6.

5.2 Formulation of System Model

Consider an *n*-link planar manipulator with *n* inputs and *n* outputs shown in Fig.5-1. The dynamic equations are given by

$$\begin{aligned} &\mathbf{H(q)}\ddot{\mathbf{q}} + \mathbf{C(q,\dot{q})}\dot{\mathbf{q}} + \mathbf{g(q)} = \mathbf{u} \\ &\mathbf{y} = \mathbf{q} \end{aligned} \tag{5-1}$$

where $\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}} \in \Re^n$ denote the link position, velocity, and acceleration vectors, respectively. $\mathbf{H(q)} \in \Re^{n\times n}$ is the inertia matrix, $\mathbf{C(q,\dot{q})} \in \Re^{n\times n}$ is the centripetal-Coriolis matrix, $\mathbf{g(q)} \in \Re^n$ is the vector of gravitational, $\mathbf{u} \in \Re^n$ is the input vector of applied joint torques, $\mathbf{y} \in \Re^n$ represents output vector. The reference vector is define as $\mathbf{y}_d = [\mathbf{y}_d, \dot{\mathbf{y}}_d, \ldots, \mathbf{y}_d^{(n_p-1)}]$. Our control objective is to develop the backstepping controller so that the output trajectory $\mathbf{y}$ can asymptotically track a bounded command $\mathbf{y_d}$. Define $\mathbf{x}_1 = \mathbf{q}, \mathbf{x}_2 = \dot{\mathbf{q}}$. And then the formulation (5-1) can be denoted by (5-2)

$$\begin{aligned} &\dot{\mathbf{x}}_1 = \mathbf{x}_2 \\ &\dot{\mathbf{x}}_2 = \mathbf{H}^{-1}(\mathbf{x}_1)[\mathbf{u} - \mathbf{C}(\mathbf{x}_1, \mathbf{x}_2)\mathbf{x}_2 - \mathbf{g}(\mathbf{x}_1)] \\ &\mathbf{y} = \mathbf{x}_1 \end{aligned} \tag{5-2}$$

where $\mathbf{H}^{-1}(\mathbf{x}_1)$ and $\mathbf{C}(\mathbf{x}_1, \mathbf{x}_2)$ are unknown nonlinear smooth function. The dynamics (5-2) has the following properties:

Property 1: The inertia matrix $\mathbf{H(q)}$ is symmetric and positive definite, and there exist $\sigma_0 > 0, \sigma_0 > \Re$ such that $\mathbf{H(q)} \geq \sigma_0 \mathbf{I}$.

Property 2: The inertia matrix $\mathbf{H(q)}$ and centripetal-Coriolis matrix $\mathbf{C(q,\dot{q})}$ have the following property

$$\dot{\mathbf{q}}^T(\dot{\mathbf{H}} - 2\mathbf{C})\dot{\mathbf{q}} = 0 \tag{5-3}$$

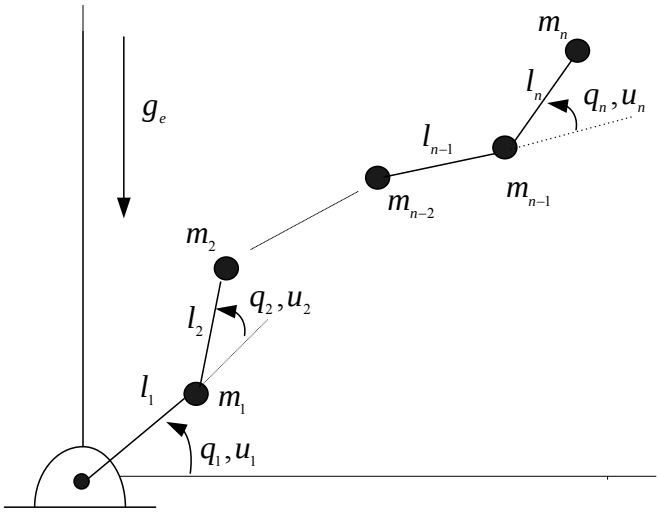

Fig.5-1. *n*-link planar manipulator.

5.3 Backstepping Control Design

The detail design procedure of the backstepping controller is described as follows.

Step 1) Define a tracking error as

$$\mathbf{z}_1 = \mathbf{x}_1 - \mathbf{y}_d \tag{5-4}$$

Then, differentiating $\mathbf{z}_1$ can be expressed as

$$\dot{\mathbf{z}}_1 = \dot{\mathbf{x}}_1 - \dot{\mathbf{y}}_d \tag{5-5}$$

Define a virtual control as

$$\boldsymbol{\alpha}_1 = \dot{\mathbf{y}}_d - c_1\mathbf{z}_1 \tag{5-6}$$

where $c_1 > 0$ is design scalar parameters. From (5-5) and (5-6), if $\boldsymbol{\alpha}_1 = \dot{\mathbf{x}}_1$, then $\lim_{t\to\infty}\mathbf{z}_1 \to 0$, that is, the state trajectory $\mathbf{x}_1$ can asymptotically track the bounded command $\mathbf{y}_d$. Thus, define an error state as $\mathbf{z}_2 = \dot{\mathbf{x}}_1 - \boldsymbol{\alpha}_1 = \mathbf{x}_2 - \boldsymbol{\alpha}_1$. Then, our next goal is to force the error state $\mathbf{z}_2$ to decay to zero. By using

(5-6) and the fact that $\dot{\mathbf{x}}_1 = \mathbf{z}_2 + \boldsymbol{\alpha}_1$, equation (5-5) can be rewritten as

$$\dot{\mathbf{z}}_1 = \mathbf{z}_2 - c_1\mathbf{z}_1 \tag{5-7}$$

Consider the Lyapunov function of the first subsystem as follows

$$V_1 = \frac{1}{2}\mathbf{z}_1{}^T\mathbf{z}_1 \tag{5-8}$$

By differentiating (5-8) and using (5-7), we have

$$\begin{aligned} \dot{V}_1 &= \mathbf{z}_1{}^T\dot{\mathbf{z}}_1 = \mathbf{z}_1{}^T(\dot{\mathbf{x}}_1 - \dot{\mathbf{y}}_d) = \mathbf{z}_1{}^T(\dot{\mathbf{x}}_2 - \dot{\mathbf{y}}_d) \\ &= \mathbf{z}_1{}^T(\mathbf{z}_2 + \boldsymbol{\alpha}_1 - \dot{\mathbf{y}}_d) \\ &= -c_1\mathbf{z}_1{}^T\mathbf{z}_1 + \mathbf{z}_1{}^T\mathbf{z}_2 \end{aligned} \tag{5-9}$$

Step 2) Differentiating $\mathbf{z}_2$ can be expressed as

$$\begin{aligned} \dot{\mathbf{z}}_2 &= \dot{\mathbf{x}}_2 - \dot{\boldsymbol{\alpha}}_1 \\ &= \mathbf{H}^{-1}(-\mathbf{C}\mathbf{x}_2 - G + \mathbf{u}) - \dot{\boldsymbol{\alpha}}_1 \end{aligned} \tag{5-10}$$

Define a control law as

$$\mathbf{u} = -c_2\mathbf{z}_2 - \mathbf{z}_1 - \mathbf{f} \tag{5-11}$$

where $\mathbf{f} \in \Re^n$ will be designed later, $c_2 > 0$ is design scalar parameters.

Consider the Lyapunov function of the second subsystem as follows

$$V_2 = \frac{1}{2}\mathbf{z}_2{}^T\mathbf{H}\mathbf{z}_2 \tag{5-12}$$

By differentiating (5-12), we have

$$\begin{aligned} \dot{V}_2 &= \frac{1}{2}\mathbf{z}_2{}^T\mathbf{H}\dot{\mathbf{z}}_2 + \frac{1}{2}\dot{\mathbf{z}}_2{}^T\mathbf{H}\mathbf{z}_2 + \frac{1}{2}\mathbf{z}_2{}^T\dot{\mathbf{H}}\mathbf{z}_2 \\ &= \mathbf{z}_2{}^T\mathbf{H}\dot{\mathbf{z}}_2 + \frac{1}{2}\mathbf{z}_2{}^T\dot{\mathbf{H}}\mathbf{z}_2 \\ &= \mathbf{z}_2{}^T(-\mathbf{C}\mathbf{x}_2 - \mathbf{g} + \mathbf{u} - \mathbf{H}\dot{\boldsymbol{\alpha}}_1) + \mathbf{z}_2{}^T\mathbf{C}\mathbf{z}_2 \\ &= \mathbf{z}_2{}^T(-\mathbf{C}\mathbf{x}_2 + \mathbf{C}\mathbf{z}_2 - \mathbf{g} - \mathbf{H}\dot{\boldsymbol{\alpha}}_1 + \mathbf{u}) \\ &= \mathbf{z}_2{}^T(-\mathbf{C}\boldsymbol{\alpha}_1 - \mathbf{g} - \mathbf{H}\dot{\boldsymbol{\alpha}}_1 + \mathbf{u}) \\ &= \mathbf{z}_2{}^T(\mathbf{f} + \mathbf{u}) \end{aligned} \tag{5-13}$$

where $\mathbf{f} = -\mathbf{C}\boldsymbol{\alpha}_1 - \mathbf{g} - \mathbf{H}\dot{\boldsymbol{\alpha}}_1$, by using (5-11), it follows that

$$\dot{V}_2 = \mathbf{z}_2{}^T(-c_2\mathbf{z}_2 - \mathbf{z}_1) \tag{5-14}$$

Step 3) Select Lyapunov function as

$$V = V_1 + V_2 \tag{5-15}$$

By differentiating (5-15) and using (5-9), (5-14), we have

$$\begin{aligned} \dot{V} &= \dot{V}_1 + \dot{V}_2 \\ &= -c_1\mathbf{z}_1{}^T\mathbf{z}_1 - c_2\mathbf{z}_2{}^T\mathbf{z}_2 \\ &\le -c_1\mathbf{z}_1{}^T\mathbf{z}_1 \end{aligned} \tag{5-16}$$

From (5-15) and (5-16), we can conclude that $\mathbf{z}_1$ is bounded. Moreover, from (5-7), $\mathbf{z}_1$ is also bounded. Integrating (5-16) can be expressed as

$$c_1\int_0^\infty \mathbf{z}_1{}^T(\tau)\mathbf{z}_1(\tau)d\tau \le -V(\infty) + V(0) \tag{5-17}$$

Because of the fact that the right side of (5-17) is bounded, we have $\mathbf{z}_1 \in L_2$. According to the Barbalat's Lemma, $\lim_{t\to\infty}\mathbf{z}_1 = 0$, that is, the state trajectory $\mathbf{x}_1$ can asymptotically track the bounded command $\mathbf{y}_d$.

5.4 Development of Genetic Adaptive Fuzzy-neural Control Scheme for Manipulators

In this section, based on the compact genetic algorithm as mentioned above, a method for developing on-line backstepping adaptive fuzzy-neural controllers is proposed.

Since $\mathbf{f}$ is uncertain, the optimal control law (5-11) cannot be obtained. To solve this problem, the fuzzy-neural system is used to approximate the uncertain continuous function $\mathbf{f}$. First, the uncertain continuous function $\mathbf{f}$ in

(5-11) is replaced by fuzzy-neural networks (2-2), i.e., $\hat{\mathbf{f}}$. The resulting control law is

$$\mathbf{u}_c = -c_2\mathbf{z}_2 - \mathbf{z}_1 - \hat{\mathbf{f}} \tag{5-18}$$

Suppose that $\mathbf{u} = \mathbf{u}_c$. Then, substituting (5-18) in (5-13), we have

$$\begin{aligned} \dot{V} &= -c_1\mathbf{z}_1{}^T\mathbf{z}_1 + \mathbf{z}_1{}^T\mathbf{z}_2 + \mathbf{z}_2{}^T(\mathbf{f} - c_2\mathbf{z}_2 - \mathbf{z}_1 - \hat{\mathbf{f}}) \\ &= -c_1\mathbf{z}_1{}^T\mathbf{z}_1 - c_1\mathbf{z}_2{}^T\mathbf{z}_2 + \mathbf{z}_2{}^T(\mathbf{f} - \hat{\mathbf{f}}) \\ &\le -c_1\mathbf{z}_1{}^T\mathbf{z}_1 - c_2\mathbf{z}_2{}^T\mathbf{z}_2 + \left|\mathbf{z}_2{}^T\right|\mathbf{f}^U - \mathbf{z}_2{}^T\hat{\mathbf{f}} \end{aligned} \tag{5-19}$$

where $|\mathbf{f}| \le \mathbf{f}^U < \infty$. By using the fact that the states must move in the direction of smaller values of V if $\dot{V} < 0$, the energy fitness function $\mathbf{z}_2{}^T\hat{\mathbf{f}}$ for the system is defined in order to instantaneously evaluate the stability of the closed-loop system. A chromosome with the largest energy fitness function denotes the optimal solution. So, a better chromosome can be obtained according to the energy fitness function.

Since the system states may go into the unsafe region if the genetic operations can not instantaneously generate the appropriate weightings of the fuzzy-neural networks in some time interval, the concept of the safe controller is incorporated into the genetic adaptive backstepping fuzzy-neural controller to guarantee that the system states are confined to the safe region. By incorporating a safe control term $\mathbf{u}_s$ into $\mathbf{u}_c$, the control law becomes

$$\mathbf{u} = \mathbf{u}_c + \mathbf{u}_s \tag{5-20}$$

The safe control term $\mathbf{u}_s$ is added when the function V is greater than a positive limit V^u. If $V \le V^u$, then the safe control term $\mathbf{u}_s$ is canceled. That is, if the system tends to enter the unsafe region ($V > V^u$), then $\mathbf{u}_s$ forces the

system to return to the safe region.

Substituting (5-20) into (5-13), we have

$$\begin{aligned}\dot{V} &= -c_1\mathbf{z}_1{}^T\mathbf{z}_1 - c_2\mathbf{z}_2{}^T\mathbf{z}_2 + \mathbf{z}_2{}^T(\mathbf{f} - \hat{\mathbf{f}} + \mathbf{u}_s) \\ &\le -c_1\mathbf{z}_1{}^T\mathbf{z}_1 - c_2\mathbf{z}_2{}^T\mathbf{z}_2 + \left|\mathbf{z}_2{}^T\right|(|\mathbf{f}| - |\hat{\mathbf{f}}|) + \mathbf{z}_2{}^T\mathbf{u}_s\end{aligned} \tag{5-21}$$

Suppose that the safe control term $\mathbf{u}_s$ is given as

$$\mathbf{u}_s = \begin{cases} -\text{sgn}(\mathbf{z}_2)[\mathbf{f}^U(\mathbf{x}) + |\hat{\mathbf{f}}|] & \text{if } V > V^u \\ 0 & \text{if } V \le V^u \end{cases} \tag{5-22}$$

where V^u is a design parameter and $\text{sgn}(\mathbf{z}_2) = 1(-1)$ if $\mathbf{z}_2 \ge 0(< 0)$. Suppose that $V > V^u$. Then, substituting (5-22) into (5-21), we have

$$\begin{aligned}\dot{V} &\le -c_1\mathbf{z}_1{}^T\mathbf{z}_1 - c_2\mathbf{z}_2{}^T\mathbf{z}_2 + \left|\mathbf{z}_2{}^T\right|(|\mathbf{f}| - |\hat{\mathbf{f}}| - \mathbf{f}^U(\mathbf{x}) + |\hat{\mathbf{f}}|) \\ &\le -c_1\mathbf{z}_1{}^T\mathbf{z}_1 - c_2\mathbf{z}_2{}^T\mathbf{z}_2 \le 0\end{aligned} \tag{5-23}$$

From (5-23), the bounded stability of the closed-loop system for the nonlinear system in (5-2) can be guaranteed.

The design algorithm of proposed method is as follows.

Design algorithm:

Step 1: Construct the fuzzy-neural networks for $\mathbf{f}(\mathbf{x}, \mathbf{y}_d \mid \mathbf{w})$ including fuzzy sets for $\mathbf{x}$, $\mathbf{y}_d$ and the weighting vectors $\mathbf{w}$.

Step 2: Adjust the weighting vectors by using the compact genetic algorithm (4-21)-(4-27) with the energy fitness function $\mathbf{z}_2{}^T\hat{\mathbf{f}}$.

Step 3: Compute $\hat{\mathbf{f}}(\mathbf{x}, \mathbf{y}_d \mid \mathbf{w})$ according to (2-2). Then, obtain the control law (5-18).

5.5 Simulation Results

This section presents the simulation results of the proposed genetic

adaptive fuzzy-neural controller for a two-link planar manipulator, then to illustrate that the stability of the closed-loop system is guaranteed, and all signals involved are bounded.

The system can be denoted by (5-1) with $\mathbf{q} = [q_1 \; q_2]^T$, $\mathbf{u} = [u_1 \; u_2]^T$,

$$\mathbf{H(q)} = \begin{bmatrix} (m_1 + m_2)l_1^2 + m_2 l_2^2 + 2m_2 l_1 l_2 \cos q_2 & m_2 l_2^2 + m_2 l_1 l_2 \cos q_2 \\ m_2 l_2^2 + m_2 l_1 l_2 \cos q_2 & m_2 l_2^2 \end{bmatrix},$$

$$\mathbf{C(q,\dot{q})} = \begin{bmatrix} -m_2 l_1 l_2 \dot{q}_2 \sin q_2 & -m_2 l_1 l_2 (\dot{q}_1 + \dot{q}_2) \sin q_2 \\ m_2 l_1 l_2 \dot{q}_1 \sin q_2 & 0 \end{bmatrix} \text{ and}$$

$$\mathbf{g(q)} = \begin{bmatrix} (m_1 + m_2) l_1 g_e \cos q_1 + m_2 l_2 g_e \cos(q_1 + q_2) \\ m_2 l_2 g_e \cos(q_1 + q_2) \end{bmatrix}.$$

The definition of variables involved is given as follows m_1: mass of link L1, m_2: mass of link L2, l_1: length of link L1, l_2 :length of link L2, q_1 :angle of link L1, q_2 :angle of link L2.

The parameter values are $m_1 = 0.5\text{kg}$, $m_2 = 0.5\text{kg}$, $l_1 = 1\text{m}$, $l_2 = 0.8\text{m}$ and $g_e = 9.8\,\text{m/sec}$. The design parameters of the genetic algorithm are given as k=4, $p_m = 0.06$, and β=4. The adjustable parameters $\mathbf{w}$ of $\hat{\mathbf{f}}(\mathbf{x}, \mathbf{y}_d)$ are in the intervals $\mathbf{D}$ =[-1,1]. The initial states are set as $\mathbf{x}(0) = [0.1, 0.2, 0, 0.2]$ (case 1) and $\mathbf{x}(0) = [0.1, 0.1, -0.1, 0.1]$ (case 2). The design parameters are selected as $c_1 = [120 \; 15]$, $c_2 = [100 \; 10]$, $V^u = 1$(case 1) and $c_1 = [125 \; 15]$, $c_2 = [100 \; 10]$, $V^u = 1$(case 2), The membership functions for $\mathbf{x}_i$, $i = 1,2$ are given as $\mu_{A_i^1}(\mathbf{x}_i) = \exp[-0.5((\mathbf{x}_i + 1.25)/0.6)^2]$, $\mu_{A_i^2}(\mathbf{x}_i) = \exp[-0.5((\mathbf{x}_i)/0.6)^2]$, $\mu_{A_i^1}(\mathbf{x}_i) = \exp[-0.5((\mathbf{x}_i - 1.25)/0.6)^2]$. To apply

the proposed method to the system, the bounds $\mathbf{f}^U \in \Re^2$ should be obtained as

$$\left|\mathbf{f}\right| = \left|-\mathbf{C}\boldsymbol{\alpha}_1 - \mathbf{g} - \mathbf{H}\dot{\boldsymbol{\alpha}}_1\right| \equiv \mathbf{f}^U \tag{5-24}$$

The first case is to control the outputs q_1 and q_2 of the two-link planar manipulator to track the desired trajectories $y_{d1} = (\pi/30)\sin(t)$ and $y_{d2} = (\pi/30)\cos(t)$, respectively. The second case is to control the outputs q_1 and q_2 to track the desired trajectories $y_{d1} = 0.4 - 0.4e^{-t}$ and $y_{d2} = 0.5 - 0.5e^{-t}$, respectively. Then the m_1 and m_2 are increased to $m_1 = 1kg$ and $m_2 = 1kg$ at the time $t = 5\text{sec}$, respectively. The simulation results are shown in Figs. 5-2, 5-3, 5-4, 5-5, 5-6, 5-8 and 5-9. In case 1, Fig. 5-2 shows the steady trajectories of the first output q_1 and the reference output y_{d1}, Fig. 5-3 shows the steady trajectories of the first output q_2 and the reference output y_{d2}. Fig. 5-4 shows the control inputs u_1 and u_2. Fig. 5-5 shows the tracking errors $q_1 - y_{d1}$ and $q_2 - y_{d2}$. In case 2, Fig. 5-6 shows the steady trajectories of the first output q_1 and the reference output y_{d1}, and Fig. 5-7 shows the steady trajectories of the first output q_2 and the reference output y_{d2}. Fig. 5-8 shows the control inputs u_1 and u_2. Fig. 5-9 shows the tracking errors $q_1 - y_{d1}$ and $q_2 - y_{d2}$.The simulation results demonstrate that the trajectories of the actual outputs q_1 and q_2 follow the reference signals y_{d1} and y_{d2}, respectively, very well.

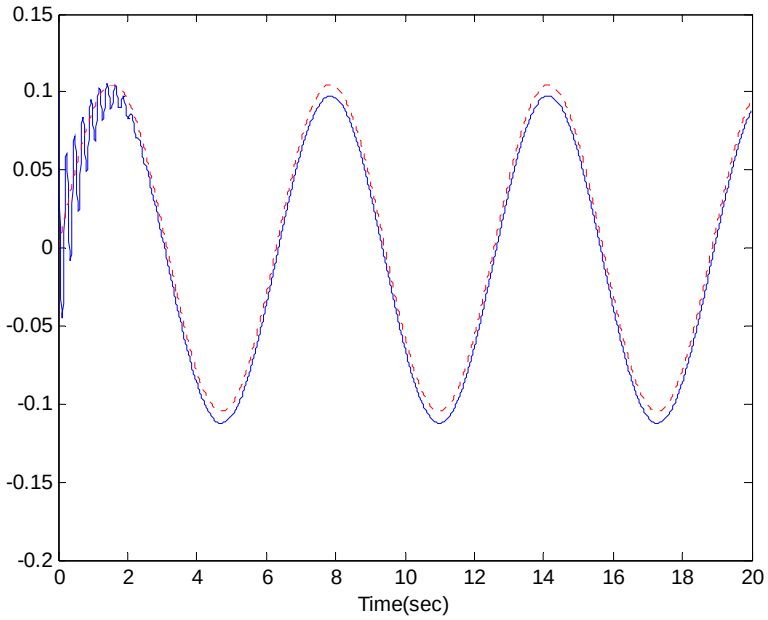

Fig. 5-2. The first output q_1 (solid line) and bounded reference y_{d1} (dashed line) (case 1).

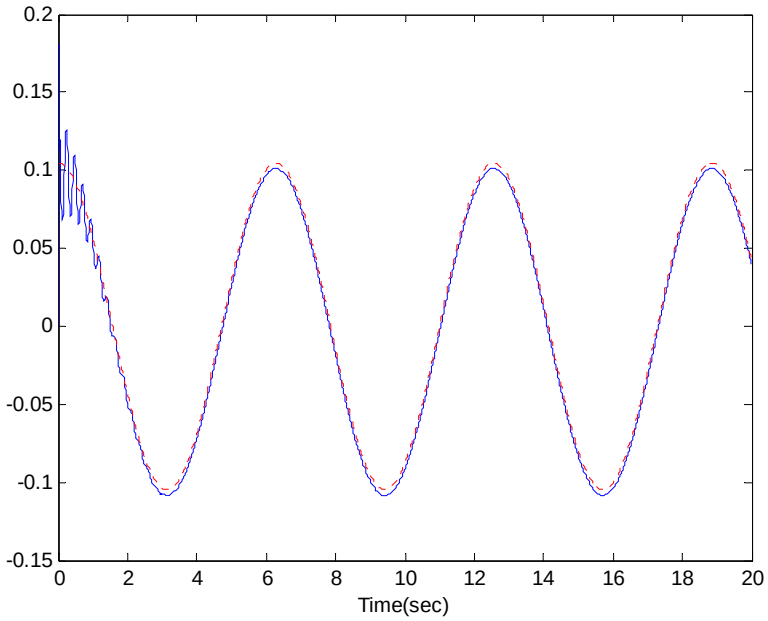

Fig. 5-3. The second output q_2 (solid line) and bounded reference y_{d2} (dashed line)(case 1).

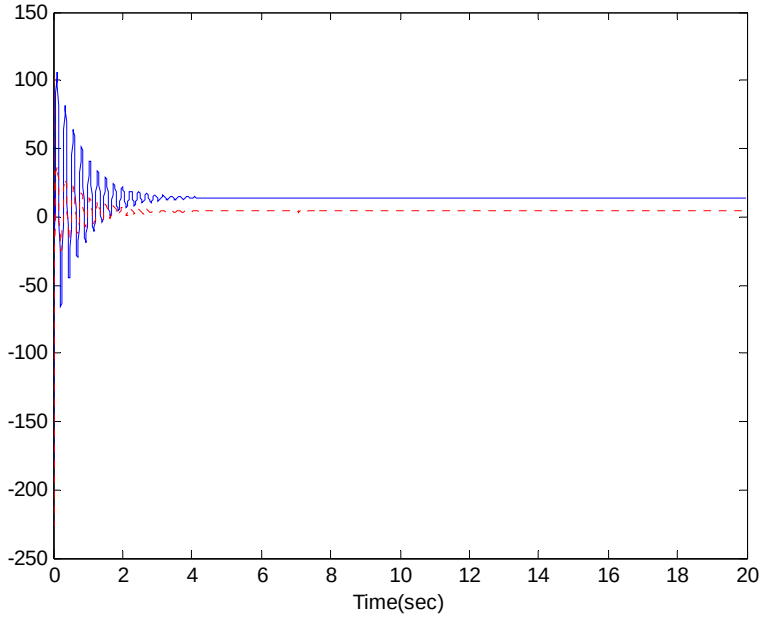

Fig. 5-4. The control inputs u_1 (solid line) and u_2 (dashed line)(case 1).

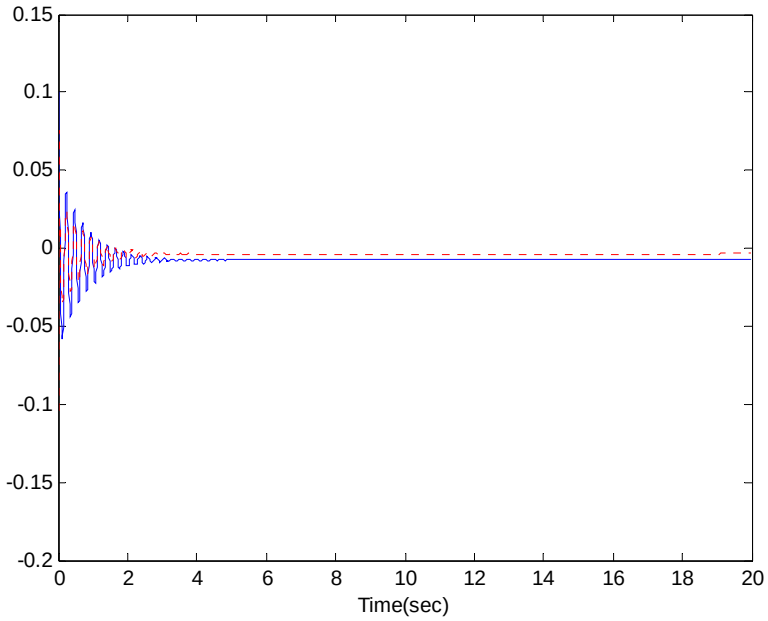

Fig. 5-5. The tracking errors $q_1 - y_{d1}$ (solid line) and

$q_2 - y_{d2}$ (dashed line)(case 1).

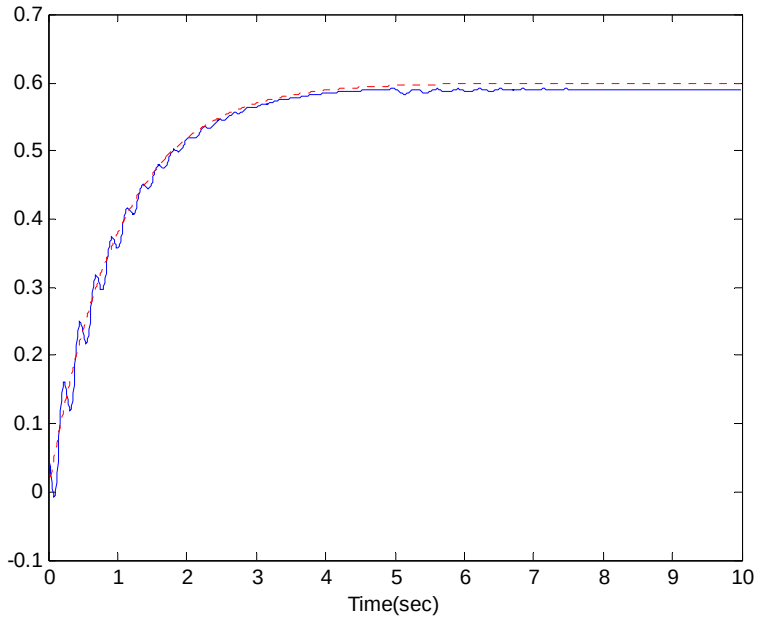

Fig. 5-6. The first output q_1 (solid line) and bounded reference y_{d1} (dashed line) (case 2).

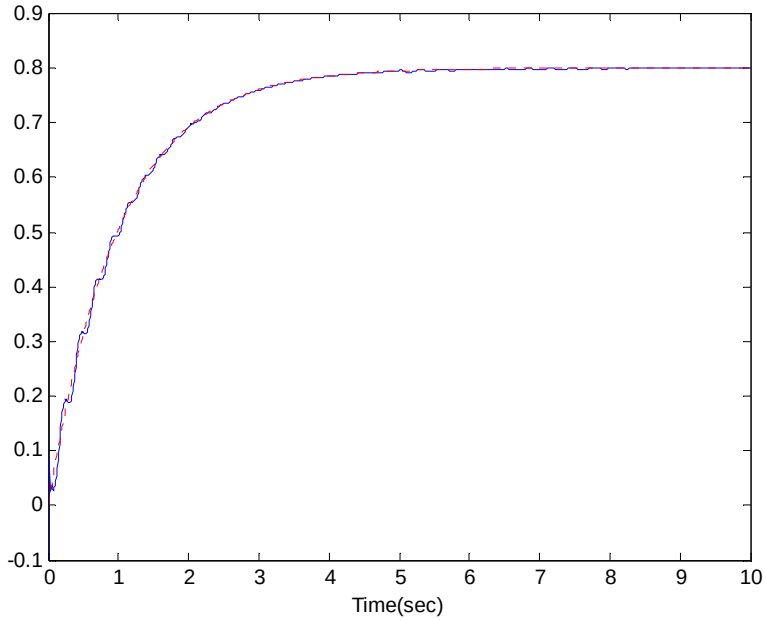

Fig. 5-7. The second output q_2 (solid line) and bounded reference y_{d2} (dashed line)(case 2).

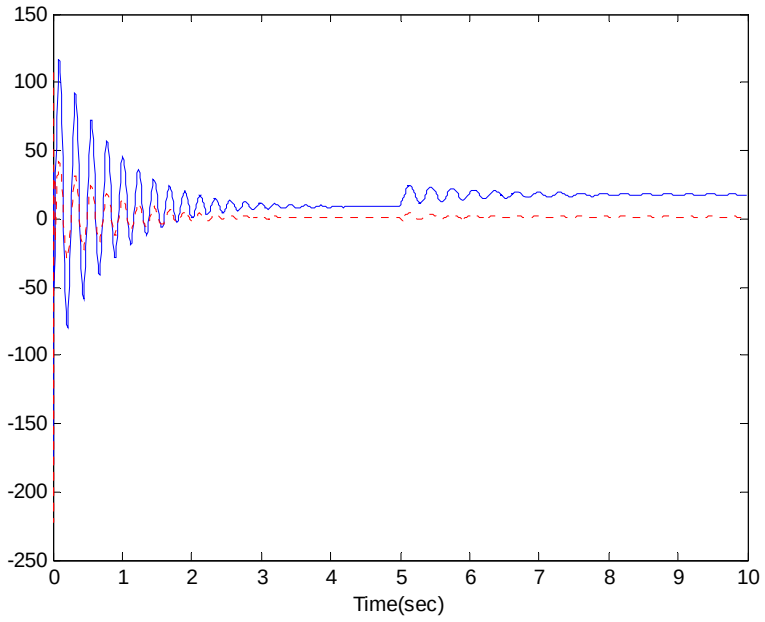

Fig. 5-8. The control inputs u_1 (solid line) and u_2 (dashed line)(case 2).

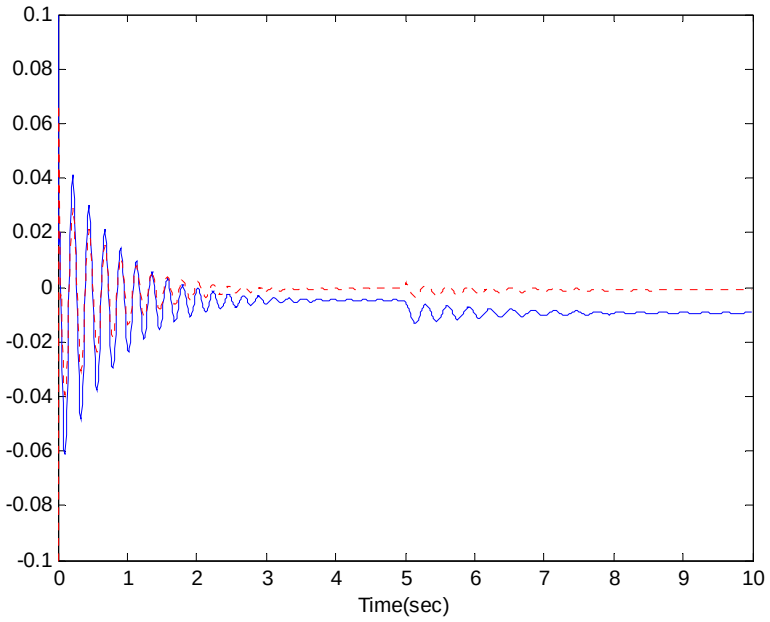

Fig. 5-9. The tracking errors $q_1 - y_{d1}$ (solid line) and $q_2 - y_{d2}$ (dashed line)(case 2).

5.6 Conclusions

The genetic adaptive fuzzy-neural control scheme for a class of MIMO robotic systems has been developed. The preliminary off-line tuning of the weighting factors of the fuzzy controller is not required. The free parameters of the genetic adaptive backstepping fuzzy-neural controller can be successfully tuned on-line via the compact genetic algorithm with an energy evaluation mechanism. Using the energy evaluation mechanism can evaluate the real-time stability of the closed-loop systems in order to generate the appropriate control strategy. In addition, the safe control term added into the genetic adaptive backstepping fuzzy-neural networks guarantees the safeness of the closed-loop system. The applicability of the proposed control scheme is demonstrated by the simulation results.

Chapter 6

Design of Fuzzy-neural Controllers for DC Servo motors Using Compact Genetic Algorithm

6.1 Overview

In this chapter, to demonstrate the propose method on practical system, the compact genetic adaptive fuzzy-neural controller discussed in Chapter 4 is used to control a DC servo motor. The experimental platform is MT22R2-24 DC servo motor, and uses the back DC-DC converter to perform DC servo motor drive applications. Finally, simulation and experimental results demonstrate the applicability and effectiveness of the proposed method. The organization of this chapter is summarized as follows. Section 6.2 mainly introduces dynamic system of a DC servo motor, and our control objective. Section 6.3 shows the computer simulation results of propose method for a DC servomotor. Section 6.4 introduces the servo motor experimental platform and the hardware framework, and shows the experimental results to illustrate the feasibility and applicability of the proposed method. Conclusions are given in section 6.5.

6.2 Problem Description of DC Servo motor

The equivalent model of a DC servo motor is shown in Fig. 6.1.

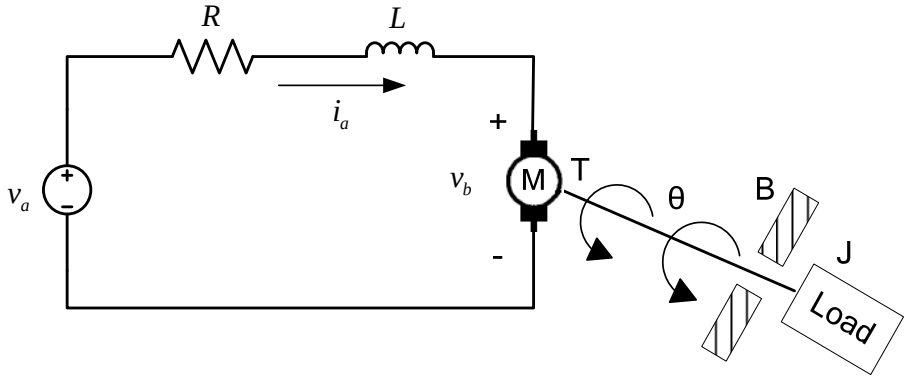

Fig.6-1. DC servo motor with load.

where R is the armature resistance, L is the armature inductance, θ is the angular displacement of the motor shaft, B is the friction constant, J is the armature moment of inertia of the motor, and T is the load torque.

From the motor circuit shown in Fig. 6-1, the differential equation for the armature circuit is

$$v_a = Ri_a + L\frac{di_a}{dt} + v_b \tag{6-1}$$

where i_a denotes the armature current. The voltage v_b is the back electromotive force of the motor and is proportional to the speed of the motor shaft. It is given by

$$v_b = K_b\frac{d\theta}{dt} \tag{6-2}$$

where K_b is back-emf constant. For a DC servo motor, the torque generated by the motor is proportional to the armature current. That is

$$T = K_t i_a \tag{6-3}$$

where K_t is torque constant. We also know that

$$T = J\frac{d^2\theta}{dt^2} + B\frac{d\theta}{dt} \tag{6-4}$$

Let $\mathbf{x}(t)=\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}=\begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \\ i_a(t) \end{bmatrix}$. Then, substituting (6-3) in (6-4), and according to (6-1), (6-2), (6-4), the representation of the state space is given as

$$\begin{bmatrix} \dot{\theta}(t) \\ \ddot{\theta}(t) \\ \dot{i}_a(t) \end{bmatrix}=\begin{bmatrix} 0 & 1 & 0 \\ 0 & -B/J & -K_t/J \\ 0 & -K_b/L & -R/L \end{bmatrix}\begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \\ i_a(t) \end{bmatrix}+\begin{bmatrix} 0 \\ 0 \\ 1/L \end{bmatrix}v_a$$
$$y(t)=\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \\ i_a(t) \end{bmatrix} \tag{6-5}$$

Suppose that L =0. Then, equation (6-1) can be rewritten as

$$v_a = Ri_a + v_b \tag{6-6}$$

Let $\mathbf{x}(t)=\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}=\begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \end{bmatrix}$. According to (6-2), (6-4) and (6-6), equation (6-5) can be written as:

$$\dot{\mathbf{x}}(t)=\begin{bmatrix} 0 & 1 \\ 0 & -\left(K_tK_b+RB\right)/JR \end{bmatrix}\mathbf{x(t)}+\begin{bmatrix} 0 \\ K_t/JR \end{bmatrix}v_a$$
$$y(t)=\begin{bmatrix} 1 & 0 \end{bmatrix}\mathbf{x}(t) \tag{6-7}$$

Our control objective is to develop the genetic adaptive backstepping fuzzy-neural controller so that the DC servo motor system output can asymptotically track a bounded command.

6.3 Computer Simulation Results

In this section, we present the computer simulation results of the compact genetic algorithm fuzzy-neural controller for a DC servomotor, and then illustrate that the stability of the closed-loop system is guaranteed.

According to section 6.2, consider the servo motor dynamics system

described as

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= -(\frac{1}{\tau})x_2 + (\frac{K_m}{\tau})u = f(x_1, x_2) + bu\end{aligned} \tag{6-8}$$

where $x_1(t) = \theta(t)$, $x_2(t) = \dot{\theta}(t)$, $K_m = \dfrac{K_t}{K_t K_b + BR}$, $\tau = \dfrac{JR}{K_t K_b + BR}$ and $u = v_a$. Clearly, (6-8) is in the form of chapter 4 for SISO system. Thus, the proposed method in chapter 4 is suitable to control the system. The control objective is to force the state x_1 of the servo motor to follow the reference signal y_d. The control law of proposed method is

$$u = u_c + u_s \tag{6-9}$$

where u_c is backstepping adaptive fuzzy-neural controller designed as

$$u_c = -\hat{\bar{f}} - c_2 z_2 - z_1 \tag{6-10}$$

and u_s is a safe control term designed as

$$u_s = \begin{cases} -\text{sgn}(z_2)[\bar{f}^U(\mathbf{x}) + \left|\hat{\bar{f}}\right|] & \text{if } V > V^u \\ 0 & \text{if } V \le V^u \end{cases} \tag{6-11}$$

The parameters of the servo motor are given as $K_t = 0.23\text{Nm/A}$, R=3.1180 Ω , $K_b = 0.23\text{volts}\cdot\text{sec/rad}$, J=0.0006 Kgm^2 and B=0.02 N-m-s /rad. The design parameters of the genetic algorithm are given as k=4, p_m=0.06, and $\beta = 4$. The adjustable parameters $\mathbf{w}$ of $\hat{f}$ are in the intervals D=[-5,5]. The reference signals are set as $y_d(t) = 60 - 100e^{-t}$ rad (case 1) and $y_d(t) = 70\sin(0.1t)$ rad (case 2). The membership functions for x_i, i=1,2 are

given as $\mu_{A_i^1}(x_i) = \frac{1}{(1+\exp(0.5\times(x_i+80)))}$, $\mu_{A_i^2}(x_i) = \exp(-0.001\times(x_i+60)^2)$,

$\mu_{A_i^3}(x_i) = \exp(-0.001\times(x_i+30)^2)$, $\mu_{A_i^4}(x_i) = \exp(-0.001\times(x_i)^2)$,

$\mu_{A_i^5}(x_i) = \exp(-0.001\times(x_i-30)^2)$, $\mu_{A_i^6}(x_i) = \exp(-0.001\times(x_i-60)^2)$,

$\mu_{A_i^7}(x_i) = \frac{1}{(1+\exp(-5\times(x_i-3)))}$.

To apply the proposed method to the system, the bounds f^U should be obtained as:

$$\begin{aligned} |f(x_1,x_2)| &= \left|-(\frac{1}{\tau})x_2\right| \le 62.0919x_2 \equiv f^U(x_1,x_2) \\ \overline{f}^U &= \frac{1}{b^L}(f^U + |\dot{\alpha}_1|) = \frac{1}{122}(62.0919x_2 + |c_1\dot{z}_1 + \ddot{y}_d|) \end{aligned} \tag{6-12}$$

The initial states are set as $\mathbf{x}(0) = [\text{-}1\,\text{rad}, 0]$ (case 1) and $\mathbf{x}(0) = [35\,\text{rad}, 0]$ (case 2). The design parameters are set as $c_1 = 30$, $c_2 = 10$, $V^u = 0.1$ (case 1) and $c_1 = 15$, $c_2 = 10$, $V^u = 0.1$ (case 2). The simulation results are shown in Fig. 6-2 to Fig. 6-7. In the case 1, from Fig. 6-2, it is shown that the system output $x_1(t)$ tracks the bounded $y_d(t)$ command well. Figs. 6-3 and 6-4 are the control input $u(t)$ and the tracking error $z_1(t)$, respectively. In the case 2, from Fig. 6-5, it is shown that the system output $y(t)$ also tracks the bounded $y_d(t)$ command well. Figs. 6-6 and 6-7 are the control input $u(t)$ and the tracking error $z_1(t)$, respectively.

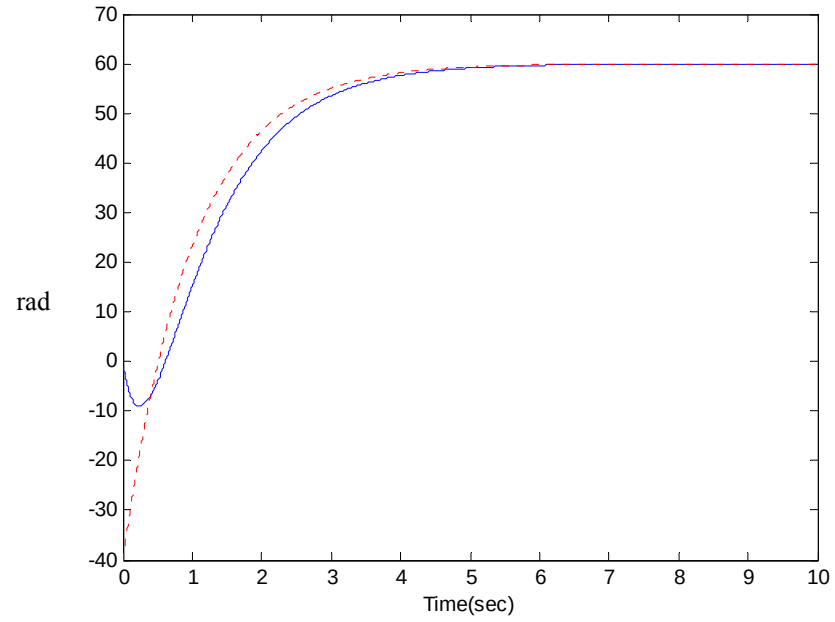

Fig. 6-2. The system output x_1 (solid line) and bounded reference y_d (dashed line) (case 1).

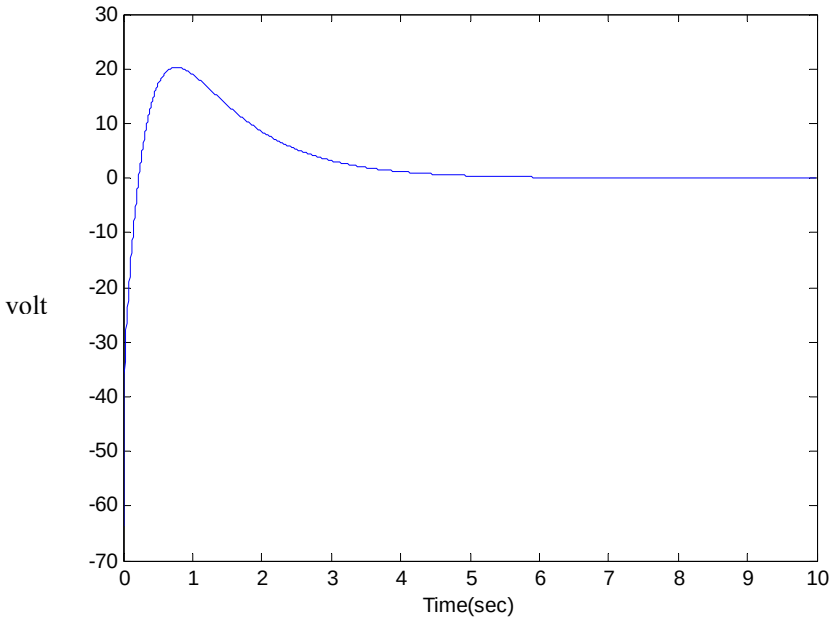

Fig. 6-3. The control input u (case 1).

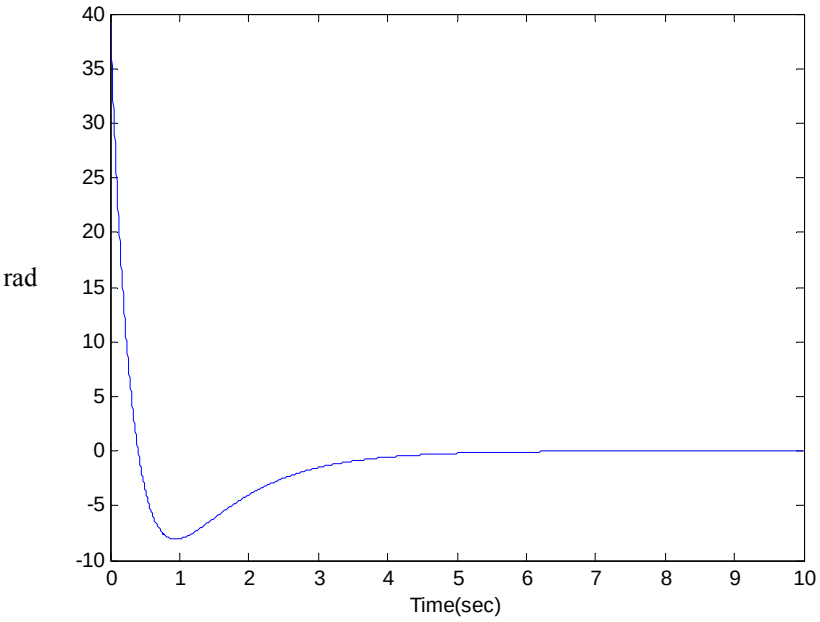

Fig. 6-4. The tracking error $z_1 = x_1 - y_d$ (case 1).

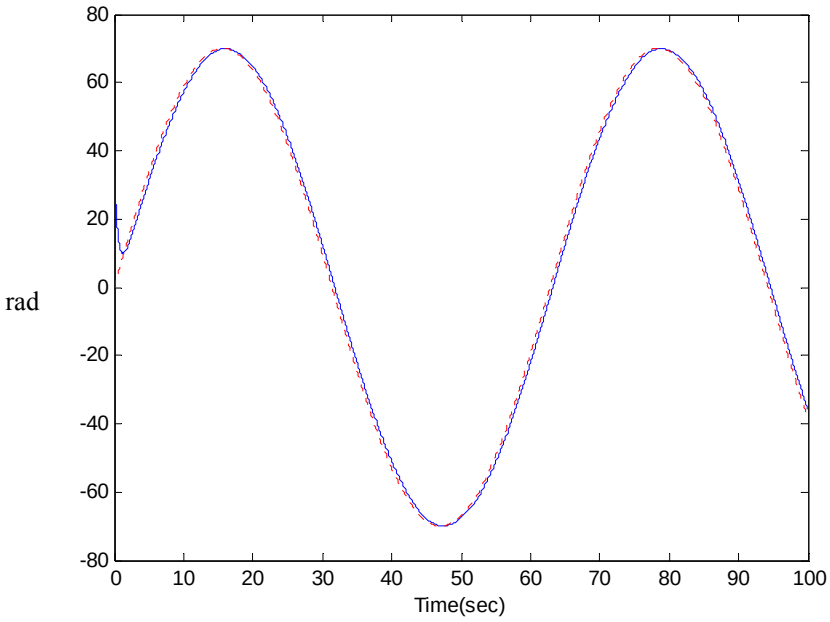

Fig. 6-5. The system output x_1 (solid line) and bounded reference y_d (dashed line) (case 2).

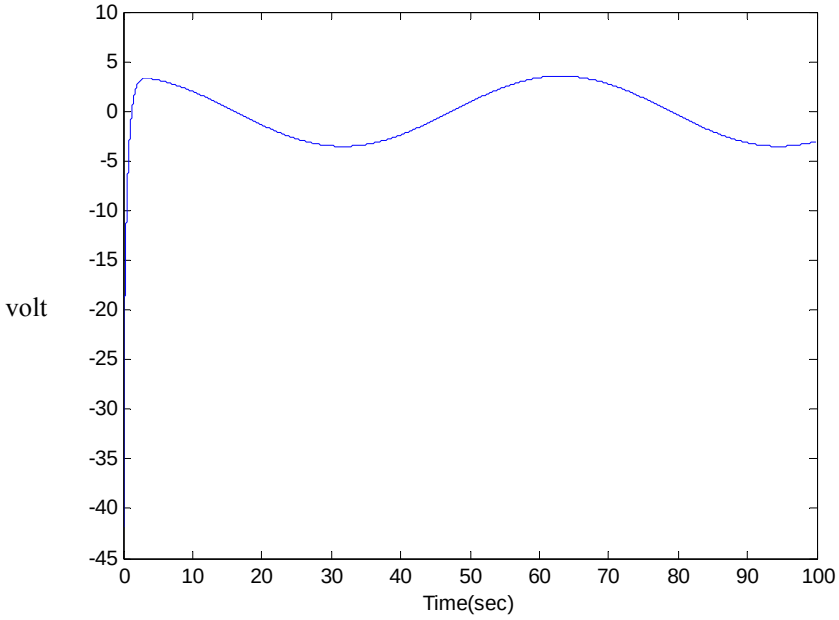

Fig. 6-6. The control input u (case 2).

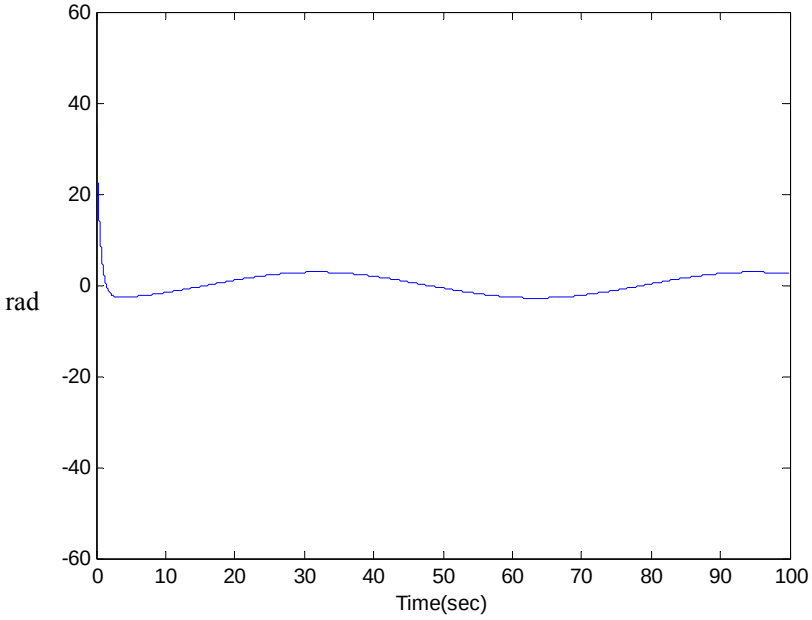

Fig. 6-7. The tracking error $z_1 = x_1 - y_d$ (case 2).

6.4 Introduction of Experiment

In this section, we will describe the DC servo motor experimental platform and hardware framework. In addition, we use the genetic adaptive

fuzzy-neural control scheme for tracking control of a DC servo motor.

6.4.1 Hardware Framework

The experimental platform MT22R2-24 DC servomotor system shown in Fig. 6-8 was made by SME Company. The maximum output voltage of control module is 10V and the maximum output voltage of motor is 75V (max. 3000RPM). The relational ratio is shown in Fig. 6-9.

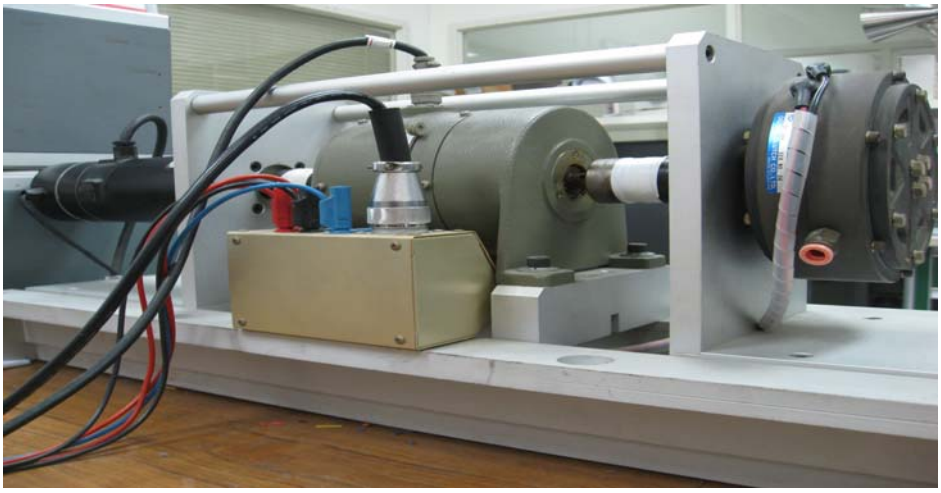

Fig. 6-8. MT22R2-24 DC servomotor.

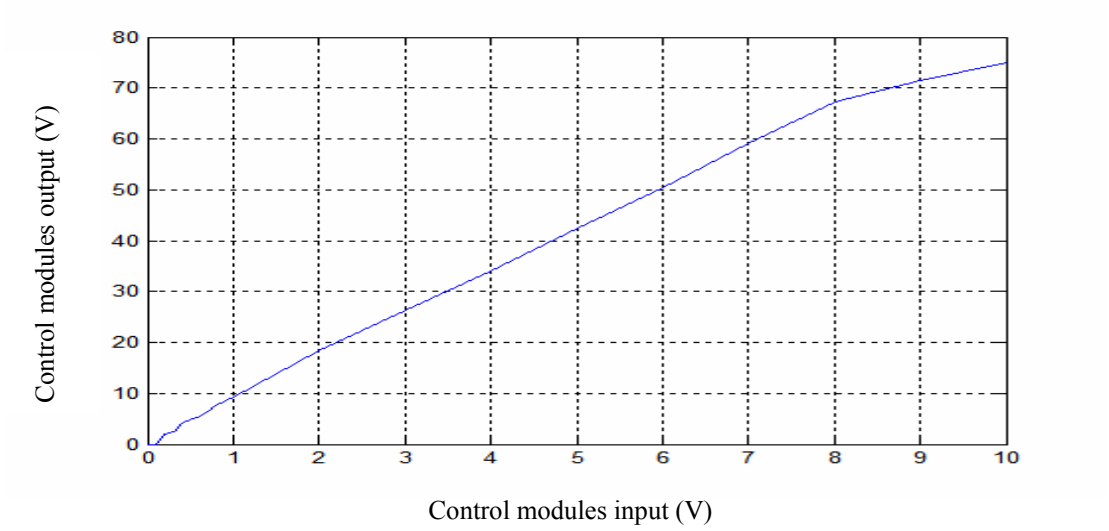

Fig.6-9. Relationship of the control modules input and output.

The specification of MT22R2-24 is shown in Table 6-1.

Table 6-1. Specification of MT22R2-24.

Physical quantity	Numerical value	Unit
Max. Voltage(V)	120	Volts
Max. Speed(RPM)	5000	RPM
Armature Moment of inertia(J)	0.0006	Kg-m^2
Torque Constant(K_t)	0.23	N-m/Amp
Voltage Constant(K_b)	0.23	Volts-sec/Rad
Resistance(R)	3.11809	Ohm
Peak Stall Torque	8.0	N-m
Acceleration at Peak Torque	13300	Rad/sec^2
Mechanical Time Constant	16	Milliseconds
B	0.00203	N-m-s/rad
Motor Weight	4.1	Kg

The block diagram of hardware implementation is shown in Fig. 6-10. The hardware structure is composed of a person computer, USB I/O 24 module, microcontroller (82G516), switching DC-DC converter and DC servo motor system. The proposed method is implemented by using the person computer with 5-ms sampling interval, and the servo driver motor system is controlled by Pulse-Width Modulation (PWM) method where switch-duty ratio $D \in [-1,1]$ is varied to adjust the output of the Buck DC-DC converter. The control input voltage u in PC needs to be transformed into the switch duty ratio D, and then the servo driver motor system is controlled by the output voltage of the

converter. The duty ratio of PWM is described as :

$$D = \begin{cases} 1 & ,if \quad u > 75 \\ \dfrac{u}{75} & ,if \quad -75 \le u \le 75 \\ -1 & ,if \quad u < -75 \end{cases} \tag{6-13}$$

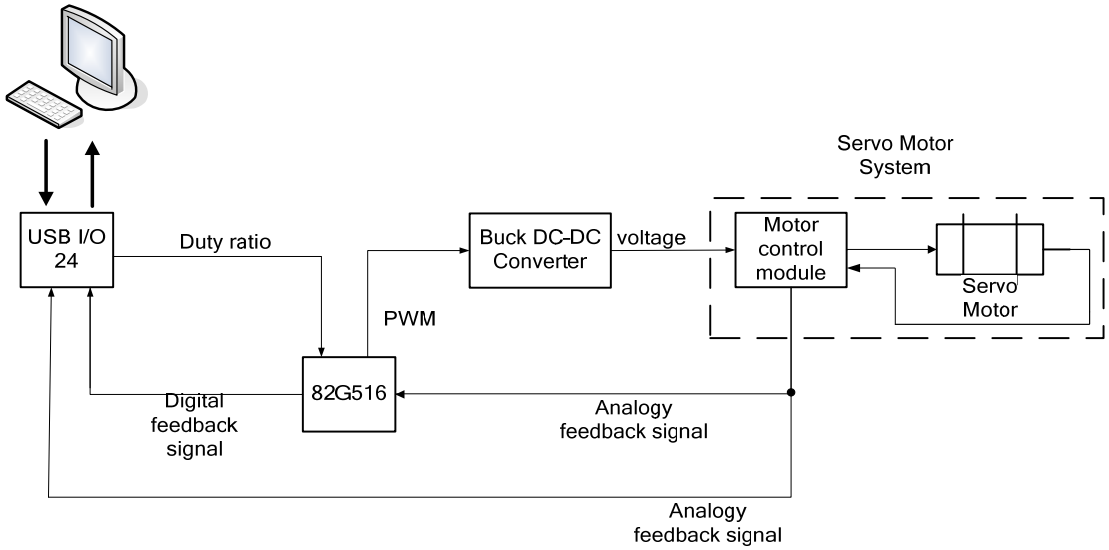

Fig.6-10. The block diagram of hardware implementation.

6.4.2 Experimental Results

The experimental control system block diagram is shown in Fig. 6-11. According to initial states, design parameters and reference signals, two experimental cases are given to test the performance of the controlled systems. The experimental results are shown in Figs. 6-12, 6-13, 6-14, 6-15, 6-16, 6-17, 6-18 and 6-19. In the cases 1, from Fig. 6-12, it is shown that the system output x_1 tracks the bounded y_d command well. The tracking error responses are shown in Fig. 6-13. And the associated switch duty ratio *D* and the associated control input voltage u are shown in Figs. 6-14 and 6-15, respectively. In the

cases 2, from Fig. 6-16, it is shown that the system output x_1 tracks the bounded y_d command well. The tracking error responses are shown in Fig. 6-17. And the associated switch duty ratio *D* and the associated control input voltage u are shown in Figs. 6-18 and 6-19, respectively. As shown in Fig. 6-15, the state x_1 of the servo motor can also track the reference signal well even though the reference signal is a sine wave.

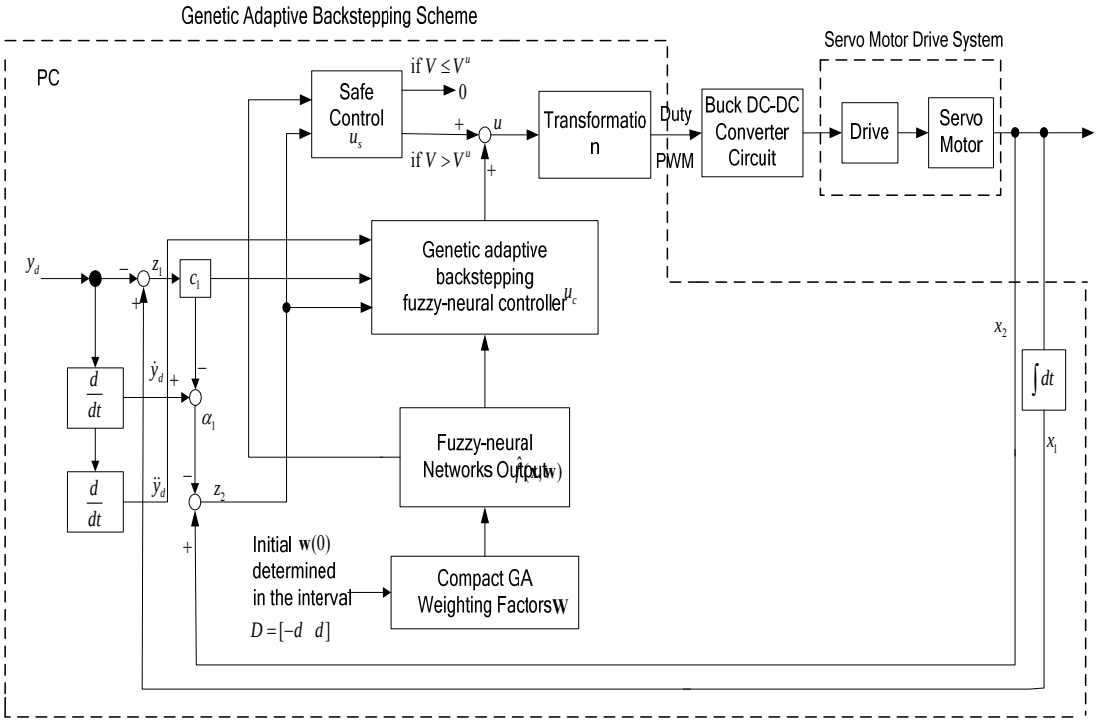

Fig. 6-11. Control system block diagram.

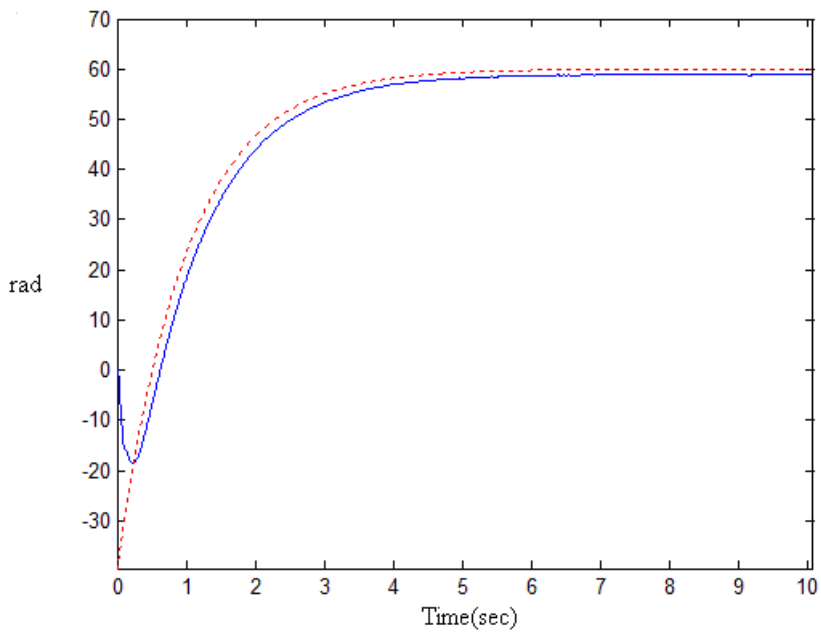

Fig. 6-12. The system output x_1 (solid line) and bounded reference y_d (dashed line) (case 1).

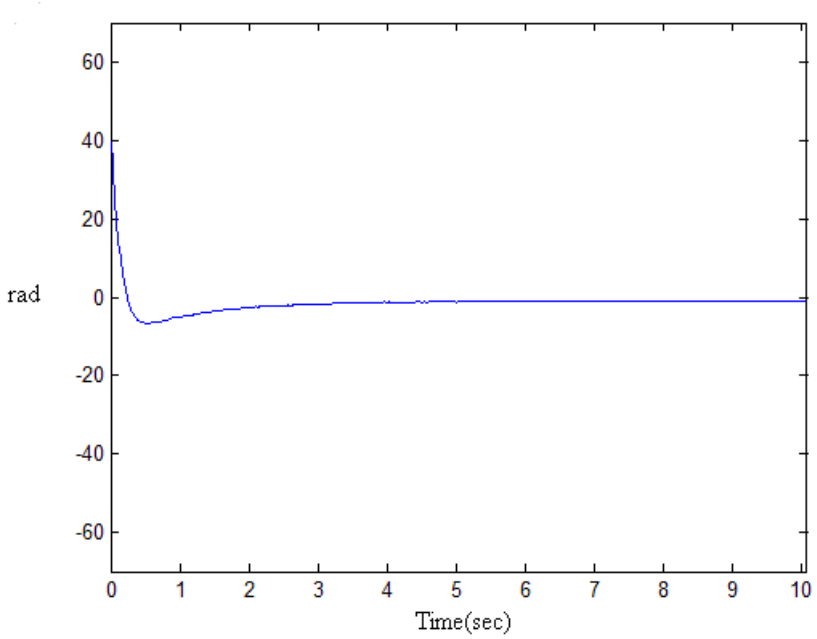

Fig. 6-13. The tracking error $z_1 = x_1 - y_d$ (case 1).

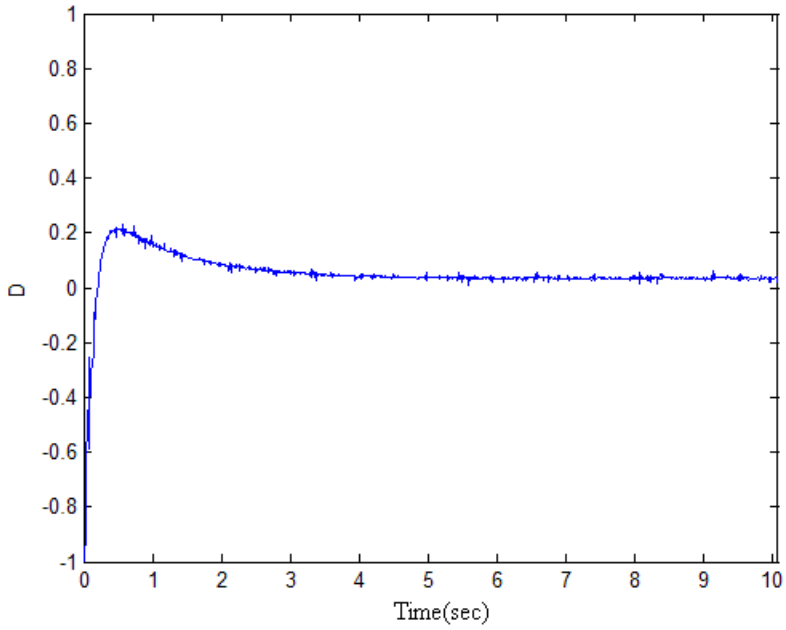

Fig. 6-14. The switch duty ratio *D* (case 1).

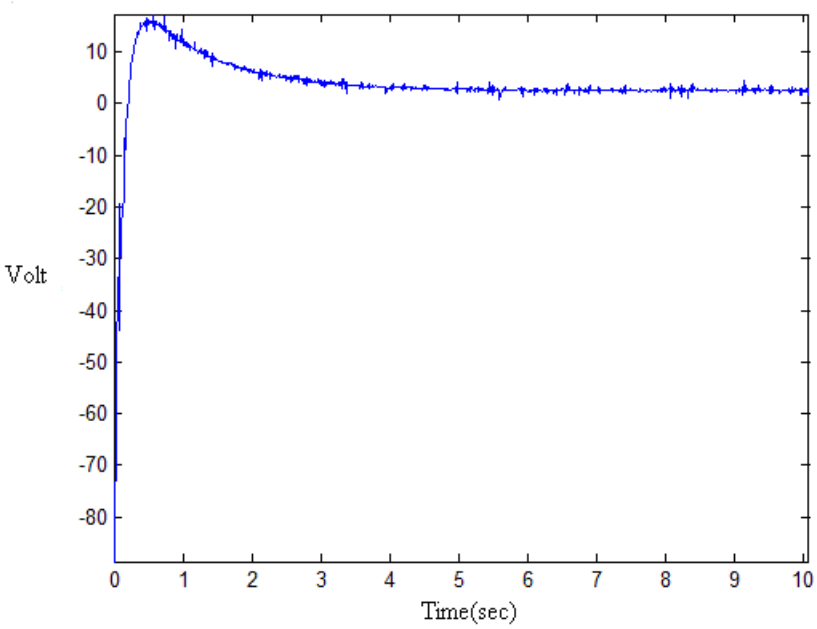

Fig. 6-15. The control input voltage u (case 1).

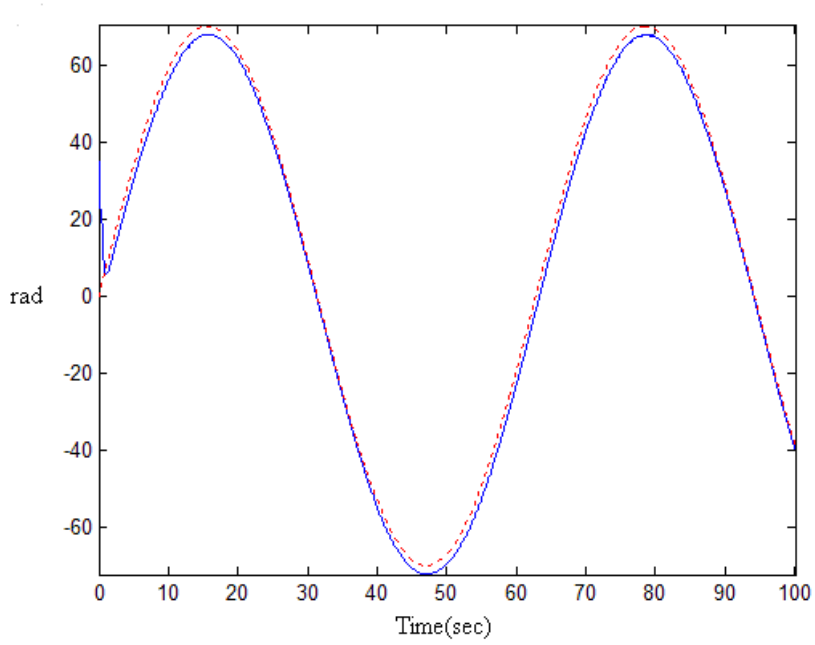

Fig. 6-16. The system output x_1 (solid line) and bounded reference y_d (dashed line) (case 2).

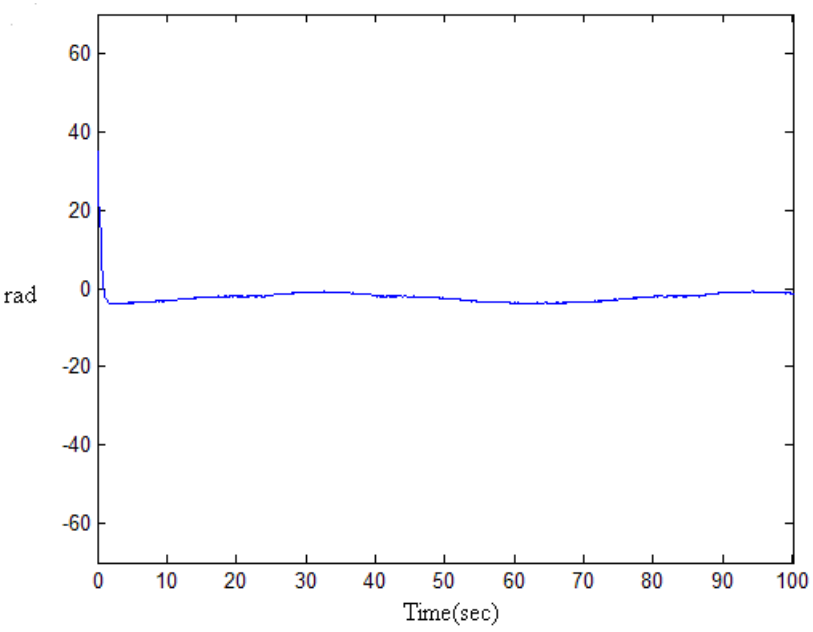

Fig. 6-17. The tracking error $z_1 = x_1 - y_d$ (case 2).

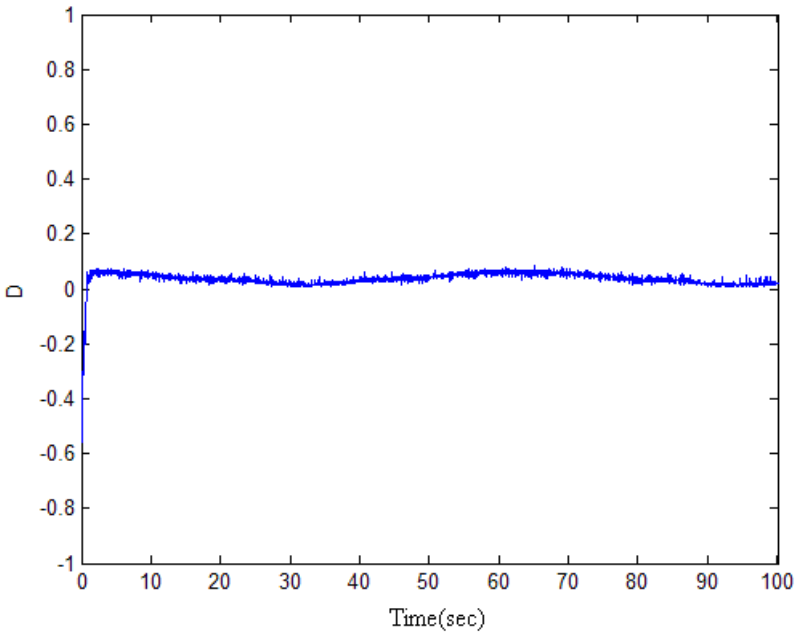

Fig. 6-18. The switch duty ratio *D* (case 2).

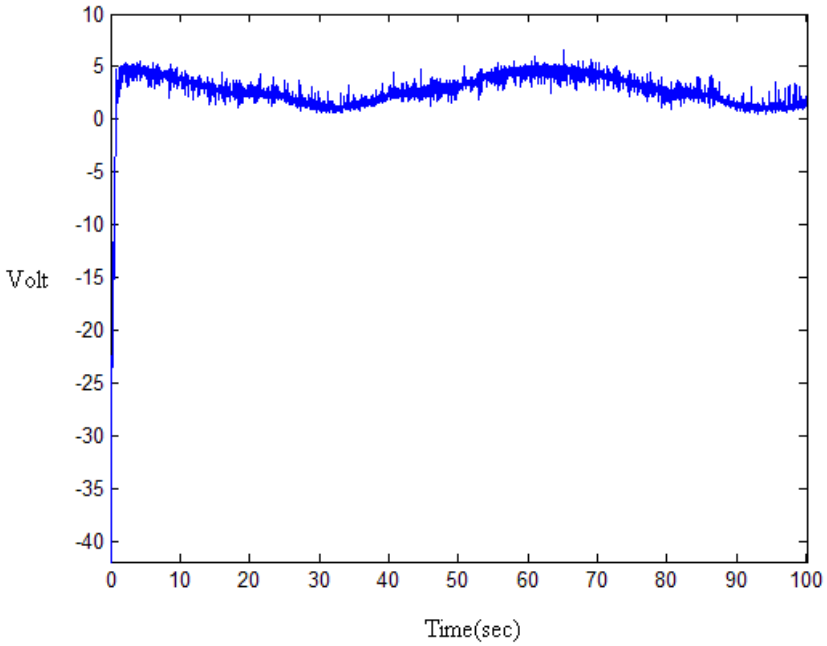

Fig. 6-19. The control input voltage u (case 2).

6.5 Conclusions

In this chapter, the genetic adaptive fuzzy-neural control scheme as discussed in Chapter 4 is used in dc servomotor for simulation and experiment.

Compare with experimental results, we find that performance in simulation is better than in experiment with the same conditions. That results from the temperature of motor, the signal disturbance, the damage of component or equipment, the preciseness of circuits design and so on. However, the genetic adaptive fuzzy-neural control still achieves the desired effect in position tracking of the servo motor experiment.

Chapter 7

Summaries and Suggestions for Future Research

7.1 Summaries

In general, the universal approximators are trained via gradient-based methods, which may only find a local minimum solution during the learning process. Therefore, in Chapter 2 of this thesis, we proposed a compact GA (Genetic Algorithms) optimization algorithm to adjust the weightings of fuzzy-neural networks.

The CGA-based fuzzy neural backstepping control scheme has been proposed to control a class of single-input single-output (SISO) nonlinear systems in Chapter 3. The control scheme incorporates the backstepping design technique with a fuzzy neural network. The weightings of the fuzzy neural controller are tuned on-line via the CGA approach. The free parameters of the fuzzy-neural controller can be successfully tuned on-line via the CGA approach with a special evaluation mechanism. In order to guarantee that the system states are confined to the safe region, a safe controller is incorporated into the CGA-based fuzzy neural backstepping controller. Simulation results have shown that the proposed CGA-based fuzzy neural backstepping control scheme can rapidly learn the unknown system dynamics, and achieve favorable tracking performance.

Based on the presented method in Chapter 3 for SISO system, the CGA-based fuzzy neural backstepping control scheme to control a class of multiple-input multiple-output (MIMO) nonlinear systems is proposed in Chapter 4. In addition, the genetic adaptive fuzzy-neural control scheme for a class of MIMO robotic systems has been developed in Chapter 5.

Finally, a genetic adaptive fuzzy-neural control scheme for tracking control of a DC servomotor has been proposed in Chapter 6. The experimental equipment is composed of a person computer, USB I/O 24 module, micro controller (82G516), DC-DC converter and a DC servo motor system. Simulation and experiment results have shown that the proposed genetic adaptive fuzzy-neural control scheme can rapidly learn the unknown system dynamics, and achieve favorable tracking performance.

7.2 Suggestions for Future Research

The future research is recommended to work toward the following two directions. Since in this thesis we only discuss with the affine nonlinear systems, the first future work is expected to extend the presented method to more general nonlinear system. In addition, to demonstrate the applicability of propose method in practical system, we use the control scheme for tracking control of a DC servo motor. So the second future work is that the control scheme could be applied in other practical systems, such as the voltage control of DC-DC converters.

References

[1] W.Y. Wang, Y.G. Leu, and T.T. Lee, "Output-feedback control of nonlinear systems using direct adaptive fuzzy-neural controller," *Fuzzy Sets and Systems* 140, pp. 341-358, 2003.

[2] W. Y. Wang, M. L. Chan, T. T. Lee, and C. H. Liu, "Recursive Back-stepping Design of Adaptive Fuzzy Controller for Strict Output Feedback Nonlinear Systems," *Asian Journal of Control*, vol. 4, no.3, Sept., 2002.

[3] W. Y. Wang, M. L. Chan, C. C. Hsu, and T. T. Lee, "H_∞ Tracking-Based Sliding Mode Control for Uncertain Nonlinear Systems via an Adaptive Fuzzy-Neural Approach," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 32, no.4, pp.483-492, 2002.

[4] L. X. Wang, "A Supervisory Controller for Fuzzy Control Systems that Guarantees Stability," *IEEE Trans. On Automatic Control*, vol. 39, no. 9, pp.1845-1847, 1994.

[5] Y.G. Leu, T.T. Lee, and W.Y. Wang, "Observer-based Adaptive Fuzzy-Neural Control for Unknown Nonlinear Dynamical Systems," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 29, no. 5, pp.583-591, Oct., 1999.

[6] C.H. Wang, H.L. Liu, T.C. Lin, "Direct adaptive fuzzy-neural control with state observer and supervisory controller for unknown nonlinear dynamical systems," *IEEE Transactions on Fuzzy Systems,* vol. 10, no.1, pp.39-49, 2002.

[7] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward

networks are universal approximators," *Neural Networks*, no. 2, pp. 359-366, 1989.

[8] Li-Xin Wang, *Adaptive Fuzzy Systems and Control*, Prentice Hall, 1994.

[9] H. K. Lam, F. H. Leung, and P. K. S. Tam, "Design and stability analysis of fuzzy model-based nonlinear controller for nonlinear systems using genetic algorithm," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 33, no. 2, pp. 250–257, Apr. 2003.

[10] C.H. Chou, "Genetic algorithm-based optimal fuzzy controller design in the linguistic space," *IEEE Transactions on Fuzzy Systems,* vol. 14, no.3, pp. 372 - 385, 2006.

[11] L. Cai, A. B. Rad, and W.L. Chan, "A Genetic Fuzzy Controller for Vehicle Automatic Steering Control," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 2, pp.529-543, 2007.

[12] A. Mirzaei, M. Moallem, B. M. Dehkordi, and B. Fahimi, "Design of an Optimal Fuzzy Controller for Antilock Braking Systems," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 6, pp.1725-1730, 2007.

[13] M. Cheng, Q. Sun, and E. Zhou, "New Self-Tuning Fuzzy PI Control of a Novel Doubly Salient Permanent-Magnet Motor Drive," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 3, pp.814-821, 2006.

[14] F.J. Lin, P.K. Huang, and W.D. Chou, "Recurrent-Fuzzy-Neural- Network -Controlled Linear Induction Motor Servo Drive Using Genetic Algorithms," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 3, pp.1449-1461, 2007.

[15] Z. Yang, T. Hachino, and T. Tsuji, "Model reduction with time delay combining the least-squares method with the genetic algorithm," *IEE Proc.*

Control Theory Appl., vol. 143, no. 3, 1996.

[16] Z. Michalewicz, *Genetic Algorithms + Data Structure = Evolution Program*, Springer-Verlag, 1996.

[17] D. Goldberg, *Genetic Algorithms in Search, Optimisation, and Machine Learning*, Addison-Wesley, 1989.

[18] D. Lawrence, *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, 1991.

[19] M. Iwasaki, M. Miwa, and N. Matsui, "GA-based evolutionary identification algorithm for unknown structured mechatronic systems," *IEEE Transaction on Industrial Electronics*, vol. 52, pp. 300-305, 2005.

[20] Antonio Gonzalez and Raul Perez, "Selection of Relevant Features in a Fuzzy Genetic Learning Algorithm, " *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 31, no. 3, June 2001.

[21] R. Caponetto, L. Fortuna, S. Graziani, and M. Xibilia, "Genetic algorithms and applications in system engineering: a survey," *Trans. Inst. Meas. Control,* vol. 15, pp. 143-156, 1993.

[22] Peter Merz and Bernd Freisleben, "Fitness Landscape Analysis and Memetic Algorithms for the Quadratic Assignment Problem," *IEEE Trans. On Evolutionary Computation,* vol. 4, no. 4, pp. 337-352, 2000.

[23] G W Game and C D James, "The Application of Genetic Algorithms to the Optimal Selection of Parameter Values in Neural networks for Attitude Control System,*" IEEE Colloquium on High Accuracy Platform Control in Space,* pp. 3/1-3/3, 1993.

[24] S.W. Moon and S.G. Kong, **"**Block-based neural networks," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 307 –317, 2001.

[25] C.T. Lin and C.P. Jou," GA-based fuzzy reinforcement learning for control

of a magnetic bearing system, ”*IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 30, no. 2, pp. 276 –289, April 2000.

[26] C.H. Wang, H.L. Liu, and C.T. Lin, “Dynamic optimal learning rates of a certain class of fuzzy neural networks and its applications with genetic algorithm, ”*IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 31, no. 3 , pp.467 –475, June 2001.

[27] W. A. Farag, Victor H. Quintana, and Germano Lambert-Torres,” A Genetic-Based Neuro-Fuzzy approach for modeling and control of dynamical systems,” *IEEE Transactions on Neural Networks,* vol. 9, no. 5, Sept. 1998.

[28] M. Brown and C. J. Harris, *Neurofuzzy adaptive modelling and control*, Prentice Hall, 1994.

[29] W.Y. Wang and Y.H. Li, “Evolutionary learning of BMF fuzzy-neural networks using a reduced-form genetic algorithm,” *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 33, pp.966 – 976, 2003.

[30] J. L. Castro, “Fuzzy logical controllers are universal approximators,” *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 25, pp 629-635, Apr. 1995.

[31] P. A. Ioannou and J. Sun*, Robust Adaptive Control,* Prentice-Hall, 1996.

[32] B. S. Chen , C. H. Lee, and Y. C. Cheng, “H^{∞} tracking design of uncertain nonlinear SISO systems: Adaptive fuzzy approach,” *IEEE Transactions on Fuzzy Systems,* vol. 4, pp. 23-43, Feb. 1996.

[33] C.W. Park and Y.W. Cho, “T-S model based indirect adaptive fuzzy control using online parameter estimation,” *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 34, no. 6, pp.2293-2302, 2004.

[34] J.W. Park, R.G. Harley and G.K. Venayagamoorthy, **“**Indirect adaptive

control for synchronous Generator: comparison of MLP/RBF neural networks approach with Lyapunov stability analysis," *IEEE Transactions on Neural Networks*, vol. 15, no. 2, pp. 460 –464, 2004.

[35] J. Y. Choi and J.A. Farrell, "Adaptive observer backstepping control using neural networks," *IEEE Transactions on Neural Network*, vol. 12, no. 5, pp.1103-1112, 2001.

[36] C.F. Hsu, C.M. Lin, and T.T. Lee, "Wavelet Adaptive Backstepping Control for a Class of Nonlinear Systems," *IEEE Transactions on Neural Networks*, vol. 15, no. 3, 2004.

[37] T. Zhang, S.S. Ge and C.C. Hang, "Adaptive neural network control for strictfeedback nonlinear systems using backstepping design," Automatica, vol. 36, pp.1835-1846, 2000.

[38] Y. Zhang, P.Y. Peng, and Z.P. Jiang, "Stable Neural Controller Design for Unknown Nonlinear Systems Using Backstepping," *IEEE Transactions on Neural Networks*, vol. 11, no. 6, 2000.

[39] C. Kwan and F. L. Lewis, "Robust backstepping control of nonlinear systems using neural networks," *IEEE Transactions on Systems, Man and Cybernetics, Part A*, vol. 30, pp. 753–765, 2000.

[40] T. Knohl and H. Unbehauen, "ANNNAC—extension of adaptive backstepping algorithm with artificial neural networks," *Inst. Elect. Eng. Proc. Contr. Theory Appl.*, vol. 147, pp. 177–183, 2000.

[41] Y. Zhang, P.Y. Peng, and Z.P. Jiang, "Stable Neural Controller Design for Unknown Nonlinear Systems Using Backstepping," *IEEE Transactions on Neural Networks*, vol. 11, no. 6, November 2000.

[42] Q. H. M. Meng, "Application of an efficient neural network to identification and adaptive control of unknown robot dynamics,"

Internation Journal Intelligent Control System, vol. 1, no. 4, pp. 459–468, 1996.

[43] G. Yuzheng, W. Pengyung, "An adaptive fuzzy sliding mode controller for robotic manipulators," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 33, no. 2, pp.149-159, 2003.

[44] B. K. Yoo, W. C. Ham, "Adaptive control of robot manipulator using fuzzy compensator," *IEEE Transactions on Fuzzy Systems,* vol. 8, no.2, pp. 186-199, 2000.

[45] S. Jagannathan, F. L. Lewis, "Robust backstepping control of a class of nonlinear system using fuzzy logic," *Information Sciences*, vol. 123, no. 3, pp. 223-240, 2000.

[46] J. E. Slotine and W. Li, *Applied Nonlinear Control.* Englewood Cliffs, NJ: Prentice-Hall, Inc., 1991.

[47] H. B. Zhang, C. G. Li, and X. F. Liao, "Stability Analysis and H_∞ Controller Design of Fuzzy Large-Scale Systems Based on Piecewise Lyapunov Functions," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 36, pp.685-698, 2006.