STUDY ON SIMPLE ADAPTIVE CONTROL FOR NONLINEAR SYSTEMS

January 2008

MUHAMMAD YASSER

Graduate School of Science and Technology

CHIBA UNIVERSITY

(千葉大学学位申請論文)

STUDY ON SIMPLE ADAPTIVE CONTROL FOR NONLINEAR SYSTEMS

非線形システムの簡易型適応制御に関する研究

2008 年 1 月

千葉大学大学院 自然科学研究科

情報科学専攻 情報システム科学講座

ムハンマド ヤッセル

Study on Simple Adaptive Control for Nonlinear Systems

A dissertation
submitted to
the Graduate School of Science and Technology
of Chiba University
in partial fulfillment of the requirements for the
degree of
Doctor of Philosophy
in
Information Science

By

Muhammad Yasser

January 2008

Acknowledgements

All praises and thanks to ALLAH for ALLAH's Mercy and Grace to me and all in this life.

I would like to express my deepest gratitude to my research advisor and supervisor, Professor Takashi Yahagi, for giving me valuable advice, guidance, and encouragement on this research and influencing my way of thinking. He taught me many things not only on my research but also on how an engineer and a researcher should be. I have benefited from his vast wealth of knowledge and his constructive comments during this research.

I would also like to thank to Associate Professor Jianming Lu for for his contructive comments.

I wish to express my gratitude to Research Assistant Dr. Sekiya Hiroo for his guidance and encouragement throughout my study and research. He gave me many benefit comments and practical instructions on my research.

I am greatly thankful to all of the previous and current members of Yahagi, Lu, and Sekiya Laboratory for their help and support. Special thanks to Dr. Agus Trisanto, Mr. Ayman Haggag, Mr. Mohamed Ghoneim and Miss Xiaowei Zhang for their help and discussions with me.

Finally, I would like to express my great gratitude to my late father, my mother, my sister and brother, and all of my good friends and other relatives for their encouragement and support.

May ALLAH bless all of us in this life and the hereafter.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In any control problem, there will typically be difficulties in designing a good controller for the actual plant. These difficulties are caused by unknown or unmodelled dynamics, variation or changes in system parameters, disturbances, uncertainties, and discrepancies between the actual plant and the mathematical model developed for the controller design [1–18]. Furthermore, since the actual plants are nonlinear in reality, nonlinearities also contribute to those difficulties [16].

Simple adaptive control (SAC) is a class of direct adaptive control methods. It was proposed by Sobel et al. [19,20]. SAC algorithm is called "simple" because it does not use identifiers or observers in its feedback loop. For linear plants with unknown structures, SAC is an important class of adaptive control schemes and can be applied to solve the above problems of designing controllers. However, for nonlinear plants with unknown structures, it is not possible to ensure a perfect plant output that follows the output of a reference model by using conventional SAC [21, 22]. On the other hand, neural networks and sliding mode control (SMC) have shown good performance for nonlinearities. However, there are also some weak points and drawbacks in the conventional methods of using neural networks and SMC.

By combining SAC with neural networks and with SMC, this research expects to develop methods of SAC for nonlinear systems, that can solve the above problems of applying the conventional SAC for nonlinear systems. Furthermore, it is also expected that the weak points and drawbacks in the existing conventional methods of using neural networks and SMC can be reduced.

A brief background of adaptive control, SAC, neural networks, and SMC will be presented in the following subsections for preliminary knowledges.

1.1.1 Adaptive Control

Adaptive control was developed as an attempt to overcome difficulties of controlling plants with unknown parameters and uncertainties [1, 2, 5, 7, 8, 13, 14, 17]. There have been many attempts over years to define adaptive control formally. At an early symposium in 1961, a long discussion ended with the following suggestion: "An adaptive system is any physical system that has been designed with an adaptive viewpoint." A renewed attempt was made by an IEEE committe in 1973. It proposed a new vocabulary based on notions like self-organizing control (SOC) system, parameter-adaptive SOC, performance-adaptive SOC, and learning control system. However, these efforts were not widely accepted. Åström et al. has defined in their book [7] that "An adaptive controller is a controller with adjustable parameters and a mechanism for adjusting the parameters."

Historically, adaptive control was conceived in the 1950s. Since then, it has firmly remained in the mainsteram of research activity with hundred papers and several books published every year [8]. In its about fifty years existence, adaptive control theory has been steadily growing into a well-formed scientific discipline: from inventions to rigorous problem formulations; from solutions of basic problems to more demanding task for broader classes of systems; from questions of existence and solvability to application-oriented issues of robustness and performance [8]. The most success of adaptive control is for plant models in which the unknown parameters appear linearly.

In the early 1950s, there was an extensive research on adaptive control in connection with the design of autopilots for high-performance aircraft [7]. Such aircraft operate over a wide range of speeds and altitudes. It was found that ordinary constant-gain, linear feedback control could work well in one operating condition but not over the whole flight regime. A more sophisticated controller that could work well over a wide range of operating conditions was therefore needed. After a significant effort, it was found that gain scheduling of adaptive control was a suitable technique for flight control systems. However, the interest in adaptive control diminished partly beacuse the adaptive control problem was too hard to deal with using the techniques that were available at the time.

In the 1960s, there were much research in the control theory that contributed to the development of adaptive control [7]. State space and stability theory were introduced. There were also important results in stochastic control theory. Dynamic programming, introduced by Bellman, increased the understanding of adaptive processes. Fundamental contributions were also made by Tsypkin, who showed that many schemes for learning and adaptive control could be described in a common framework. There were also major developments in system identification. A renaissance of adaptive control occured in the 1970s, when different estimation schemes were combined with various design methods. Many applications were reported, but theoretical results were very limited.

In the late 1970s and early 1980s, analysis and proofs of stability of adaptive system appeared [23–25]. It also sparked new and interesting research into the robustness of adaptive control, as well as into controllers that are universally stabilizing [7]. Research in the late 1980s and early 1990s gave new insights into the robustness of adaptive controllers. Investigations of nonlinear systems led to significantly incerased understanding of adaptive control. Lately, it has also been established that adaptive control has strong relations to ideas on learning that are emerging in the field of computer science.

There have been many experiments on adaptive control in laboratories and industry. The rapid progress in microelectronics was a strong stimulation. Interaction between theory and experimentation resulted in a vigorous development of the field. As a result, adaptive controllers started to appear commercially in the early 1980s. And more recently, the research focus of adaptive control theory has been transferred to nonlinear and time varying plants [26].

1.1.2 Simple Adaptive Control (SAC)

SAC method was proposed by Sobel et al. [19, 20] as an attempt to simplify the adaptive controllers. This method is called "simple" because it does not use identifiers or observers in its feedback loop. Other advantages of this method are that the order of the reference model used in this method is allowed to be much smaller than the order of the real plant, and this method was developed in consideration of multi-input multi-output (MIMO) plants. Furthermore, this method is also easily implementable and applicable to real control problems [13].

Basically, adaptive control methods can be divided into two categories: direct or implicit adaptive control and indirect or explicit adaptive control [13, 27]. Block diagrams of a direct and an an indirect adaptive control are shown in Figs 1.1 and 1.2, respectively. Indirect adaptive control methods separate parameter identification and control schemes. Direct control methods merge the identification and control function into one scheme. In such direct methods, the control gains are computed directly without an explicit identification of the system parameters. Thus, with fewer computations to perform, direct adaptive control has one advantage over indirect adaptive control: speed [13].

Model reference adaptive methods might be classified as evolving from three different approaches. First is the full state access method described by Landau in [28], which assumes that the state variables are measurable. Second is the input-output method, which originates from Monopoli's augmented error signal concept [29]. In this approach, adaptive observers are incorporated into the controller to overcome the inability to access the entire state vector. Third is the SAC approach originated by Sobel et al. [20]. This approach is an output feedback method, which requires neither full state feedback nor adaptive observers. Other important properties of this class of algorithms are as follows [13]:

1. They are applicable to nonminimum phase systems.
2. The order of the plant (physical system) is allowed to be much larger than the order of the reference model.
3. This approach considers plants with multiple inputs and outputs.

Direct model reference adaptive control was first proposed by Osborn, Whitaker, and Keezer, which uses a performance index minimization approach. This approach later became known as the MIT design rule. This was later extended to an accelerated gradient procedure, but stability could not be guaranteed with either procedure. The stability of a linear system combined with an adaptive controller is often in question because of the highly nonlinear nature of the closed loop system [13].

A significant contribution to the theory of the direct model adaptive control of single-input single-output (SISO) sytems was the augmented error signal concept of Monopoli [29]. This technique eliminated the need for either state feedback or derivatives of the output by incorporating adaptive observers into the control law. Monopoli's contribution encouraged much research into this class of algorithms, which will be referred to as the input-output research. Other contributions to this approach include the work of Morse [30], Feuer and Morse [31], and Narendra et al. [23–25, 27]. As stated in subsection 1.1.1, Narendra et al. presented the stability analysis and proofs in [23–25]. The contribution of Narendra et al. [24] settled the question of stability for the input-output approach.

Some of the recent work in the input-output approach to model reference adaptive control (MRAC) is the stability in the presence of disturbances and unmodeled dynamics. This area has become known as the robust adaptive control problem.The observation that earlier stability proofs were not valid in the presence of disturbances and unmodeled dynamics is discussed by Rohrs et al. [32]. However, it is generally accepted that robust stability is a necessary characteristic of any adaptive control algorithm. One of the robust MRAC laws for SISO systems has been proposed that utilizes a fixed σ-modification [13]. Nevertheless, the extension of the input-output approach to MIMO plants is an interesting research area.

During the same time period that Monopoli was working on the augmented error signal concept, Landau [28], Lindorff and Carroll [33], and others were proposing a full state access approach to direct MRAC [13]. Stability was ensured by using either Lyapunov's stability theory or Popov's hyperstability theory. Although this approach applied to MIMO plants, the satisfaction of Erzberger's perfect model-following conditions was required [13].

The simple approach to direct MRAC, also called SAC, of MIMO plants was first proposed by Sobel, Kaufman, and Mabius in their papers [19, 20] in 1979 and 1982, respectively. This approach uses a control structure that is a linear combination of feedforward of the model states and inputs and feedback of the error between plant and model outputs. This class of algorithms requires neither full state access nor satisfaction of the perfect model-following conditions. Asymptotic stability is ensured provided that the plant is almost strictly positive real (ASPR) [13].

The appealing characteristics of this SAC algorithm over indirect and other direct model reference adaptive methods include [13]:

1. independence on plant parameter estimates,
2. applicability to MIMO plants,
3. sufficiency conditions that are independent of plant dimension,
4. control calculation that does not require adaptive observers or full state feedback,
5. ease of implementation,
6. and successful experimental validation.

The above characteristics make this SAC approach attractive to the practitioner.

This SAC approach is described in detail by Sobel, Kaufman, and Mabius [20]. Later this method was developed further by Bar-Kana and Kaufman [34, 35]. Bar-Kana and Kaufman extended the SAC approach to a robust SAC law for ASPR MIMO plants. Later, this ASPR algorithm was extended again by Bar-Kana and Kaufman [34] to the class of non-ASPR plants for which there exists a known dynamic output stabilizing feedback with transfer matrix $H(s)$. In this case, it is shown that an augmented system consisting of the plant in parallel with $H^{-1}(s)$ is ASPR. For practical applicability of SAC, the problem of designing a feedforward compensator containing $H^{-1}(s)$ to form an ASPR augmented system is discussed by Iwai and Mizumoto [36].

As the recent trend of focus of adaptive control theory has been transferred to nonlinear and time varying plants [26], the implementation of SAC to solve the control problems caused by the nonlinearities of nonlinear plants started to be considered. However, for nonlinear plants with unknown structures, using only conventional SAC described in [19, 20, 34, 35] cannot ensure a perfect plant output that follows the output of a reference model [21, 22]. Attempts to extend SAC to control nonlinear plants have been performed by combining it with neural networks [21,22]. However, the theoretical analysis for proofs of convergence and stability is not performed yet, and class of nonlinear systems which can be controlled by this method of SAC with neural networks is not provided yet either. Furthermore, the use neural networks itself has some weak points such as will be explained in the next subsections.

A brief mathematical and theoretical explanation about the conventional SAC for linear plants as proposed in [19, 20, 34, 34, 35] is given in section 2.2 as a further introduction.

Based on the basic idea proposed in [21, 22], this thesis will extend the conventional method of SAC to provide SAC methods to control nonlinear plants. Furthermore, a thorough analysis for proofs of convergence and stabilty will also be provided. From the convergence and stability analysis, class of the control target nonlinear plants of the methods of this thesis is defined.

1.1.3 Neural Network

A neural network is a machine that is designed to model the way of the brain performing a spesific task [3, 4, 6, 9, 11, 37]. A definition of neural network in the view as an adaptive system was proposed by Haykin [37] as follows: "A neural network is a massively parallel distributed processor that has a natural propensity for storing experiental knowledge and making it avalaible for use. It resembles the brain in two respects:

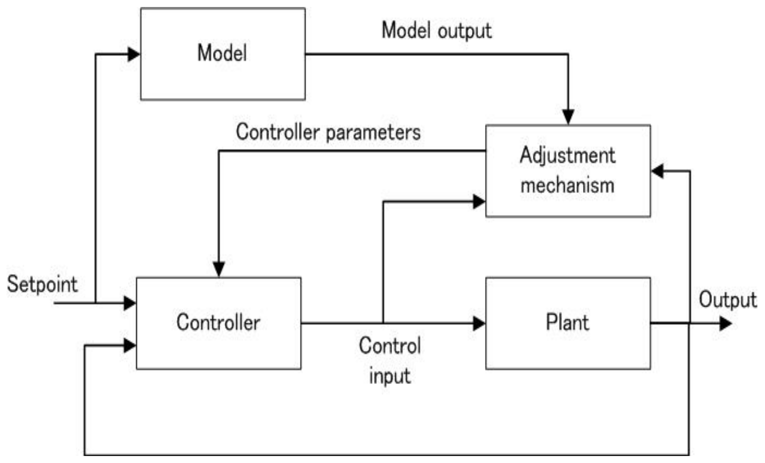

Figure 1.1: Block diagram of a direct adaptive control system

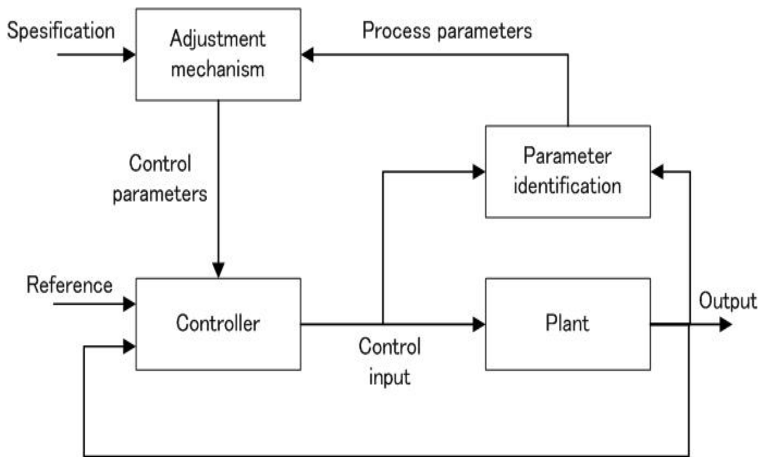

Figure 1.2: Block diagram of an indirect adaptive control system

1. Knowledge is acquired by the network through a learning process.
2. Interneuron connection strengths known as synaptic weights are used to store the knowledge."

The procedure used to perform the laerning process is called a learning algorithm. This learning algorithm is the function to modify the synaptic weights of the network in and orderly fashion so as to achieve a desired design objective.

Research on neural networks has been motivated in the begining by the recognition that the brain computes in an entirely different way from the conventional digital computer [37]. The struggle to understand the brain owes much to the pioneering work of Ramon y Cajal in 1911, who introduced the idea of neurons as structural constituents of the brain. The brain itself is a highly complex, nonlinear, and parallel information processing system. It has the capability of organizing neurons so as to perform certain computations, such as pattern recognition, perception, and motor control, many time faster than the fastest digital computer exists today.

In 1943, McCulloch and Pitts proposed a mathematical model of the neurons and showed how neuronal-like networks could be computed. It described a neuron to be developed of parts such as a net input (summarizing all of the inputs for the neuron), a threshold function, and an output, as shown in Fig. 1.3.

The first set of ideas of learning in neural networks was contained in Hebb's book entitled "The Organization of Behaviour" in 1949. Before Hebb's work, it was believed that some physical change ust occur in a neural network to support learning, however, it could not be determined what this change was [11]. Hebb assumed that a reasonable biological change would be needed to strengthen the connections between elements of the neural network only when both the pre-synaptic and post-synaptic cell membranes were active simultaneously. The essence of Hebb's ideas occurs in various learning paradigms. Although the details of the rules for changing the weights may be different, Hebb's essential notion that the strength of connections between the units must change in response to some function of the correlated activity of the connected units has been adpted in many learning models.

In 1951, Edmonds and Minsky built their learning machine using Hebb's idea. However, the real beginning of a meaningful neuron-like network learning can be traced to the work of Rosenblatt in 1962. Before that, in 1958 Rossenblat published a book on perceptrons, a machine that is capable of learning how to classify information by adapting the weights. In 1960–1962, Widrow and Hoff developed adalines and LMS rule. In their book in 1969, Minsky and Papert show theoretical limits of perceptrons as general computers.

From 1969 to 1982, there was about 23 years of hybernation in the research and development of neural networks. But, in that period, there were some 'stubborn' individuals, such as Grossberg, Amari, Fukushima, Kohonen, and Taylor, who continued doing research in neural networks.

In a breakthrough paper published in 1982, Hopield introduced a neural network architecture which is called Hopfield network. In this paper, he describes how computational capabilities can be built from neural networks. This paper marked a re-emergence of the field of neural networks.

The backpropagation is another approach which has been widely used in the neural network paradigms. The conceptual basis of backpropagation was first presented in 1974 by Werbos. Later, it was independently re-invented in 1986 by Rumelhart et al.. Rumelhart et al. introduced in their book entitled “Parallel Distributed Processing” a broad persepective of the neural network approaches.

The best known neural network architecture is the multilayer feedforward neural network [9]. It is a static network that consist of a number of layers: an input layer, one or more hidden layers, and an output layer connected in a feedforward way. Each layer consists of a number of neurons, and each neuron in each layer is linked with every neurons in previous and next layers with weighted connections. In 1989 it was shown independently by Hornik et al., Funahashi, and Cybenko that a multilayer feedforward neural network with one or more hidden layers is sufficient in order to approximate any continuous nonlinear function arbitrarily well on a compact interval, provided sufficient hidden neurons are available [9]. The structure of multilayer feedforward neural network is shown in Fig. 1.4.

The following features of neural networks makes them very attractive and promising for application to modelling and control of nonlinear plants [9]:

1. Neural networks are universal approximators: it is proven tht any continuous nonlinear function can be aproximated arbitrarily well over a compact interval by a multilayer neural network that consists of one or more hidden layers.

2. Parallel distributed processing: the network has a highly parallel structure and consists of many processing elements with a very simple structure, which is interesting from the viewpoint of implementation.

3. Hardware implementation: dedicated hardware is possible, resulting in additional speed.

4. Learning and adaptation: the intelligence of neural networks comes from their generalization ability with respect to fresh, unkown data. Online adaptation of the weights is possible.

5. Multivariable systems: neural networks have naturally many inputs and outputs, which makes it easy to model multivariable systems.

Thus, unknown nonlinear functions in dynamical neural networks can be parametrized by using multilayer neural networks. Furthermore, there is very

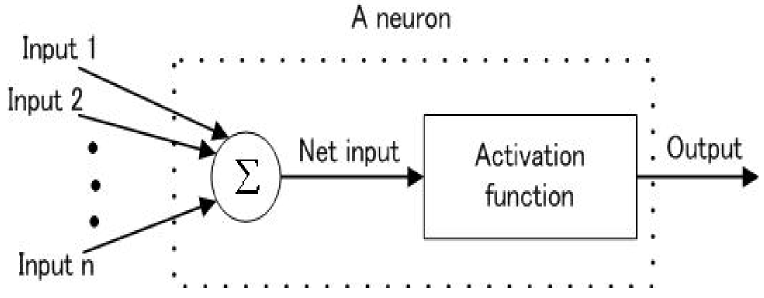

Figure 1.3: Structure of a neuron

strong relation between neural networks for control with the field of adaptive control, since the multilayer feedforward neural networks can be readily thought of as performing an adaptive, nonlinear vector mapping [4].

However, there also a number of weak points of neural networks, such as:

1. the existence of many local optima in learning algorithms,

2. the choice of complexity of the neural networks,

3. the convergence and stability analysis of the dynamical systems that contain neural network architectures (related to the complexity of the neural networks in point no. 2).

There has been many research works presenting theories and applications of neural networks for control, where some of the most recent works are such as [38–52]. Some of them also discussing about the convergence and stability analysis of the dynamical systems that contain neural networks [41, 42, 44, 46, 49–52].

As a further introduction, brief mathematical and theoretical explanations about the multilayer feedforward neural networks and the backpropagation algorithm are given in sections 2.4 and 2.5, respectively.

This thesis uses multilayer feedforward neural networks based on the backpropagation algorithm for the learning process in its proposed methods of combination of SAC with neural networks.

1.1.4 Sliding Mode Control (SMC)

In the formulation of any control problem, there, there will typically be discrepancies between the actual plant and the mathematical model developed for controller design. This mismatch can be caused by unmodelled dynamics, variation in system parameters or the approximation of complex plant

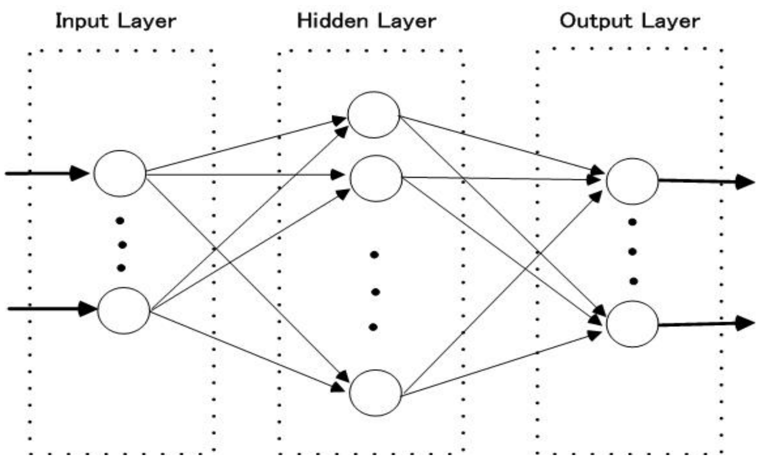

Figure 1.4: Structure of a multilayer feedforward neural network

behaviour by a straightforward model [12]. In a design process, the controller must be ensured to have the ability to produce the required performance levels in practice despite such plant and model mismatches. This has led to an intense interest in the development of "robust" control methods to seek the solution for this problem. One particular approach to robust controller design is the SMC methodology [12, 15, 16, 18].

SMC is a particular type of variable structure control. Variable structure control systems (VSCS) are characterised by a suite of feedback control laws and a decision rule. The decision rule, termed the switching function, has as its input some measure of the current system behaviour and produces as an output the particular feedback controller which should be used at that instant in time. The result is a variable structure system, which can be regarded as a combination of subsystems where each subsystem has a fixed control structure and is valid for specified regions of system behaviour. One of the advantages of introducing this additional complexity into the system is the ability to combine useful properties of each of the composite structures of the system. Furthermore, the system may be designed to posses new properties not present in any of the composite structures alone. Utilisation of these natural ideas began in the Soviet Union in the late 1950s.

SMC is fundamentally a consequence of discontinuous control [18]. In the early 1960, discontinuous control (at least in its simplest form of bang-bang control) was a subject of study for mechanics and control engineers. As an example, Hamel's work in 1949 in France and Cypkin's in 1955 and Emelyanov's in 1963 in Russia solved in a rigorous way the problem of oscillations appearing in bang-bang control systems. These first studies turned rapidly to synthesis problems in various ways. One of them was related to optimal control, another to linearization and robustness. Although both approaches and objectives were at the beginning quite different, it is interesting to note that they turned out to have much in common.

In fact, it was when looking for ways to design what we call now robust control laws that sliding mode was discovered at the beginning of 1960s. For the needs of military aeronautics, and even before the term of robustness was used, control engineers were looking for control laws insensitive to the variations of the system to be controlled. The existing linear controllers used at these time did not bring enough compensation to use high gains required to get parametric insensitivity.

SMC evolved from the pioneering work in Russia of Emelyanov and Barbashin in the early 1960s. The ideas did not appear outside of Russia until the mid 1970s when a book by Itkis in 1976 and a survey paper by Utkin in 1977 [53] were published in English. Although this subject of SMC has been treated in many papers, such as [53–73], and books, such as [12, 15, 18], it remains the object of many studies (theoretical or related to various applications) until nowadays.

In SMC, the controllers are designed to drive and then maintain the system states to lie within a neighbourhood of the switching function or usually also called "sliding surface". Thus, there are two phases in SMC: the approaching phase and the sliding phase, as shown in Fig. 1.5. In the approaching phase, states of system is outside the sliding surface. In this phase, a corrective control is applied to drive the states of system onto the sliding surface. When any states that reach the sliding surface remain on it, a sliding mode or sliding motion is occured and a sliding phase is started. In this sliding phase, an equivalent control is capable of maintaining the system states to stay on the sliding surface and driving them to the origin. The steps in designing a sliding mode controller are: to construct a sliding surface that represents a desired system dynamics, and then to develop a switching control law such that a sliding mode exists on every point of the sliding surface, and any state outside the surface is driven to reach the surface in a finite time. There are two main advantages of this SMC approach:

1. it robustness against a large class of perturbations or model uncertainies,
2. and its applicability to control nonlinear systems which are usually difficult to control using conventional linear feedback control laws.

However, the general approach of conventional SMC also has two drawbacks [62,66,69,73]:

1. the chattering phenomenon,
2. and the difficulty in calculating its equivalent control law.

In the design of the conventional SMC law, it is assumed that the corrective control, which usually uses a discontinuous switching function, can be switched from one value to another infinitely fast [71]. However, this is impossible to achieve in practical systems because finite tim delays are present for the control computation, and limitations exist in physical actuators.

This nonideal switching results in a first drawback, the chattering phenomenon. This chattering phenomenon is highly undesirable [55,62,71] and may excite the high-frequency unmodeled dynamics which could result in unforeseen instability. Many research works have been performed and focused in solving the first problem, the chattering phenomenon, such as [55,56,58,61,65,67,71]. One of the methods to overcome the chattering phenomenon is to include an equivalent control law into the SMC law [12,18].

Thus, the second drawback, which is the difficulty in calculating the equivalent control law of SMC, appears. This second drawback is caused by the requirement of thorough knowledge of parameters and dynamics of the nominal controlled plant [62,66,69,73]. Since those parameters and dynamics are difficult to obtain or even unknown, the calculation of the equivalent control law of SMC is very difficult and causes computational burden [73–75]. To overcome the second problem, the difficulty in calculating the equivalent control law of SMC, recently, intelligent techniques based on fuzzy logic and neural networks have been applied to SMC [62,64,66,68,73]. However, those methods still require complex calculation process and consume time to calculate the control law of SMC.

For a further introduction, a brief mathematical and theoretical explanation about the general approach of SMC is provided in section 5.2.

One of the methods proposed by this thesis is a method of adaptive SMC strategy using SAC. It is expected that this proposed method can also overcome the difficulty in calculating the equivalent control law of SMC.

1.2 Objective

To solve the problems of applying the conventional SAC for nonlinear plants, this thesis proposes adaptive control methods for nonlinear plants by combin-

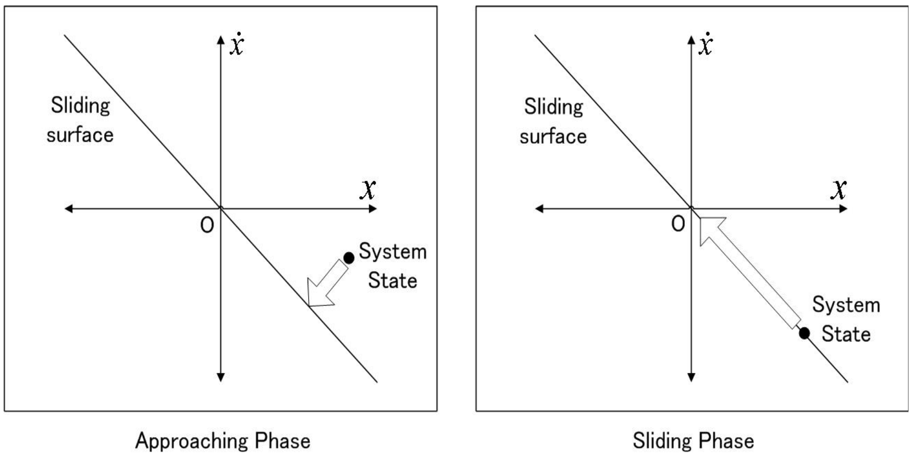

Figure 1.5: Phases of SMC: approaching phase and sliding phase

ing the conventional SAC with neural networks and with SMC. The control input is given by the sum of the output of a simple adaptive controller and the output of either the neural networks or SMC. The role of either the neural networks or SMC is to compensate for constructing a linearized system so as to minimize the output error caused by nonlinearities in the controlled system. The role of the simple adaptive controller is to perform the model matching for the linearized system to a given linear reference model. Furthermore, as the advantages and weak points of each of the conventional methods of neural networks and SMC have been described briefly in section 1.1, it is also expected that in the proposed methods of this thesis those advantages can be maintained and the weak points can be reduced.

Theoretical analysis for proofs of convergences and stabilities of those methods are performed, and the required assumptions and conditions are provided. From the convergence and stability analysis, class of the nonlinear plants that can be controlled using the methods of this thesis is defined. Thus, it is expected that the convergences and stabilities of the methods proposed in this thesis can be guaranteed. Finally, the effectiveness of the proposed methods of this thesis will be confirmed by using either simulations or experiments.

1.3 Outline of the Thesis

This thesis consists of a study on SAC for nonlinear systems, using neural networks and using SMC. The explanations in this thesis are performed in the general scope of MIMO, and it can be easily adapted and applied to the SISO case.

The outline of this thesis is as follows:
Chapter 2 proposes and gives an introduction and a fundamental theoretical explanation about the control method of SAC using a single neural network for nonlinear systems. The analysis for proofs of convergence and stability is discussed. Based on that convergence and stability analysis, the control target nonlinear plants of this method are limited to a class of nonlinear systems with bounded-input bounded-output (BIBO) and bounded nonlinearity.
Chapter 3 proposes and shows applications of a control method using a discrete-time SAC and a neural network, which is developed based the method explained in chapter 2, to SISO and MIMO configurations of a nonlinear magnetic levitation system. In this chapter, the method proposed in chapter 2 is presented in the discrete time domain. The conditions to guarantee the stability of the system by keeping the convergence and stability conditions given in chapter 2 is also provided.
Chapter 4 proposes a control method for nonlinear systems using SAC with multiple neural networks. In this chapter, a more general and thorough theoretical explanation is presented. The theoretical analysis and explanation of convergence and stabilty proofs are developed and presented based on the convergence and stability conditions given in chapter 2.
Chapter 5 proposes and gives an introduction and a fundamental theoretical explanation about a new method of adaptive SMC strategy using SAC for nonlinear systems. The stability analysis and proof are shown. As in chapters 2–4, the control target nonlinear plants of this method also are limited to a class of nonlinear systems with BIBO and bounded nonlinearity.
Finally, chapter 6 gives the general conclusions.

Chapter 2

SAC Using A Neural Network for Nonlinear Systems

2.1 Introduction

Adaptive control methods were developed as an attempt to overcome difficulties connected with the ignorance of systems structure and critical parameter values as well as changing control regimes [2,5,7,14,17]. However, some prior knowledge about the plant to be controlled must be given [14,39,48,76]. Most self-tuning and adaptive control algorithms usually use reference models, controllers, or identifiers of almost the same order as the controlled plant. Since the dimension of the plants in the real world may be very large or unknown, implementation of adaptive control procedures may be very difficult.

To overcome this problem, SAC procedure was developed by Sobel et al. [19, 20] as an attempt to simplify the adaptive controllers, since no observers or identifiers are needed in the feedback loop [34]. Furthermore, the reference model is allowed to be of very low order compared with the controlled plant. For linear plants with unknown structures, SAC is an important class of adaptive control scheme [13,34,36].

Recently, dealing with nonlinear systems using the concept of SAC has been investigated [77,78]. However, for nonlinear plants with unknown structures, it may not be possible to ensure a perfect plant output that follows the output of a reference model by using conventional SAC [21,22]. On the other hand, for nonlinear plants, many methods for control using neural network are proposed. It has been proven that these control methods show excellent performance with nonlinearity [4,39,48,79].

In our proposed method, we discuss SAC for continuous-time system. Hence, we deal with a problem concerning actual realization of SISO and MIMO SAC systems in this chapter. However, for generalization, we discuss it in term of MIMO system, where it can be directly applied to SISO system. The synthesizing of a MIMO controller problem can be stated as follows.

The problem of compensation is compound, by the multiplicity of inputs and outputs, and the interaction (coupling) between the different inputs.

In many multivariable processes, there exists strong interaction (coupling) between inputs and control loops. In such a case, it is important to consider a decoupling control strategy so as to improve performance of the closed-loop systems. When the model parameters are unknown, a feasible approach is to adopt an adaptive decoupling scheme [80]. For a nonlinear plant which has unknown model structure and parameters, the decoupling control problem becomes more complicated, as we need to design a controller that can solve simultaneously the decoupling problem, the unknown model structure and parameters problem, and nonlinearity problem of the plant. For decoupling problem and unknown model structure and parameters problem we consider that SAC is the best approach. And for nonlinearity problem, neural network is considered to be the best approach.

Control methods for nonlinear systems using a combination of SAC and neural networks have been proposed in [21, 22]. However, the methods in [21, 22] have some deficiencies, the theoretical explanations about the convergence and stability analysis are not provided, and the class of nonlinear systems which can be controlled is not defined. We attempt to overcome these deficiencies by providing the convergence and stability analysis, and defining the class of nonlinear systems which can be controlled.

This chapter proposes a method of SAC for nonlinear systems using a neural network for a class of nonlinear systems with BIBO and bounded nonlinearity. The control input is given by the sum of the output of a simple adaptive controller and the output of the neural network. The role of the neural network is to compensate for constructing a linearized system so as to minimize the output error caused by nonlinearities in the controlled system. The role of the simple adaptive controller is to perform the model matching for the linearized system to a given linear reference model. In this chapter, we use a design method using backpropagation training algorithm of simple multilayer feedforward neural network, using the direct neural adaptive control method as mentioned in references [4, 9], in order to design the proposed method. Furthermore, convergence and stability analysis for this proposed method is performed. Finally, the numerical simulations for an SISO system and an MIMO system consists of 2-inputs 2-outputs are executed and the effectiveness of this control system is confirmed.

2.2 Linear SAC

In this section, we briefly describe a linear continuous-time SAC, where the controller is designed to realize a plant output which converges to reference model output.

In a realistic environment, let us consider the following controllable and observable but unknown parameter plant model of order n_p

$$\dot{x}_p(t) = A_p x_p(t) + B_p u(t) \tag{2.1}$$

$$y_p(t) = C_p x_p(t) \tag{2.2}$$

where $x_p(t)$ is the n_pth-order plant state vector, $u(t) \in R^{n_j \times 1}$ is the system input vector, $y_p(t) \in R^{n_j \times 1}$ is the system output vector, and A_p, B_p, C_p are matrices with the appropriate dimensions.

It is necessary for us in the realization of linear SAC to control plant in (2.1), (2.2) to make the following assumption.

Assumption 2-1

(a) Plant in (2.1), (2.2) is almost strictly positive real (ASPR). That is, there exists a constant gain k_e^* such that the transfer function

$$G_p(s) = C_p(sI - A_c)^{-1} B_p \tag{2.3}$$

is SPR (strictly positive real), where $G_p(s)$ is the plant transfer function, and $A_c = A_p + k_e^* B_p C_p$.

(b)

$$det \begin{bmatrix} A_p & B_p \\ C_p & 0 \end{bmatrix} \neq 0$$

Furthermore, let us consider that the plant is required to follow the input-output behaviour of a reference model of the form

$$\dot{x}_m(t) = A_m x_m(t) + B_m u_m(t) \tag{2.4}$$

$$y_m(t) = C_m x_m(t) \tag{2.5}$$

where $x_m(t)$ is the n_mth-order reference model state vector, $u_m(t) \in R^{n_j \times 1}$ is the model input, $y_m(t) \in R^{n_j \times 1}$ is the model output,and A_m, B_m, C_m are matrices with the appropriate dimensions. The reference model can be independent of the controlled plant, and it is permissible to assume $n_m \ll n_p$.

However, the condition in assumption 2-1(a) is not satisfied by most of the real systems. Therefore, to satisfy assumption 2-1(a), let us define the supplementary values of an augmented plant as

$$y_a(t) = y_p(t) + y_s(t) \tag{2.6}$$

$$y_s(t) = D_p(s)u(t) \tag{2.7}$$

$$e_y(t) = y_m(t) - y_a(t) \tag{2.8}$$

where $D_p(s)$ is simple parallel feedforward compensator (PFC)

$$D_p(s) = \frac{D_p}{1+\rho s} \tag{2.9}$$

across the controlled plant to fulfill the condition in Assumption 2-1(a) to guarantee robust stability of SAC system [34–36]. The augmented plant we use here must satisfy the following conditions

1 Plant in (2.6) is ASPR.

2 $y_a(t) = y_p(t) + y_s(t) \cong y_p(t)$ which can be fulfilled by setting the value of D_p to be very small [34, 36].

3 $D_p(s)$ is physically realizable.

The control objective is to achieve the following relation

$$\lim_{t\to\infty} e_y(t) = 0 \tag{2.10}$$

Since the plant is unknown, the actual control input of the plant will be generated by the following adaptive algorithm using the values that can be measured, namely $e_y(t)$, $x_m(t)$ and $u_m(t)$, to get the low-order adaptive controller

$$u(t) = u_p(t) \tag{2.11}$$

$$\begin{aligned} u_p(t) &= K_e(t)e_y(t) + K_x(t)x_m(t) + K_u(t)u_m(t) \\ &= K(t)r(t) \end{aligned} \tag{2.12}$$

where

$$K(t) = [K_e(t) \quad K_x(t) \quad K_u(t)] \tag{2.13}$$

$$r^T(t) = [e_y^T(t) \quad x_m^T(t) \quad u_m^T(t)] \tag{2.14}$$

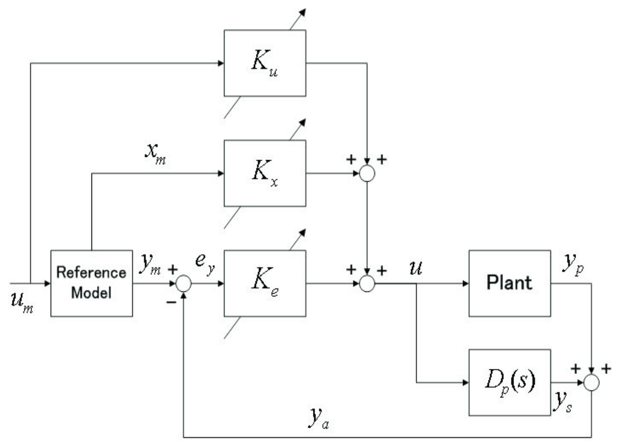

Figure 2.1: Schematic representation of the conventional SAC

and the adaptive gains are obtained as a combination of 'proportional' and 'integral' terms as follows

$$
\begin{aligned}
K(t) &= K_p(t) + K_i(t) && (2.15)\\
K_p(t) &= \left[\begin{array}{ccc} e_y(t)e_y^T(t)T_{p_{e_y}} & e_y(t)x_m^T(t)T_{p_{x_m}} & e_y(t)u_m^T(t)T_{p_{u_m}} \end{array}\right] \\
&= e_y(t)r^T(t)T_p && (2.16)\\
\dot{K}_i(t) &= e_y(t)r^T(t)T_i - \sigma K_i(t) && (2.17)\\
&\quad (T_p = T_p^T > 0, T_i = T_i^T > 0)
\end{aligned}
$$

where the augmented plant error $e_y(t)$ can be reduced to a very small value by increasing T_p and T_i to very large values, and σ is set to a sufficiently small positive value to prevent $K_i(t)$ from diverging [13].

The linear continuous-time SAC is represented in Fig.2.1.

2.3 Nonlinear SAC

When the input-output characteristic of the controlled object is nonlinear, it is not possible to express it as (2.1), (2.2). Then, let the unknown nonlinear plant to be expressed by a system that consists of a linear part and a nonlinear part as

$$
\begin{aligned}
\dot{x}_p(t) &= A_p x_p(t) + B_p u(t) + f_x(x_p(t), u(t)) && (2.18)\\
y_p(t) &= C_p x_p(t) + f_y(x_p(t)) && (2.19)
\end{aligned}
$$

where $x_p(t) \in R^{n_p \times 1}$ is the plant state vector, $u(t) = [u_1(t), u_2(t), \cdots, u_{n_j}(t)]^T \in R^{n_j \times 1}$ is the control input vector, and $y_p(t) = [y_{p_1}(t), y_{p_2}(t), \cdots, y_{p_{n_j}}(t)]^T \in$

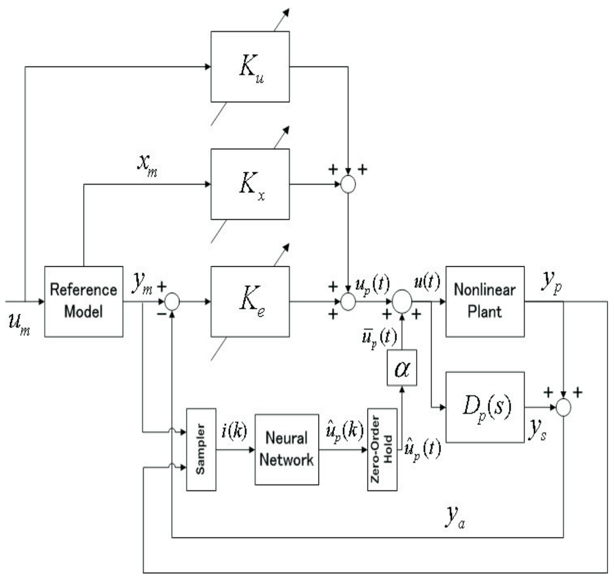

Figure 2.2: Structure of MIMO nonlinear continuous-time SAC system with neural network

$R^{n_j \times 1}$ is the output vector. $f_x(\cdot)$ and $f_y(\cdot)$ are unknown nonlinear function vectors. Then, we assume that the nonlinear plant in (2.18), (2.19) satisfies the following assumption.

Assumption 2-2

(a) The nonlinear plant in (2.18), (2.19) is BIBO, where its linear and nonlinear parts are unknown.

(b) For the system in (2.18), (2.19), there exists an augmented plant which its linear part satisfies assumption 2-1(a). This augmented plant, as in (2.6), is formed by incorporating the system in (2.18), (2.19) with the supplementary values in (2.6)–(2.9) [35].

(c) The nonlinear part of the system in (2.18), (2.19), which is represented by $f_x(\cdot)$ and $f_y(\cdot)$, is bounded.

However, in this case, when the SAC rule in (2.11)–(2.17) is used to control the nonlinear plant in (2.18), (2.19) which satisfies assumption 2-2, the problem of output error will arise [21].

To overcome this problem of output error and to keep the plant output $y_p(t)$ converging to the reference model output $y_m(t)$, the control input can be expressed as

$$u(t) = h(y_m^T(t), y_a^T(t), y_p^T(t), x_p^T(t)) \tag{2.20}$$

according to (2.8), (2.18), (2.19), where $h(\cdot)$ is an unknown nonlinear function vector.

In this chapter, we synthesize the control input $u(t)$ by the following equation

$$u(t) = u_p(t) + \bar{u}_p(t) \tag{2.21}$$

where, $u_p(t) = [u_{p_1}(t), \cdots, u_{p_{n_j}}(t)]^T$ is the control input vector of the simple adaptive controller, as mentioned in (2.12). And $\bar{u}_p(t) = [\bar{u}_{p_1}(t), \cdots, \bar{u}_{p_{n_j}}(t)]^T$ is a control input vector of the neural network given as

$$\begin{aligned} \bar{u}_p(t) &= \alpha \hat{u}_p(t) & (2.22)\\ \hat{u}_p(t) &= f_{zoh}(\hat{u}_p(k)) & (2.23) \end{aligned}$$

where α is a positive constant, $\hat{u}_p(t)$ is a continuous-time output vector of the neural network, $\hat{u}_p(k)$ is a discrete-time output vector of the neural network, and $f_{zoh}(\cdot)$ is a zero-order hold function.

The structure of nonlinear continuous-time SAC system with neural network is shown in Fig.2.2. A sampler is implemented in front of the neural network with appropriate sampling period to obtain discrete-time multi-input of the neural network, and a zero-order hold is implemented to transform the discrete-time output $\hat{u}_p(k)$ of the neural network back to continuous-time output $\hat{u}_p(t)$ as shown in Fig.2.2 and (2.23). For systems having bandwidths of a few Hertz, appropriate sample rates are often in the order of $100Hz$, so that appropriate sample periods are in the order of $0.01sec$ [81].

Consequently, we can assume the discrete-time output $\hat{u}_p(k)$ as follows

$$\begin{aligned} \hat{u}_p(k) &= \hat{h}(y_m^T(k-1), \\ &\qquad y_p^T(k-1), \cdots, y_p^T(k-n)) & (2.24) \end{aligned}$$

where $\hat{h}(\cdot)$ is an unknown nonlinear function vector, and n is the number of past data of outputs of the plant.

Using the above approach, the neural network will be trained. The training is done by adjusting the weights of the neural network until the output error $e(t)$ given as

$$e(t) = y_m(t) - y_p(t) \tag{2.25}$$

satisfies the following relation

$$\lim_{t\to\infty} |e(t)| = \lim_{t\to\infty} |y_m(t) - y_p(t)| \leq \epsilon \tag{2.26}$$

where ϵ is a small positive value.

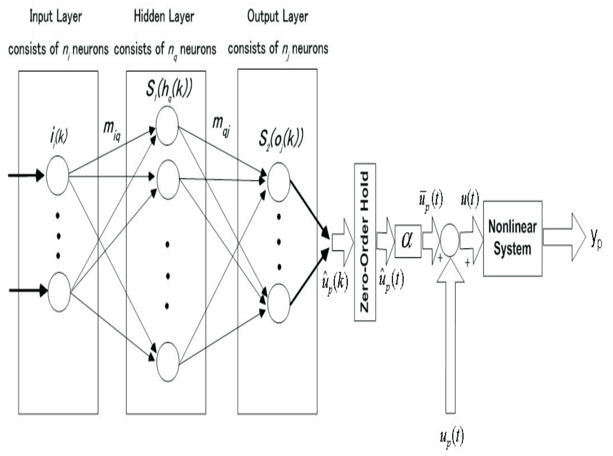

Figure 2.3: System configuration with neural network

2.4 Composition of the Neural Network

Figure 2.3 shows system configuration of the input-output relation for the system with neural network. The neural network consists of three layers: an input layer, an output layer and a hidden layer. Let $i_i(k)$ be the input to the i-th neuron in the input layer ($i = 1, \cdots, n_i$), $h_q(k)$ be the input to the q-th neuron in the hidden layer ($q = 1, \cdots, n_q$), $o_j(k)$ be the input to the j-th neuron in the output layer ($j = 1, \cdots, n_j$), where n_i, n_q, and n_j are the number of neurons in input layer, hidden layer, and output layer, respectively. Furthermore, let m_{iq} be the weights between the input layer and the hidden layer, m_{qj} be the weights between the hidden layer and the output layer.

In Fig.2.3, the control input is given by the sum of the output of the simple adaptive controller and the output of the neural network. The neural network is used to compensate for the nonlinearity of the plant dynamics that is not taken into consideration in the usual SAC. The role of the neural network is to construct a linearized model by minimizing the output error caused by the nonlinearities in the control systems. Refer to (2.24), the input $i(k)$ of the neural network is given as

$$i(k) \quad = \quad [y_m^T(k-1), y_p^T(k-1), \cdots, y_p^T(k-n)]^T. \tag{2.27}$$

Therefore, the nonlinear function of an MIMO system can be approximated by the neural network. Furthermore, values n should be chosen appropriately according to practical nonlinear systems.

2.5 Learning of the Neural Network

From Fig.2.3, we can obtain

$$\begin{aligned} h_q(k) &= \sum_i i_i(k) m_{iq}(k) && (2.28) \\ o_j(k) &= \sum_q S_1(h_q(k)) m_{qj}(k) && (2.29) \\ \hat{u}_{p_j}(k) &= S_2(o_j(k)) && (2.30) \end{aligned}$$

where $S_1(\cdot)$ is a sigmoid function, $S_2(\cdot)$ is a pure linear function, and $j = 1, 2, \cdots, n_j$. The sigmoid function is chosen as

$$S_1(X) = \frac{2}{1 + \exp(-\mu X)} - 1 \qquad (2.31)$$

where $\mu > 0$, and the pure linear function is chosen as

$$S_2(X) = X \qquad (2.32)$$

Consider the case when $S_1(X) = a$. Then the derivative of the sigmoid function $S_1(\cdot)$ and the pure linear function $S_2(\cdot)$ are as follows

$$\begin{aligned} S_1'(X) &= \frac{\mu}{2}(1 - a^2) \\ S_2'(X) &= 1 \end{aligned}$$

The objective of training is to minimize an error function $E(k)$ by taking the error gradient with respect to the parameters or the weight vector $m(k)$, that is to be adapted. The error function is defined as

$$\begin{aligned} E(k) &= \frac{1}{2} e^T(k) e(k) \\ &= \frac{1}{2} \sum_j^{n_j} \left[y_{m_j}(k) - y_{p_j}(k) \right]^2 && (2.33) \end{aligned}$$

where $e(k)$ is a discrete-time form of the output error $e(t)$ in (2.25). Then, the weights are adapted by using

$$\varDelta m(k) = -c \cdot \frac{\partial E(k)}{\partial m(k)} \qquad (2.34)$$

where $c > 0$ is the learning parameter. For the learning process, (2.34) will be expanded as follows

$$\begin{aligned}\mathit{\Delta} m_{qj}(k) &= -c\cdot\frac{\partial E(k)}{\partial y_{p_j}(k)}\cdot\frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_j}(k)}\cdot\frac{\partial \hat{u}_{p_j}(k)}{\partial S_2(o_j(k))}\\ &\quad\cdot\frac{\partial S_2(o_j(k))}{\partial o_j(k)}\cdot\frac{\partial o_j(k)}{\partial m_{qj}(k)} \qquad (2.35)\\ \mathit{\Delta} m_{iq}(k) &= -c\cdot\sum_{j}^{n_j}\frac{\partial E(k)}{\partial y_{p_j}(k)}\cdot\frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_j}(k)}\\ &\quad\cdot\frac{\partial \hat{u}_{p_j}(k)}{\partial S_2(o_j(k))}\cdot\frac{\partial S_2(o_j(k))}{\partial o_j(k)}\\ &\quad\cdot\frac{\partial o_j(k)}{\partial S_1(h_q(k))}\cdot\frac{\partial S_1(h_q(k))}{\partial h_q(k)}\\ &\quad\cdot\frac{\partial h_q(k)}{\partial m_{iq}(k)} \qquad (2.36)\end{aligned}$$

where

$$\begin{aligned}\frac{\partial E(k)}{\partial y_{p_j}(k)} &= -\left[y_{m_j}(k)-y_{p_j}(k)\right]\\ \frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_j}(k)} &= J_{plant_j}\\ \frac{\partial \hat{u}_{p_j}(k)}{\partial S_2(o_j(k))} &= 1\\ \frac{\partial S_2(o_j(k))}{\partial o_j(k)} &= 1\\ \frac{\partial o_j(k)}{\partial m_{qj}(k)} &= S_1(h_q(k))\\ \frac{\partial o_j(k)}{\partial S_1(h_q(k))} &= m_{qj}(k)\\ \frac{\partial S_1(h_q(k))}{\partial h_q(k)} &= \frac{\mu}{2}\left[1-S_1^2(h_q(k))\right]\\ \frac{\partial h_q(k)}{\partial m_{iq}(k)} &= i_i(k)\end{aligned}$$

Furthermore, J_{plant_j} represents the Jacobian of the plant. According to reference [38], this plant Jacobian can be estimated by using a identified parameter and the internal variables of the neural network model in the indirect neural adaptive control. In many cases this J_{plant_j} is clear from physical insight or can be estimated through some experiments, as mentioned and proposed in the references [4,9]. Therefore, considering for holding the fundamental design concept of SAC i.e. without any identifiers, in this chapter we utilize from the direct neural adaptive control method [4,9]

$$J_{plant_j} = \mathrm{SGN}\left(\frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_j}(k)}\right) \qquad (2.37)$$

where SGN($\cdot$) is sign function.

2.6 Convergence and Stability

The stability analysis of SAC for controllable and observable linear plant with disturbances with unknown parameter has been presented in [13], where the plant is as follows

$$\dot{x}_p(t) = A_p x_p(t) + B_p u_p(t) + d_i(t) \tag{2.38}$$

$$y_p(t) = C_p x_p(t) + d_o(t) \tag{2.39}$$

where $d_i(t)$ and $d_o(t)$ represent bounded, unknown, and unmeasurable plant and output disturbances, respectively, and we assume that the plant in (2.38) and (2.39) fulfill the conditions in assumption 2-1. Thus, the following theorem 2-1 given in [13] will hold.

Theorem 2-1: Assume that the linear augmented plant, which is formed by incorporating the plant in (2.38), (2.39) with the supplementary values in (2.6)–(2.9), is ASPR and that the input and output disturbances are bounded. Then, the adaptive control system in (2.12)–(2.17) is globally stable with respect to boundedness. In other words, all values (states, gains, and errors) involved in the control of the linear augmented plant are bounded. Furthermore, the output tracking error $\mathbf{e}_y(t)$ can be directly controlled and thus reduced via the adaptation coefficient T_p.
Proof: The detailed proof of theorem 2-1 is presented in [13].

For the stability analysis of our proposed method we will modify and apply the stability proof of theorem 2-1 presented in [13]. As mentioned in assumption 2-2(b), the PFC in (2.7) and (2.9) is incorporated with the nonlinear system in (2.18) and (2.19) to form the augmented plant, as in (2.6), which its linear part is ASPR. However, for convenience, first it is necessary for the PFC in (2.7) and (2.9) to be transformed into a state-space form as follows

$$\dot{x}_s(t) = A_s x_s(t) + B_s u(t) \tag{2.40}$$

$$y_s(t) = C_s x_s(t) \tag{2.41}$$

then, by applying (2.40),(2.41) to (2.7),(2.18),(2.19), the augmented plant can be described as follows

$$\dot{x}(t) = Ax(t) + Bu_p(t) + \hat{\delta}_i(x(t), u(t), \bar{u}_p(t)) \tag{2.42}$$

$$y_a(t) = Cx(t) + \hat{\delta}_o(x(t)) \tag{2.43}$$

where

$$\begin{aligned} x &= \begin{bmatrix} x_p \\ x_s \end{bmatrix} \in R^{(n_p+1)\times 1}; \quad A = \begin{bmatrix} A_p & 0 \\ 0 & A_s \end{bmatrix}; \quad B = \begin{bmatrix} B_p \\ B_s \end{bmatrix}; \\ C &= \begin{bmatrix} C_p & D_p \end{bmatrix} \end{aligned} \tag{2.44}$$

and $\hat{\delta}_i(x(t), u(t), \bar{u}_p(t))$ and $\hat{\delta}_o(x(t))$ represent the nonlinear part of the augmented plant described as follows

$$\begin{aligned} \hat{\delta}_i(x(t), u(t), \bar{u}_p(t)) &= \begin{bmatrix} f_x(x_p(t), u(t)) \\ 0 \end{bmatrix} \\ &\quad + B\bar{u}_p(t) \end{aligned} \tag{2.45}$$

$$\hat{\delta}_o(x(t)) = f_y(x_p(t)). \tag{2.46}$$

The nonlinear part of the system in (2.42)–(2.43) will be compensated and minimized using the control input of the neural network $\bar{u}_p(t)$, to form a linearized system. The control input of SAC $u_p(t)$ will perform model matching of the linearized system to a given linear reference model. The nonlinearity compensation and minimization process and the linear model matching process will be performed simultaneously. Therefore, it is necessary in our proposed method that the control system is able to keep its stability while performing those processes.

To prove the stability of our proposed method, we start from the stability analysis of the SAC part of our proposed method, where its Lyapunov function and its derivative is a modification of the ones for SAC for linear plant with disturbances in (2.38), (2.39), as presented in [13], by replacing the terms $d_i(t)$ and $d_o(t)$ with $\hat{\delta}_i(x(t), u(t), \bar{u}_p(t))$ and $\hat{\delta}_o(x(t))$, respectively. The Lyapunov function of the SAC part of our proposed method is given as

$$V_{SAC}(t) = V_1(t) + V_2(t) \tag{2.47}$$

where

$$V_1(t) = e_x^T(t) P e_x(t) \tag{2.48}$$

$$V_2(t) = tr\left\{ \left[K_i(t) - \tilde{K}\right] T_i^{-1} \left[K_i(t) - \tilde{K}\right]^T \right\} \tag{2.49}$$

where $tr(\cdot)$ is trace function, and $e_x(t)$ is given as

$$e_x(t) = \hat{x}(t) - x(t) \tag{2.50}$$

where $\hat{x}(t)$ is the ideal target states of the system, and $\tilde{K} = [\tilde{K}_e \quad \tilde{K}_x \quad \tilde{K}_u]$ is the unknown ideal gains of SAC. The derivative of the Lyapunov function in (2.47)–(2.49) will be

$$\dot{V}_{SAC}(t) = \dot{V}_1(t) + \dot{V}_2(t) \tag{2.51}$$

$$
\begin{aligned}
&= -e_x^T(t)Qe_x(t) \\
&-2\sigma tr\left[(K_i(t)-\tilde{K})T_i^{-1}(K_i(t)-\tilde{K})^T\right] \\
&-2e_y^T(t)e_y(t)e_y^T(t)T_{p_{e_y}}e_y(t) \\
&-2e_y^T(t)e_y(t)\left[x_m^T(t)T_{p_{x_m}}x_m(t)\right. \\
&\left.+u_m^T(t)T_{p_{u_m}}u_m(t)\right] \\
&-2\sigma tr\left[(K_i(t)-\tilde{K})T_i^{-1}\tilde{K}^T\right] \\
&-2e_x^T(t)PF(t) \\
&-2\hat{\delta}_o^T(x_p(t))(K_i(t)-\tilde{K})r(t) \\
&-2\hat{\delta}_o^T(x_p(t))e_y(t)e_y^T(t)T_{p_{e_y}}e_y(t) \\
&-2\hat{\delta}_o^T(x_p(t))e_y(t)\left[x_m^T(t)T_{p_{x_m}}x_m(t)\right. \\
&\left.+u_m^T(t)T_{p_{u_m}}u_m(t)\right] \qquad (2.52)
\end{aligned}
$$

where $F(t)$ is given as

$$
\begin{aligned}
F(t) &= E_{Bias}(t) - B\tilde{K}_e\hat{\delta}_o(x(t)) \\
&\quad +\hat{\delta}_i(x(t),u(t),\bar{u}_p(t)) \qquad (2.53)
\end{aligned}
$$

where $E_{Bias}(t)$ is a bias term as explained in [13].

For the derivative of Lyapunov function in (2.51)–(2.53), we can directly apply to it the same method as the one used in [13] to prove the stability of our proposed method if $\hat{\delta}_i(x(t),u(t),\bar{u}_p(t))$ and $\hat{\delta}_o(x(t))$ are bounded. Refering to (2.46) and assumption 2-2(c), $f_y(\cdot)$ is bounded by assumption, then this means that $\hat{\delta}_o(x_p(t))$ is also bounded. However, refering to (2.45) and assumption 2-2(c), eventhough $f_x(\cdot)$ is assumed to be bounded, $\hat{\delta}_i(x(t),u(t),\bar{u}_p(t))$ is bounded if and only if $\bar{u}_p(t)$ is also bounded. Therefore, based on (2.22), (2.23), to prove that $\bar{u}_p(t)$ is bounded, it is necessary to prove the convergence of the neural network. It means that the stability of our proposed method requires the convergence of the neural network part. Furthermore, (2.24)–(2.27), (2.34)–(2.37) show that the parameters of the neural network part are not influenced directly by the SAC part, only influenced indirectly through the augmented plant in (2.42)–(2.43). Thus, the convergence of the neural network part can be proven separately.

To prove the convergence of the neural network part of our proposed method, we refer to the method presented in [41], [42]. The Lyapunov function of the neural network is given as

$$V_{NN}(k) = \frac{1}{2}e^2(k) \qquad (2.54)$$

and the derivative of the Lyapunov function is given as

$$\begin{aligned} \Delta V_{NN}(k) &= V_{NN}(k+1) - V_{NN}(k) \\ &= \frac{1}{2}\left[e^2(k+1) - e^2(k)\right]. \end{aligned} \tag{2.55}$$

By expanding (2.34) as follows

$$\Delta m(k) = c \cdot e(k) \cdot J_{plant} \cdot \frac{\partial o_j(k)}{\partial m(k)} \tag{2.56}$$

then as shown in [41], ΔV_{NN} in (2.55) can be represented as

$$\Delta V_{NN}(k) = \Delta e(k)\left[e(k) + \frac{1}{2}\Delta e(k)\right]$$

$$\begin{aligned} &= \left[\frac{\partial e(k)}{\partial m(k)}\right]^T \cdot c \cdot e(k) \cdot J_{plant} \cdot \frac{\partial o_j(k)}{\partial m(k)} \\ &\cdot \left\{ e(k) + \frac{1}{2}\left[\frac{\partial e(k)}{\partial m(k)}\right]^T \cdot c \cdot e(k) \cdot J_{plant} \cdot \frac{\partial o_j(k)}{\partial m(k)} \right\} \end{aligned} \tag{2.57}$$

where the convergence is guaranteed if the boundary of c is chosen as

$$0 < c < \frac{2}{J^2_{plant,\max} \cdot g^2_{\max}} \tag{2.58}$$

as proven in [41], where $J_{plant,\max}$ is the limit on the plant Jacobian, which refers to (2.37), will be

$$J_{plant,\max} = 1, \tag{2.59}$$

and

$$\begin{aligned} g_{\max} : &= \max_k \|g(k)\| && (2.60) \\ g(k) &= \frac{\partial o_j(k)}{\partial m(k)} && (2.61) \end{aligned}$$

where $\|\cdot\|$ is the usual Euclidean norm in R^n.

Furthermore, from (2.58)–(2.60), we can choose the boundary of the learning parameter for each type of weight of the neural network. For the weights between the hidden layer and the output layer, m_{qj}, the boundary of the learning parameter is chosen as [41]

$$0 < c < \frac{2}{n_q}. \tag{2.62}$$

For the weights between the input layer and the hidden layer, m_{iq}, the boundary of the learning parameter is chosen as

$$0 < c < \frac{2}{n_q} \left[\frac{1}{m_{qj,\max} \cdot i_{i,\max}} \right]^2 \tag{2.63}$$

where

$$\begin{aligned} m_{qj,\max} : &= \max_k \|m_{qj}(k)\| & (2.64) \\ i_{i,\max} &= \max_k \|i_i(k)\| \, . & (2.65) \end{aligned}$$

Proofs for (2.62) and (2.63) : See the Appendices 2A and 2B.

If the learning parameter c is set to be inside the boundaries in (2.62) and (2.63), the convergence of the neural network part of our proposed method can be guaranteed, and $\bar{u}_p(t)$ will be bounded. It means that $\hat{\delta}_i(x(t), u(t), \bar{u}_p(t))$ will be bounded too, and the stability of the SAC part of our proposed method can also be guaranteed. Then, in general, the stability of our proposed method, where the nonlinearity compensation and minimization process and the linear model matching process are performed simultaneously, can be guaranteed.

Furthermore, the convergence of the neural network part of our proposed method means that the error function $E(k)$ in (2.33) is minimized, and the output error $e(t)$ in (2.25) satisfies the relation in (2.26). This shows that the nonlinearity of the system in (2.18)–(2.19) has been compensated for and minimized using the control input of the neural network $\bar{u}_p(t)$.

2.7 Computer Simulation

As for the nonlinear systems, two cases are considered, one of SISO and one of MIMO. Then, for each of the SISO and MIMO nonlinear systems, we compare the simulation results of using only SAC, only neural network, and our proposed method.

2.7.1 SISO System

Let us consider the SISO nonlinear system, which is a modification from the one in [4], described by

$$\begin{aligned}
\begin{bmatrix} \dot{x_{p1}} \\ \dot{x_{p2}} \end{bmatrix} &= \begin{bmatrix} x_{p2} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\
&\quad + \begin{bmatrix} 0 \\ 2\overset{10}{\underset{-10}{f_{sat}}}(x_{p1}\sin(x_{p1})) \end{bmatrix} \\
y_p &= x_{p1} + \sin(x_{p1})
\end{aligned}$$

where $\overset{n_{upper}}{\underset{n_{lower}}{f_{sat}}}(\cdot)$ is a saturation function with a lower limit at n_{lower} and an upper limit at n_{upper}. Then, for the simulations using only SAC and our proposed method, the parameters are set as

$$\begin{aligned}
T_p &= diag(5 \times 10^3, 5 \times 10^3, 5 \times 10^3) \quad \text{(in (2.16))}, \\
T_i &= diag(5 \times 10^4, 5 \times 10^4, 5 \times 10^4) \quad \text{(in (2.17))}, \\
\sigma &= 1 \quad \text{(in (2.17))}, \\
\alpha &= 1 \quad \text{(in (2.22))}, \\
\mu &= 2 \quad \text{(in (2.31))}, \\
c &= 0.001 \quad \text{(in (2.34))}, \\
D_p &= 0.001 \quad \text{(in (2.9))}, \\
\rho &= 1 \quad \text{(in (2.9))}
\end{aligned}$$

and PFC

$$D_p(s) = \frac{D_p}{1+\rho s} = \frac{0.001}{1+s}$$

is fixed to guarantee that assumption 2-2(b) is satisfied. For these simulations of SISO nonlinear system, we estimate the value of J_{plant_j} by previously doing some experiments to the nonlinear system. From those experiments we get

$$J_{plant_1} = -1.$$

For the simulation using only neural network, the parameters are set as

$$\begin{aligned}
\alpha &= 1 \quad \text{(in (2.22))}, \\
\mu &= 2 \quad \text{(in (2.31))}, \\
c &= 0.0000001 \quad \text{(in (2.34))}, \\
J_{plant_1} &= 1.
\end{aligned}$$

For all simulations of SISO nonlinear system, we assume a first-order reference model with parameters

$$A_m = -10, \quad B_m = 10, \quad C_m = 1.$$

The selection of the first-order models here is to emphasize the fact that low-order models do not affect the ability of the adaptive control system.

Figure 2.4 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using only SAC. The result in Fig.2.4 shows that the error between $y_p(t)$ and $y_m(t)$ is large.

Figure 2.5 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using only neural network, where the number of neurons in the input layer is 2, in the hidden layer is 5, and in the output layer is 1. The input $i(k)$ of the neural network is given as

$$i(k) \quad = \quad [y_m(k-1), y_p(k-1)]^T.$$

Furthermore, a sampling period $0.01sec$ is selected to obtain the values of $i(k)$ from $[y_m(t), y_p(t)]$, where $i(k)$ denotes $i(t)$ at $t = kT$. The result in Fig.2.5 shows that the error between $y_p(t)$ and $y_m(t)$ is large.

Figure 2.6 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using SAC and neural network simultaneously. Here, as in the simulation using only neural network, the number of neurons in the input layer is 2, in the hidden layer is 5, and in the output layer is 1. Also the same as in the simulation using only neural network, the input $i(k)$ of the neural network is given as

$$i(k) \quad = \quad [y_m(k-1), y_p(k-1)]^T$$

where a sampling period $0.01sec$ is again selected to obtain the values of $i(k)$ from $[y_m(t), y_p(t)]$.

It can be seen that the error of the system has been reduced, and the plant output $y_p(t)$ can follow very closely the desired output $y_m(t)$.

2.7.2 MIMO System

Let us consider the MIMO two-input two-output nonlinear system, which is a modification from the one in [82], described by

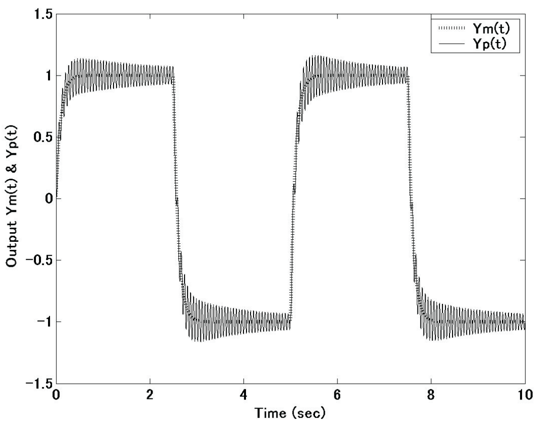

Figure 2.4: $y_m(t)$ and $y_p(t)$ using only SAC (SISO system)

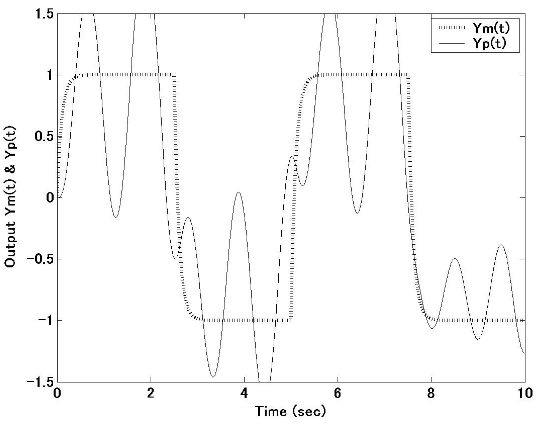

Figure 2.5: $y_m(t)$ and $y_p(t)$ using only neural network (SISO system)

$$\begin{aligned}
\begin{bmatrix} \dot{x}_{p_1} \\ \dot{x}_{p_2} \\ \dot{x}_{p_3} \\ \dot{x}_{p_4} \end{bmatrix} &= \begin{bmatrix} -x_{p_1}+x_{p_4} \\ x_{p_2} \\ -x_{p_2}-x_{p_3} \\ x_{p_3} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \\
&\quad + \begin{bmatrix} 0 \\ \underset{-10}{\overset{10}{f_{sat}}}(x_{p_1}x_{p_3}) + \underset{-10}{\overset{10}{f_{sat}}}(2x_{p_3}u_1) \\ \underset{-10}{\overset{10}{f_{sat}}}(x_{p_1}^2) + \underset{-10}{\overset{10}{f_{sat}}}(2x_{p_3}u_1) \\ 0 \end{bmatrix} \\
\begin{bmatrix} y_{p_1} \\ y_{p_2} \end{bmatrix} &= \begin{bmatrix} x_{p_2}-x_{p_3} \\ x_{p_4} \end{bmatrix} + \begin{bmatrix} \underset{-10}{\overset{10}{f_{sat}}}(x_{p_1}x_{p_3}) \\ \underset{-10}{\overset{10}{f_{sat}}}(x_{p_1}^2) \end{bmatrix}
\end{aligned}$$

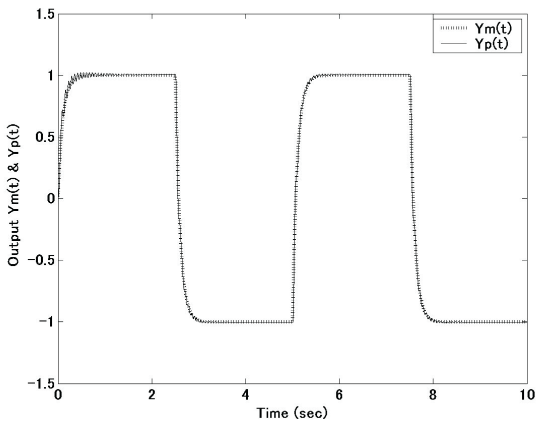

Figure 2.6: $y_m(t)$ and $y_p(t)$ using SAC and neural network simultaneously (SISO system)

where $\overset{n_{upper}}{\underset{n_{lower}}{f_{sat}}}(\cdot)$ is a saturation function with a lower limit at n_{lower} and an upper limit at n_{upper}. Then, for the simulations using only SAC and our proposed method, for SAC and our proposed method, the parameters are set as

$$\begin{aligned}
T_p &= diag(1.7\times10^5, 1.7\times10^5, 1.7\times10^5, \\
&\quad 1.7\times10^5, 1.7\times10^5, 1.7\times10^5) \quad \text{(in (2.16))}, \\
T_i &= diag(1.7\times10^6, 1.7\times10^6, 1.7\times10^6, \\
&\quad 1.7\times10^6, 1.7\times10^6, 1.7\times10^6) \quad \text{(in (2.17))}, \\
\sigma &= 0.1 \quad \text{(in (2.17))}, \\
\alpha &= 110 \quad \text{(in (2.22))}, \\
\mu &= 2 \quad \text{(in (2.31))}, \\
c &= 0.01 \quad \text{(in (2.34))}, \\
D_p &= diag(0.002, 0.002) \quad \text{(in (2.9))}, \\
\rho &= 1 \quad \text{(in (2.9))}
\end{aligned}$$

and PFC

$$D_p(s) = \frac{D_p}{1+\rho s} = \begin{bmatrix} \frac{0.002}{1+s} & 0 \\ 0 & \frac{0.002}{1+s} \end{bmatrix}$$

is fixed to guarantee that assumption 2-2(b) is satisfied. For these simulations of MIMO nonlinear system, we estimate the values of J_{plant_j} by previously doing some experiments to the nonlinear system. From those experiments we get

$$J_{plant_1} = +1, \qquad J_{plant_2} = +1.$$

For the simulation using only neural network, the parameters are set as

$$\begin{aligned}\alpha &= 0.0001 \quad (\text{in } (2.22)),\\ \mu &= 2 \quad (\text{in } (2.31)),\\ c &= 0.01 \quad (\text{in } (2.34)),\\ J_{plant_1} &= +1, \qquad J_{plant_2} = +1.\end{aligned}$$

For all simulations of MIMO nonlinear system, we assume first-order reference models with parameters

$$\begin{aligned}A_{m_1} = -10, \quad B_{m_1} = 10, \quad C_{m_1} = 1,\\ A_{m_2} = -10, \quad B_{m_2} = 10, \quad C_{m_2} = 1.\end{aligned}$$

The selection of the first-order models here is to emphasize the fact that low-order models do not affect the ability of the adaptive control system.

Figure 2.7 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using only SAC. The result in Fig.2.7 shows that the error between $y_p(t)$ and $y_m(t)$ is large.

Figure 2.8 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using only neural network, where the number of neurons in the input layer is 8, in the hidden layer is 5, and in the output layer is 2. The input $i(k)$ of the neural network is given as

$$i(k) \;\; = \;\; [y_m^T(k-1), y_p^T(k-1), y_p^T(k-2), y_p^T(k-3)]^T.$$

Furthermore, a sampling period $0.01sec$ is selected to obtain the values of $i(k)$ from $[y_m(t), y_p(t)]$, where $i(k)$ denotes $i(t)$ at $t = kT$. The result in Fig.2.8 shows that the error between $y_p(t)$ and $y_m(t)$ is large.

Figure 2.9 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using SAC and neural network simultaneously, where the number of neurons in the input layer is 8, in the hidden layer is 5, and in the output layer is 2. Also the same as in the simulation using only neural network, the input $i(k)$ of the neural network is given as

$$i(k) \;\; = \;\; [y_m^T(k-1), y_p^T(k-1), y_p^T(k-2), y_p^T(k-3)]^T$$

where a sampling period $0.01sec$ is selected to obtain the values of $i(k)$ from $[y_m^T(t), y_p^T(t)]$.

It can be seen that error of the system has been reduced, and the plant output $y_p(t)$ can follow very closely the desired output $y_m(t)$.

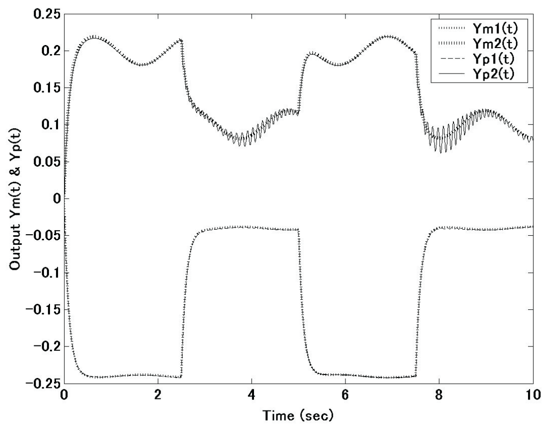

Figure 2.7: $y_m(t)$ and $y_p(t)$ using only SAC (MIMO system)

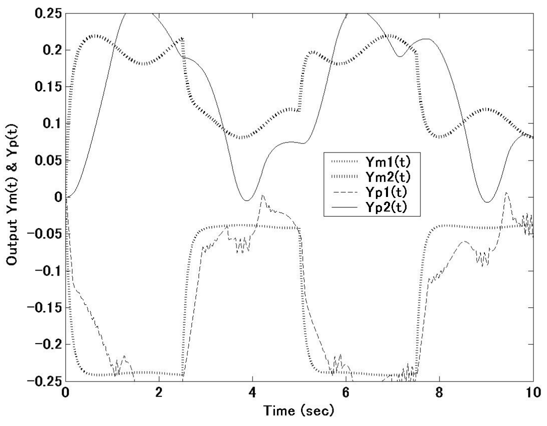

Figure 2.8: $y_m(t)$ and $y_p(t)$ using only neural network (MIMO system)

2.8 Conclusions

This chapter proposed a method of SAC using a neural network for a class of nonlinear systems with BIBO and bounded nonlinearity. The control input was given by the sum of the output of the simple adaptive controller and the output of the neural network. The neural network was used to compensate for the nonlinearity of the plant dynamics that is not taken into consideration in the usual SAC. The role of the neural network was to construct a linearized model by minimizing the output error caused by the nonlinearities in the control systems. Furthermore, the convergence and stability analysis of the proposed method has been performed, and it showed the boundary where the convergence and stability of the proposed method could be guaranteed. Finally

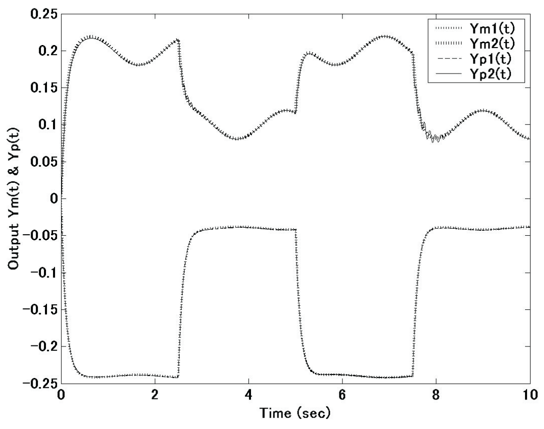

Figure 2.9: $y_m(t)$ and $y_p(t)$ using SAC and neural network simultaneously (MIMO system)

the effectiveness of the proposed method was confirmed through computer simulation, where it has been shown that the plant output $y_p(t)$ can converge to the desired output $y_m(t)$ after learning by the neural network.

Appendix 2A

Proof of (2.62)

Refering to section 2.5, for the weights between the hidden layer and the output layer, (2.61) will become

$$g(k) = \frac{\partial o_j(k)}{\partial m_{qj}(k)} = S_1(h_q(k))$$

where

$$\begin{aligned} S_1(h_q(k)) = [&S_1(h_1(k)), S_1(h_2(k)), \\ &\cdots, S_1(h_{n_q}(k))]^T. \end{aligned}$$

Since $|S_1(h_q(k))| < 1,\ q = 1, 2, \cdots, n_q$, by the definition of the usual Euclidean norm in R^{n_q}, $\|g(k)\| < \sqrt{n_q}$ and $g_{\max}^2 = n_q$. Then from (2.58),(2.59), (2.62) follows.

Appendix 2B

Proof of (2.63)

Refering to section 2.5, for the weights between the input layer and the hidden layer, (2.61) will become

$$\begin{aligned} g(k) &= \frac{\partial o_j(k)}{\partial m_{iq}(k)} \\ &= \frac{\partial o_j(k)}{\partial S_1(h_q(k))} \cdot \frac{\partial S_1(h_q(k))}{\partial h_q(k)} \cdot \frac{\partial h_q(k)}{\partial m_{iq}(k)} \\ &= m_{qj}(k) \cdot S_1'(h_q(k)) \cdot i_i(k) \end{aligned}$$

Since $0 < S_1'(h_q(k)) < 1,\ q = 1, 2, \cdots, n_q$, by the definition of the usual Euclidean norm in R^{n_q}, $\|S_1'(h_q(k))\| < \sqrt{n_q}$, then

$$\begin{aligned} \|g(k)\| &\leq \|m_{qj}(k)\| \left\|S_1'(h_q(k))\right\| \|i_i(k)\| \\ &\leq \sqrt{n_q}\, \|m_{qj}(k)\| \, \|i_i(k)\| \, . \end{aligned}$$

Then from (2.58)–(2.60), (2.63) follows.

Chapter 3

SAC Using A Neural Network for Magnetic Levitation Systems

3.1 Introduction

Magnetic levitation systems have practical importance in many engineering systems such as frictionless bearings for inertial instruments, vibration isolation tables, and high-speed trains [83]. However, designing a controller for the magnetic levitation system is very challenging, because of its strong nonlinearities due to the natural properties of magnetic fields. Thus, it is very difficult to design feedback controllers using classical methods to achieve the desired stability and performance.

On the other hand, adaptive control methods have been developed as an attempt to overcome difficulties connected with the unknown system structures and critical parameter values as well as the changing control regimes [2,5,7,14,17]. However, most self-tuning and adaptive control algorithms usually use estimators or identifiers of almost the same order as the controlled plant. Since the dimension of plants in the real world may be very large or unknown, the implementation of adaptive control procedures may be very difficult.

The SAC procedure was developed by Sobel et al. [19,20] as an attempt to simplify the adaptive controllers, since no identifiers or observers are needed in the feedback loop [34]. Furthermore, the reference model is allowed to be of a very low order compared with the controlled plant. For linear plants with unknown structures, SAC is an important class of adaptive control schemes [13, 34, 36]. However, for nonlinear plants with unknown structures, it may not be possible to ensure a perfect plant output that follows the output of a reference model by SAC [21,22,84].

For nonlinear plants, many methods for control using a neural network have been proposed. It has been proven that these control methods show excellent performance dealing with nonlinearities [4,9,21,22,79,84].

Our focus in this chapter is in designing adaptive controllers for SISO and MIMO configurations of an experimental magnetic levitation system using the method of SAC with the neural network proposed in chapter 2 and [84]. The SISO magnetic levitation system using only one magnet has two configurations, a configuration using a repulsive force from the lower coil to levitate the lower magnet and another configuration using an attractive force via the upper coil to levitate the upper magnet. The MIMO configuration of the magnetic levitation system basically uses the two magnets, the lower magnet and the upper magnet, simultaneously by applying a repulsive force from the lower coil to levitate the lower magnet, and an attractive force via the upper coil to levitate the upper magnet [85].

In principle, the problem of synthesizing an MIMO controller can be derived from the SISO case. In many multivariable processes, there is a strong interaction (coupling) between inputs and control loops. In such a case, it is important to consider a decoupling control strategy to improve the performance of closed-loop systems. When the model parameters are unknown, a feasible approach is to adopt an adaptive decoupling scheme [80]. For a nonlinear plant that has an unknown model structure and parameters, the decoupling control problem becomes more complicated, because we need to design a controller that can simultaneously solve the decoupling problem, the unknown model structure and parameter problem, and the nonlinearity problem of the plant. For the decoupling and the unknown model structure and parameter problems, we consider SAC to be the best approach, and for the nonlinearity problem, a neural network is considered to be the best approach.

In this chapter, we present a control method for the SISO and MIMO configurations of a nonlinear magnetic levitation system using the method of SAC with a neural network. The control method is implemented using a computer; therefore, it is necessary to apply the control method in the discrete-time domain. The control input is given by the sum of the output of SAC and that of the neural network. The role of the neural network is to compensate for the nonlinearity of the plant by constructing a linearized model so as to minimize the output error caused by the nonlinearities in the control system. The role of SAC is to perform model matching for a linear system with unknown structures to a given linear reference model. In this chapter, as in chapter 2 and [84], we use a design method using the backpropagation learning algorithm of a simple multilayer feedforward neural network, using a direct neural adaptive control method [4,9], to design SAC for the nonlinear magnetic levitation system. Furthermore, in this chapter, the magnetic levitation system is set to satisfy the assumptions required by chapter 2 and [84]; thus, the stability analysis in chapter 2 and [84] can be applied to it. Finally, experiments are executed and the effectiveness of this proposed control method is confirmed.

3.2 Magnetic Levitation System

The magnetic levitation system, shown in Figs. 3.1 and 3.2, consists of lower and upper drive coils that produce a magnetic field in response to a DC current, as described in [85]. The magnet travels along a precision ground-glass guide rod. The magnet is of an ultra-high field strength rare-earth type and is designed to provide large levitated displacements to clearly demonstrate the principles of levitation and motion control.

Two laser-based sensors measure the magnet positions. The lower sensor is typically used to measure the position of a given magnet near the lower coil, and the upper sensor is used near the upper coil. This sensor design utilizes light amplitude measurement and includes special circuitry to desensitize the signal to stray ambient light and thermal fluctuations.

The lower magnet is levitated through a repulsive magnetic force by energizing the lower coil. As the current in the coil increases, the field strength increases and the levitated magnet height is increased. For the upper coil, the levitating force is attractive. By separately using a repulsive force from the lower coil to levitate a single magnet, and an attractive force via the upper coil, two SISO configurations are produced, where the SISO configuration using the upper coil is more difficult to control than the SISO configuration using the lower coil [85]. Furthermore, by simultaneously using a repulsive force from the lower coil to levitate the lower magnet, and an attractive force via the upper coil to levitate the upper magnet, an MIMO configuration is created. Free-body diagrams of the suspended magnets of the magnetic levitation system are shown in Figs. 3.3 and 3.4. The magnets are acted on by forces from the lower and upper coils, from gravity, and from friction.

First, from Fig. 3.3, the dynamic model of the SISO magnetic levitation system can be derived. From [85], for the configuration using the lower coil, we have

$$m\ddot{y}_1 + c_1\dot{y}_1 = F_{u_{11}} - mg \tag{3.1}$$

and for the configuration using the upper coil, we have

$$m\ddot{y}_2 + c_2\dot{y}_2 = F_{u_{22}} - mg \tag{3.2}$$

Then, from Fig. 3.4, we derive the dynamic model of the MIMO magnetic levitation system. From [85], for the lower magnet, we have

$$m\ddot{y}_1 = F_{u_{11}} - F_{u_{21}} - F_{m_{12}} - c_1\dot{y}_1 - mg \tag{3.3}$$

and for the upper magnet, we have

$$m\ddot{y}_2 = F_{m_{12}} + F_{u_{22}} - F_{u_{12}} - c_2\dot{y}_2 - mg \tag{3.4}$$

where it is assumed that $c_1 = c_2$, and $F_{u_{11}}$, $F_{u_{12}}$, $F_{u_{22}}$, $F_{u_{21}}$, and $F_{m_{12}}$ are described as

$$\begin{aligned}
F_{u_{11}} &= \frac{i_1}{a(y_1+b)^N} &\quad (3.5)\\
F_{u_{12}} &= \frac{i_1}{a(y_c+y_2+b)^N} &\quad (3.6)\\
F_{u_{22}} &= \frac{i_2}{a(-y_2+b)^N} &\quad (3.7)\\
F_{u_{21}} &= \frac{i_2}{a(y_c-y_1+b)^N} &\quad (3.8)\\
F_{m_{12}} &= \frac{2.69}{(y_{12}+4.2)^N} &\quad (3.9)
\end{aligned}$$

where

$$y_{12} = y_c + y_2 - y_1 \tag{3.10}$$

y_1 is the position of the lower magnet, y_2 is the position of the upper magnet, $y_c = 13\ cm$ is the movement range of the magnets between the lower coil and the upper coil, and i_1 and i_2 are the coil currents in the lower and upper coils, respectively. However, in the control algorithm, the control input may be a digital word, voltage, or current, and is presumed to be linearly proportional to the coil current [85]. The parameter values of (3.1)–(3.9) are shown in Table 3.1.

For the upper magnet in the SISO configuration using the upper coil and in the MIMO configuration, we limit the movement of the magnet to the ranges of 1.1 cm and 0.8 cm, respectively, from the upper coil by using a clamp. Thus, the movement of the upper magnet will always be inside the attracting range of the upper coil.

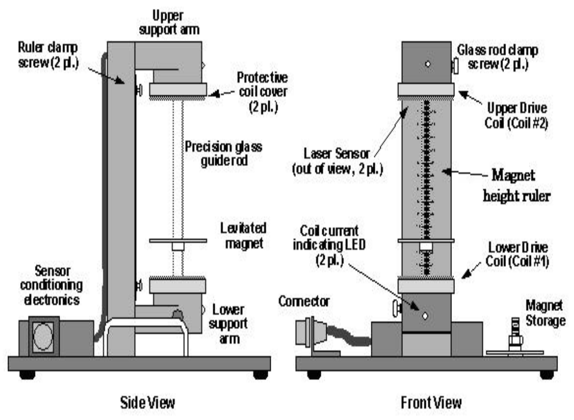
Upper support arm
Ruler clamp screw (2 pl.)
Protective coil cover (2 pl.)
Precision glass guide rod
Levitated magnet
Sensor conditioning electronics
Lower support arm
Side View
Glass rod clamp screw (2 pl.)
Upper Drive Coil (Coil #2)
Laser Sensor (out of view, 2 pl.)
Magnet height ruler
Coil current indicating LED (2 pl.)
Lower Drive Coil (Coil #1)
Connector
Magnet Storage
Front View

Figure 3.1: SISO magnetic levitation system apparatus [85]

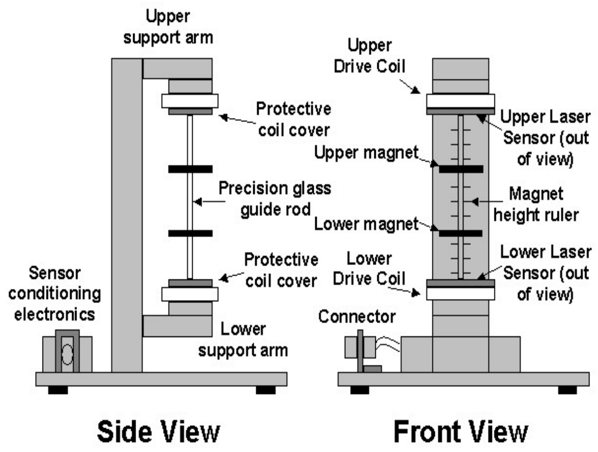

Figure 3.2: MIMO magnetic levitation system apparatus [85]

3.3 Discrete-Time Linear SAC

The SAC method in section 2.2 will be implemented in discrete-time terms. Thus, the control terms of the continuous-time SAC in (2.11)–(2.17) are transformed into discrete-time terms described as

$$u(t) = f_{zoh}(u_p(k)) \tag{3.11}$$

$$\begin{aligned} u_p(k+1) &= K_e(k)e_y(k) + K_x(k)x_m(k) + K_u(k)u_m(k) \\ &= K(k)r(k) \end{aligned} \tag{3.12}$$

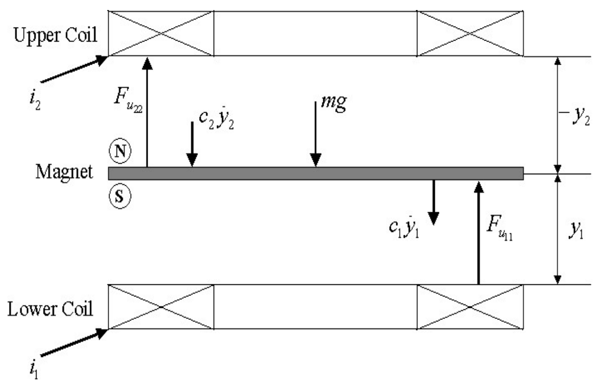

Figure 3.3: Free-body diagram and dynamic configuration of the SISO magnetic levitation system [85]

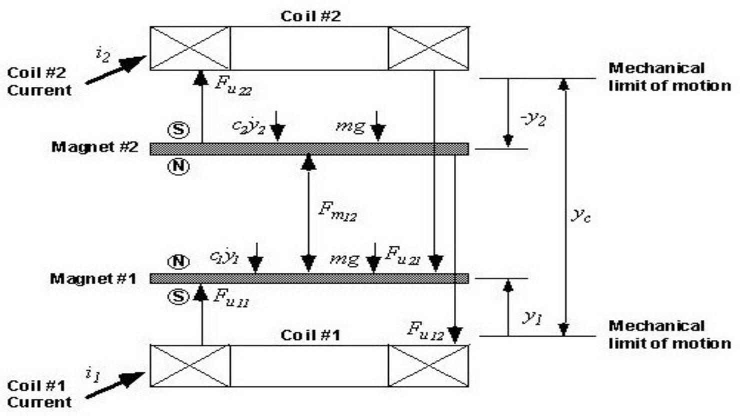

Figure 3.4: Free-body diagram and dynamic configuration of the MIMO magnetic levitation system [85]

where $u(t)$ is the continuous-time plant input vector, $u_p(k)$ is the discrete-time SAC output vector, $f_{zoh}(\cdot)$ is the zero-order hold function, and

$$K(k) = [K_e(k) \quad K_x(k) \quad K_u(k)] \tag{3.13}$$
$$r^T(k) = [e_y^T(k) \quad x_m^T(k) \quad u_m^T(k)] \tag{3.14}$$

where the adaptive gains are obtained as a combination of 'proportional' and 'integral' terms as follows

$$K(k) = K_p(k) + K_i(k) \tag{3.15}$$
$$K_p(k) = e_y(k)r^T(k)T_p \tag{3.16}$$
$$K_i(k) = e_y(k)r^T(k)T_i + \sigma' K_i(k-1) \tag{3.17}$$
$$(\sigma' = exp^{-\Delta T\sigma})$$

where exp^n is the natural exponent of n and ΔT is the sampling period.

Table 3.1: Parameter values of the magnetic levitation system [85]

Parameter	**Description**	**Values**
m	mass of the levitated magnet	0.12 kg
g	gravity	9.8 m/s^2
a	constant	1.65
b	constant	6.2
N	constant	3–4.5
c_1, c_2	friction coefficient	0–10

In (3.12)–(3.17), $e_y(k)$ is the discrete-time term of $e_y(t)$ in (2.8). The supplementary values of the augmented plant in (2.6)–(2.9) are described in discrete-time terms as

$$\begin{aligned} y_a(k) &= y_p(k) + y_s(k) & (3.18)\\ y_s(k) &= D_p(z)u(k) \\ &= exp^{-\frac{\Delta T}{\rho}} y_s(k-1) + \frac{D_p}{\rho}u(k) & (3.19)\\ e_y(k) &= y_m(k) - y_a(k) & (3.20) \end{aligned}$$

The control objective is to achieve the following relation

$$\lim_{k\to\infty} e_y(k) = 0 \tag{3.21}$$

Furthermore, the discrete-time terms of the reference model in (2.4) and (2.5) are given as

$$\begin{aligned} x_m(k+1) &= A_m x_m(k) + B_m u_m(k) & (3.22)\\ y_m(k) &= C_m x_m(k) & (3.23) \end{aligned}$$

where $x_m(k)$ is the n_mth-order state vector, $u_m(k) \in R^{n_j\times 1}$ is the input, $y_m(k) \in R^{n_j\times 1}$ is the output of the discrete-time reference model, and A_m, B_m, and C_m are matrices with appropriate dimensions. The reference model can be independent of the controlled plant, and we can safely assume that $n_m \ll n_p$.

The linear discrete-time SAC is schematically represented in Fig. 3.5. Using an appropriate sampling period ΔT, a sampler is implemented after the plant to obtain the discrete-time plant output $y_p(k)$ to produce $e_y(k)$, as in (3.14) and (3.18)–(3.20), as a part of the input of the discrete-time SAC. Then, a zero-order hold is implemented to transform the discrete-time SAC output vector $u_p(k)$ back to the continuous-time plant input vector $u(t)$, as shown in

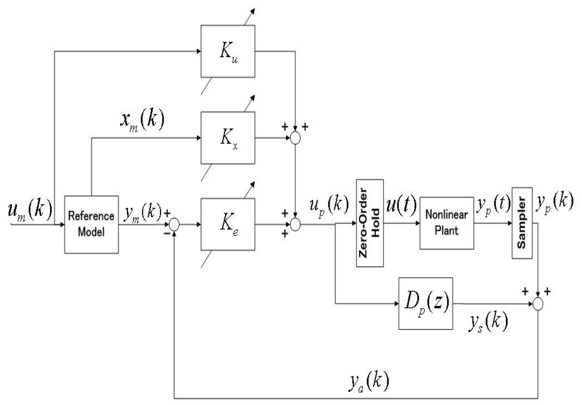

Figure 3.5: Schematic representation of the conventional discrete-time SAC

Fig. 3.5 and (3.11). For systems having bandwidths of up to $10Hz$, appropriate sample rates are often in the order of $100Hz$, so that appropriate sample periods are in the order of $0.01sec$ [81].

3.4 Nonlinear Discrete-Time SAC with Neural Network

When the input-output characteristic of the controlled object is nonlinear, as in the magnetic levitation system, it is not possible to express it as (2.1) and (2.2) [84]. Then, we express the unknown BIBO nonlinear plant as a system that consists of a linear part and a nonlinear part as in (2.18) and (2.19). Then, following chapter 2 and [84], we assume that the nonlinear plant satisfying (2.18) and (2.19) satisfies the assumptions 2-2(a), 2-2(c) and the following assumption.

Assumption 3-1: For the system (2.18) and (2.19), there exists an augmented plant with its linear part satisfying assumption 2-1(a). This augmented plant, as in (3.18), is formed by incorporating the system (2.18) and (2.19) through a sampler with the discrete-time supplementary values given by (3.18) and (3.19).

However, in this case, when the SAC rules in (2.11)–(2.11) and (3.11)–(3.17) are used to control the nonlinear plant (2.18) and (2.19), which also satisfies assumptions 2-2(a), 2-2(c) and 3-1, the problem of an output error will arise [21, 22, 84].

To overcome the problem of output error and to ensure that the plant output $y_p(k)$ converges to the reference model output $y_m(k)$, we adapt and apply the method in chapter 2 and [84]. Thus, the control input is synthesized as

$$\begin{aligned} u(t) &= f_{zoh}(u(k)) \qquad (3.24) \\ u(k) &= u_p(k) + \bar{u}_p(k) \qquad (3.25) \end{aligned}$$

where $u(t)$ is the continuous-time control input vector of the nonlinear plant, $u(k)$ is the discrete-time control input vector of the nonlinear plant, $u_p(k) = [u_{p_1}(k), \cdots, u_{p_{n_j}}(k)]^T$ is the control input vector of the simple adaptive controller, as mentioned in (3.12), and $\bar{u}_p(k) = [\bar{u}_{p_1}(k), \cdots, \bar{u}_{p_{n_j}}(k)]^T$ is a control input vector of the neural network given as

$$\bar{u}_p(k) = \alpha \hat{u}_p(k) \qquad (3.26)$$

where α is a positive constant, and $\hat{u}_p(k) = [\hat{u}_{p_1}(k), \cdots, \hat{u}_{p_{n_j}}(k)]^T$ is the discrete-time output vector of the neural network.

The structure of the nonlinear discrete-time SAC system with a neural network is shown in Fig. 3.6. Using an appropriate sampling period ΔT, a sampler is implemented after the nonlinear plant to obtain the discrete-time plant output $y_p(k)$ to be used as an input vector to the discrete-time controller. Then, a zero-order hold is implemented to transform the discrete-time control input vector $u(k)$ of the plant back to the continuous-time input vector $u(t)$, as shown in Fig. 3.6 and (3.24). Consequently, we can assume that the discrete-time output $\hat{u}_p(k)$ is as (2.24).

Using the above approach, the training of a neural network is performed. The training is carried out by adjusting the weights of the neural network until the output error $e(k)$, given as

$$e(k) = y_m(k) - y_p(k) \qquad (3.27)$$

satisfies the following relation:

$$\lim_{k\to\infty} |e(k)| = \lim_{t\to\infty} |y_m(k) - y_p(k)| \leq \epsilon \qquad (3.28)$$

where ϵ is a small positive value.

The composition and training process of the neural network is performed using the method given in sections 2.4 and 2.5.

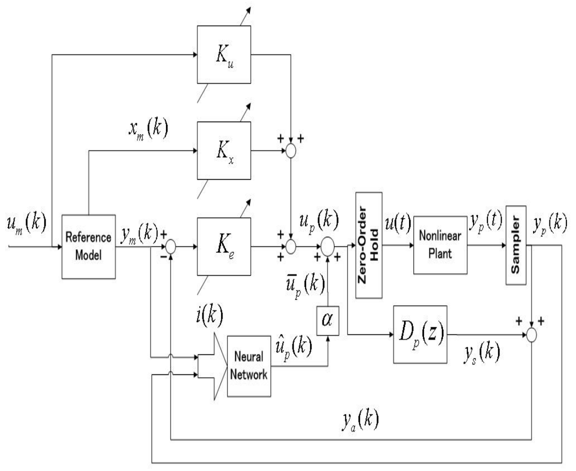

Figure 3.6: Structure of a nonlinear discrete-time SAC system with a neural network

3.5 Control Configuration of the Magnetic Levitation System

A diagram of the control configuration of the magnetic levitation system we use in our experiments is shown in Fig. 3.7. It consists of three subsystems [85]. The first subsystem is the electromechanical plant, which has been explained in section 3.2.

The second subsystem is the control box, which contains the digital-signal-processor (DSP)-based real-time controller, servo/actuator interfaces, servo amplifiers, and auxiliary power supplies [85]. The DSP is based on the M56000 processor family, which is capable of executing control laws at high sampling rates allowing the implementation to be modeled very close to continuous time. The scale of the sampling period ΔT highly depends on this high-sampling-rate capability of the DSP. Four 16-bit analog-to-digital (ADC) converters are used as samplers to digitize the laser sensor signals.

The third subsystem is the ECP Executive program, which runs on a PC under Windows 2000^{TM}. This graphical-user-interface (GUI)-based program is the user's interface to the system and supports the controller specification, trajectory definition, data acquisition, plotting, and system execution commands [85]. We discuss the implementation and testing of the conventional SAC method in section 3.3 and our proposed method in section 3.4 to control the magnetic levitation system using the ECP Executive program.

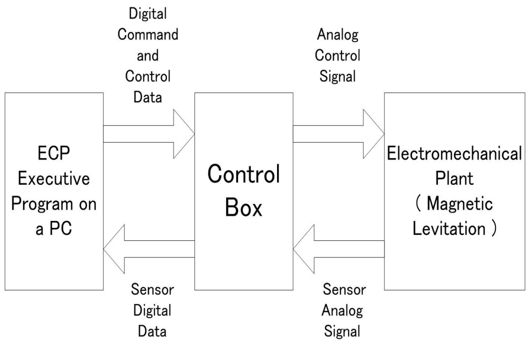
Digital Command and Control Data
Analog Control Signal
ECP Executive Program on a PC
Control Box
Electromechanical Plant (Magnetic Levitation)
Sensor Digital Data
Sensor Analog Signal

Figure 3.7: Control configuration of the magnetic levitation system

3.6 Convergence and Stability

We will apply the convergence and stability conditions in chapter 2 and [84] for this chapter. Therefore, as a first step, we are required to choose the sampling period ΔT to be the minimum value that can be supported by the real-time controller, so that the discrete-time SAC terms of our method closely approximate the continuous-time SAC terms of the method in section 2.2 and [84].

The next step is to set the magnetic levitation system for the SISO and MIMO configurations to satisfy the conditions in assumptions 2-2(a), 2-2(c) and 3-1. The SISO configuration of the magnetic levitation system using the lower coil is naturally stable and bounded, since the movement of the magnet is always inside the repulsing range of the lower coil. However, the SISO configuration using the upper coil and the MIMO configuration for the upper magnet are unbounded, since the upper magnet can move outside the attracting range of the upper coil and out of the range of control. Therefore, as explained in section 3.2, we attach a clamp to limit the movement of the upper magnet in such configurations so that it is always inside the attracting range of the upper coil, to ensure that such configurations remain bounded. Thus, all of the SISO and MIMO configurations of the magnetic levitation plant are stable and bounded, where their dynamic models in (3.1)–(3.4) can be expressed as in (2.18) and (2.19) and satisfy the conditions in assumptions 2-2(a), 2-2(c), and 3-1.

The last step is to choose a suitable value for the learning parameter c, so that it satisfies the boundaries given in chapter 2 and [84]. For the weights between the hidden layer and the output layer, m_{qj}, the boundary of the learning parameter is chosen as

$$0 < c < \frac{2}{n_q} \tag{3.29}$$

For the weights between the input layer and the hidden layer, m_{iq}, the boundary of the learning parameter is chosen as

$$0 < c < \frac{2}{n_q} \left[\frac{1}{m_{qj,\max} \cdot i_{i,\max}} \right]^2 \tag{3.30}$$

where

$$m_{qj,\max} = \max_k \|m_{qj}(k)\| \tag{3.31}$$

$$i_{i,\max} = \max_k \|i_i(k)\| \tag{3.32}$$

By carrying out the three steps above, we can ensure the convergence and stability of our proposed method for controlling the SISO and MIMO configurations of the magnetic levitation system.

3.7 Experimental Results and Discussion

In our experiment, we will apply the proposed method to both of the SISO configurations using the lower coil and upper coil and to the MIMO configuration of the nonlinear magnetic levitation system, as explained in section 3.2. We will also compare the result of our proposed method to the result of using linear SAC. In our proposed method, the plant dynamics are considered to be unknown. Therefore, in this experiment we consider it unnecessary to know the detailed parameters of the plant dynamics of the magnetic levitation system. The experimental setup of the magnetic levitation system is shown in Fig. 3.8.

3.7.1 SISO Configuration of the Nonlinear Magnetic Levitation System Using the Lower Coil

We apply the control method to produce a repulsive force from the lower coil to control the height of the magnet. The parameters of SAC and the neural

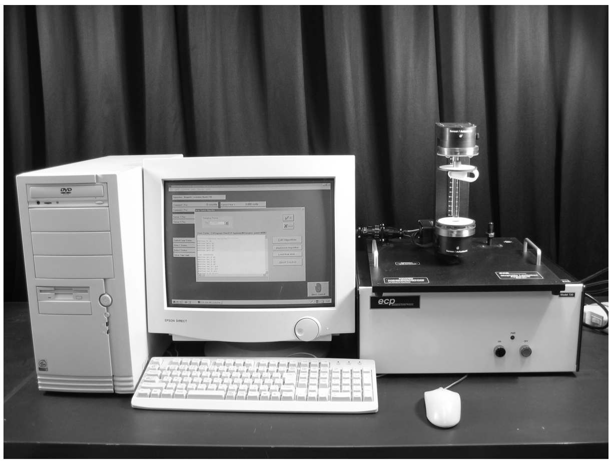

Figure 3.8: Experimental setup of the magnetic levitation system system

network are fixed as below:

$$
\begin{aligned}
T_p &= diag(1,1,1) \quad \text{(in (3.16))} \\
T_i &= diag(2,2,2) \quad \text{(in (3.17))} \\
\sigma' &= 0.99999 \quad \text{(in (3.17))} \\
D_p &= 10^{-8} \quad \text{(in (3.19))} \\
\rho &= 0.1 \quad \text{(in (3.19))} \\
\Delta T &= 0.0027 \quad \text{(in (3.19))} \\
\alpha &= 1 \quad \text{(in (3.26))} \\
\mu &= 2 \quad \text{(in (2.31))} \\
c &= 0.1 \quad \text{(in (2.34))}
\end{aligned}
$$

Furthermore, we assume the first-order reference model in (3.22) and (3.23) with parameters

$$A_m = 0.9048,\ \ B_m = 0.09516,\ \ C_m = 1$$

For this magnetic levitation system, we estimate the value of J_{plant} by performing some experiments in advance. From those experiments we obtain

$$J_{plant} = -1$$

Using the above parameters, we apply SAC to the SISO nonlinear system using the neural network to control the magnetic levitation plant. The number of neurons of the neural network in the input layer is 2, the number in the

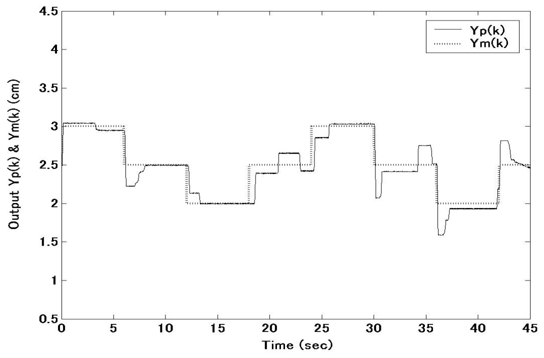

Figure 3.9: $y_m(k)$ and $y_p(k)$ using only SAC

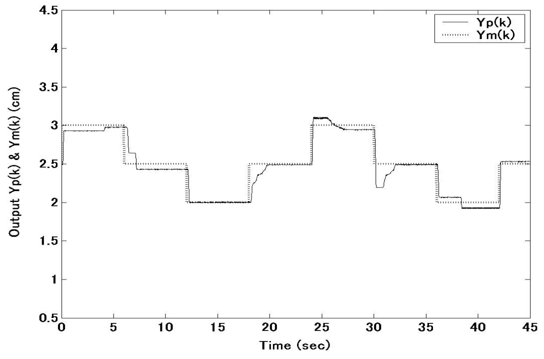

Figure 3.10: $y_m(k)$ and $y_p(k)$ using SAC and neural network simultaneously

hidden layer is 5, and the number in the output layer is 1. The input vector $i(k)$ of the neural network is given as

$$i(k) = [y_m(k-1), y_p(k-1)]^T$$

A sampling period ΔT of 0.0027 seconds is selected in this experiment. Figs. 3.9–3.11 show the results of this experiment.

3.7.2 SISO Configuration of the Nonlinear Magnetic Levitation System Using the Upper Coil

We apply the control method to produce an attractive force from the upper coil to control the height of the magnet. The parameters of SAC and the

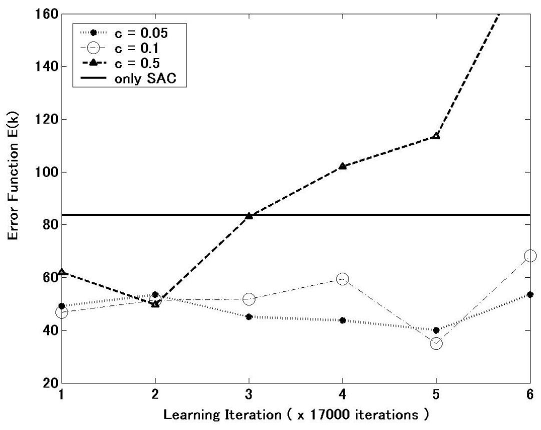

Figure 3.11: Comparison of $E(k)$ using several values of learning parameter c and using only SAC

neural network are fixed as below:

$$\begin{aligned}
T_p &= diag(1,1,1) \quad \text{(in (3.16))}\\
T_i &= diag(2,2,2) \quad \text{(in (3.17))}\\
\sigma' &= 0.99999 \quad \text{(in (3.17))}\\
D_p &= 10^{-8} \quad \text{(in (3.19))}\\
\rho &= 0.1 \quad \text{(in (3.19))}\\
\Delta T &= 0.0027 \quad \text{(in (3.19))}\\
\alpha &= 1 \quad \text{(in (3.26))}\\
\mu &= 2 \quad \text{(in (2.31))}\\
c &= 0.05 \quad \text{(in (2.34))}
\end{aligned}$$

Furthermore, we assume the first-order reference model in (3.22) and (3.23) with parameters

$$A_m = 0.9048, \ \ B_m = 0.09516, \ \ C_m = 1$$

In similar way as in subsection 3.7.1, we obtain

$$J_{plant} = -1$$

and the number of neurons of the neural network in the input layer is 2, the number in the hidden layer is 5, and the number in the output layer is 1. The input vector $i(k)$ of the neural network is given as

$$i(k) = [y_m(k-1), y_p(k-1)]^T$$

A sampling period ΔT of 0.0027 seconds is selected in this experiment. Figs. 3.12–3.14 show the results of this experiment.

3.7.3 MIMO Configuration of the Nonlinear Magnetic Levitation System

We apply the control method to simultaneously produce a repulsive force from the lower coil and an attractive force from the upper coil to simultaneously control the heights of the lower and upper magnets, respectively. The parameters of SAC and the neural network are fixed as below:

$$\begin{aligned}
T_p &= diag(1,1,1,1,1,1) \quad \text{(in (3.16))} \\
T_i &= diag(1,1,1,1,1,1) \quad \text{(in (3.17))} \\
\sigma' &= 0.99999 \quad \text{(in (3.17))} \\
D_p &= diag(10^{-10},10^{-10}) \quad \text{(in (3.19))} \\
\rho &= 0.1 \quad \text{(in (3.19))} \\
\Delta T &= 0.0044 \quad \text{(in (3.19))} \\
\alpha &= 1 \quad \text{(in (3.26))} \\
\mu &= 2 \quad \text{(in (2.31))} \\
c &= 0.05 \quad \text{(in (2.34))}
\end{aligned}$$

Furthermore, we assume the first-order reference model in (3.22) and (3.23) with parameters

$$\begin{aligned}
A_{m_1} &= 0.9048, \;\; B_{m_1} = 0.09516, \;\; C_{m_1} = 1 \\
A_{m_2} &= 0.9048, \;\; B_{m_2} = 0.09516, \;\; C_{m_2} = 1
\end{aligned}$$

In similar way as in subsections 3.7.1 and 3.7.2, we obtain

$$J_{plant_1} = -1, \;\; J_{plant_2} = -1,$$

and the number of neurons of the neural network in the input layer is 4, the number in the hidden layer is 5, and the number in the output layer is 2. The input vector $i(k)$ of the neural network is given as

$$\begin{aligned}
i(k) &= [y_{m_1}(k-1), y_{m_2}(k-1), y_{p_1}(k-1), \\
&\quad\; y_{p_2}(k-1)]^T
\end{aligned}$$

A sampling period ΔT of 0.0044 seconds is selected in this experiment. Figs. 3.15–3.17 show the results of this experiment.

3.7.4 Discussion

In our experiments, we apply the controller that uses only SAC and the controller that uses our proposed method of SAC with a neural network to control both of the SISO configurations using the lower coil and the upper coil and the MIMO configuration of the nonlinear magnetic levitation system. These controllers are required to control the real plant of the magnetic levitation system to follow the desired outputs of the reference models. Then we compare the results of these two controllers.

In subsections 3.7.1–3.7.3, we are required to select suitable values for some parameters. To select the values for these parameters, the selection methods of each SAC and neural network, as described in sections 2.2, 3.3–3.6, are applied independently.

At first, we apply only SAC to the controller and find its parameters using the selection method of SAC described in section 2.2. As mentioned in section 2.2, for linear plants, the augmented plant error $e_y(k)$ can be made very small by setting the parameters T_p and T_i of SAC to very large values. However, this cannot be applied for nonlinear plants. Furthermore, when T_p and T_i are increased to above the hardware capability limitations, $e_y(k)$ will stop decreasing further and start to increase again. Therefore, to set T_p and T_i for nonlinear plants, we start by applying very small values. We increase these values until we find the largest possible values of T_p and T_i, which produce the smallest error. Then, we fix these values of T_p and T_i for the controller using only SAC. Then, we also use these values of T_p and T_i for the SAC part of our proposed controller.

After we have fixed the values of T_p and T_i for the SAC part of our proposed controller, we start to run the neural-network part of our proposed controller to find the learning parameter c that can reduce the remaining error. We start with a small value of the learning parameter c, and increase it by small increments to find the value that can reduce the remaining error function $E(k)$ to a satisfactory minimum level sufficiently quickly while maintaining stability and convergence. In our experiments, the learning of the neural network is performed and completed while the controllers are directly applied to the experimental plants.

As mentioned in section 3.4, increasing the value of the learning parameter c of neural network will reduce $E(k)$ faster. However, in our proposed method, we must keep the value of the learning parameter c inside the boundaries given in (3.29) and (3.30). Otherwise $E(k)$ will increase again and the convergence

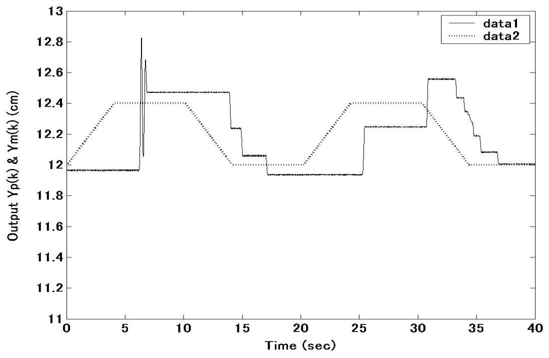

Figure 3.12: $y_m(k)$ and $y_p(k)$ using linear SAC

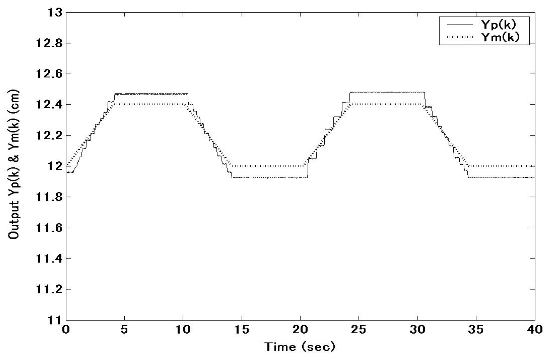

Figure 3.13: $y_m(k)$ and $y_p(k)$ using SAC and neural network simultaneously

and stability of the system cannot be guaranteed. In practical applications, the boundaries can be predicted without knowing their exact values. We can predict the boundaries by noting that by the time the value of the learning parameter c reaches the boundaries, $E(k)$ will have started to increase again. $E(k)$ will increase faster as the value of the learning parameter c increases further, and this will cause the system to become unstable.

The selection of the first-order system for reference models in subsections 3.7.1–3.7.3 is to emphasize the fact that low-order models do not affect the ability of the adaptive control system.

Figs. 3.9–3.16 show the experimental results. Since real plants have nonlinearities and uncertainties, it is difficult to achieve exactly the same plant outputs for the same actions as those shown in Figs. 3.9, 3.10, 3.12, 3.13, 3.15, and 3.16.

Figs. 3.9, 3.12, and 3.15 show the desired output $y_m(k)$ and the plant output $y_p(k)$ using linear SAC. These results show that the plant output $y_p(k)$

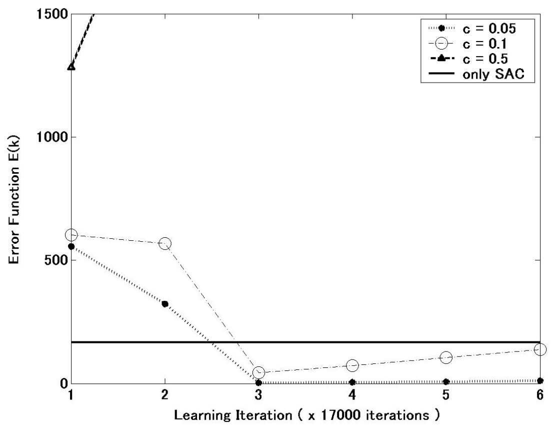

Figure 3.14: Comparison of $E(k)$ using several values of learning parameter c and using only SAC

cannot follow the desired output $y_m(k)$ closely, and also that the output error between $y_p(k)$ and $y_m(k)$ is large.

Figs. 3.10, 3.13, and 3.16 show the desired output $y_m(k)$ and the plant output $y_p(k)$ using SAC and the neural network simultaneously using the online training of the backpropagation algorithm. It can be seen that the output error has been reduced, and that the plant output $y_p(k)$ can follow the desired output $y_m(k)$ closely.

Figs. 3.11, 3.14, and 3.17 show the error function $E(k)$ versus the number of learning iterations for values of the learning parameter from $c = 0.05$ to $c = 0.5$ and also using only SAC. For the SISO configuration using the lower coil, it can be seen from Fig. 3.11 that the error function $E(k)$ can reach its minimum value at $c = 0.1$. When we apply $c = 0.5$, the error function starts to increase again. This shows that $c = 0.5$ is outside the boundaries given in (3.29) and (3.30). For the SISO configuration using the upper coil, it can be seen from Fig. 3.14 that the error function $E(k)$ can reach its minimum value at $c = 0.05$. When we apply $c = 0.1$, the error function starts to increase again. Furthermore, when we apply $c = 0.5$, the error function becomes very large and the system becomes unstable and diverges. This shows that $c = 0.1$ and $c = 0.5$ are outside the boundaries given in (3.29) and (3.30). For the MIMO configuration, it can be seen from Fig. 3.17 that the error function $E(k)$ can reach its minimum value at $c = 0.05$. When we apply $c = 0.1$, the error function does not change much. Furthermore, when we apply $c = 0.5$, the error function becomes very large and the system becomes unstable and diverges. This shows that $c = 0.5$ is outside the boundaries given in (3.29)

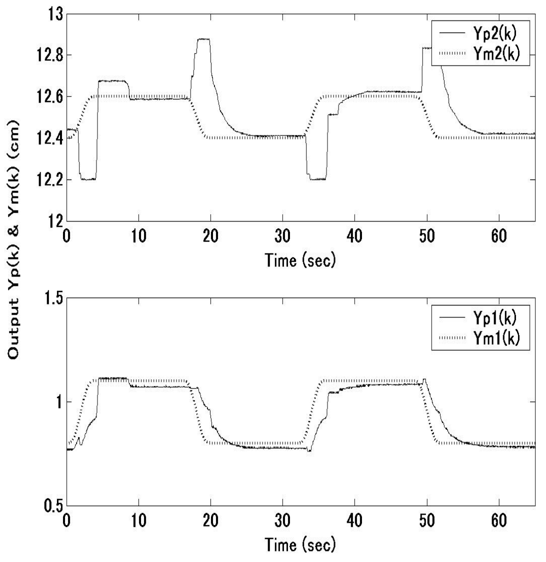

Figure 3.15: $y_m(k)$ and $y_p(k)$ using linear SAC

and (3.30).

3.8 Conclusions

In this chapter, we have proposed a discrete-time control method for SISO and MIMO configurations of a nonlinear magnetic levitation system using the method of SAC with a neural network. The control input was given by the sum of the output of the simple adaptive controller and the output of the neural network. The role of the neural network was to compensate for the nonlinearity of the plant by constructing a linearized model so as to minimize the output error caused by nonlinearities in the control system. Furthermore, in this chapter, the magnetic levitation system was set to satisfy the assumptions required by chapter 2 and [84]; thus, the stability analysis in chapter 2 and [84] could be applied. Finally, experiments were executed, and the effectiveness of this proposed control method was confirmed; it was shown that the output $y_p(k)$ of the magnetic levitation system, for all of the SISO and MIMO configurations, could converge to the desired output $y_m(k)$ after the learning of the neural network was performed.

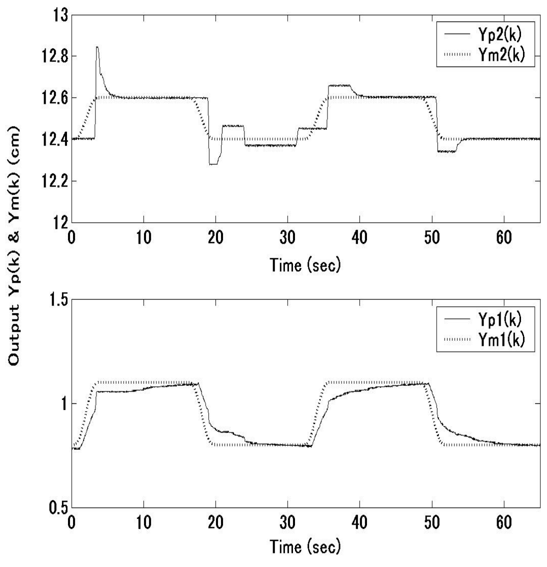

Figure 3.16: $y_m(k)$ and $y_p(k)$ using SAC and neural network simultaneously

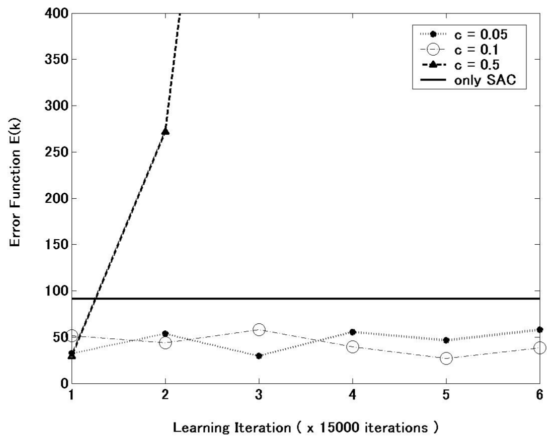

Figure 3.17: Comparison of $E(k)$ using several values of learning parameter c and using only SAC

Chapter 4

A Control Method for Nonlinear Systems Using SAC with Multiple Neural Networks

4.1 Introduction

Adaptive control methods were developed as an attempt to overcome difficulties connected with the ignorance of system structure and critical parameter values as well as changing control regimes [2, 5, 7, 14, 17]. Among adaptive control methods, SAC procedure was developed by Sobel et al. [19, 20] as an attempt to simplify adaptive controllers, since no observers or identifiers are needed in the feedback loop [34]. Furthermore, the reference model is allowed to be of a very low order compared to the controlled plant. For linear plants with unknown structures, SAC is an important class of adaptive control schemes [13, 34, 36].

In the beginning, researches in the field of adaptive control methods were focussed on linear plants. However, recently, their focus has been transferred to nonlinear plants [26]. Therefore, dealing with nonlinear systems using the concept of SAC has also been investigated [77, 78]. However, for nonlinear plants with unknown structures, it is difficult to ensure a perfect plant output that follows the output of a reference model by using conventional SAC [21, 22, 84, 86].

To solve the problem of using SAC to control nonlinear systems, at first, control methods for nonlinear systems using a combination of SAC and neural networks have been proposed in [21] for SISO systems and in [22] for MIMO systems. Chapter 2 and reference [84] have provided the theoretical explanations about convergence and stability analysis of the method of SAC using neural network in the general form of MIMO, and defined the class of nonlinear systems that can be controlled using that method. Chapter 3 and reference [86] have discussed the application of the control method of SAC using neural network for nonlinear magnetic levitation systems.

The methods in chapters 2 and 3 and in [22,84,86] are using a single neural network. Because of that, the performance of those methods can be increased by increasing the size of neural network, which means increasing the number of the neurons in the neural network [9,87]. As the size of the neural network is increased, the calculation time required for each sampling time in the training process will also increase. When the size of the neural network is very large, the calculation process will be very time consuming. If these methods are applied to control an actual plant, it is necessary to drastically reduce the calculation time required for each sampling time in the training process of the neural network [21].

To solve this problem, references [21,88] have proposed the using of several parallel small-scale neural networks instead of a single neural network. However, the methods in [21,88] have some deficiencies, theoretical explanations about convergence and stability analysis are not provided and the class of nonlinear systems which can be controlled is not provided. Basing on and extending the analysis given in chapter 2 and in reference [84], we attempt to overcome these deficiencies by providing a thorough theoretical explanations about convergence and stability analysis, and defining the class of nonlinear systems which can be controlled. Thus, we expect to have a more general method with a more general and thorough theoretical explanations compared to chapters 2 and 3 and references [21,22,84,86,88].

This chapter proposes a control method for nonlinear systems using SAC with multiple neural networks. The control input is given by the sum of the output of the simple adaptive controller and the output of the multiple neural networks. The multiple neural networks consists of several parallel small-scale neural networks having identical structures. The role of the multiple neural networks is to compensate for constructing a linearized system so as to minimize the output error caused by nonlinearities in the controlled system. By using the multiple neural networks, we expect to drastically reduce the calculation time required for each sampling time in the training process of neural networks. The role of the simple adaptive controller is to perform the model matching for the linearized system to a given linear reference model. In this chapter, we use a design method using backpropagation training algorithm of simple multilayer feedforward neural network, using the direct neural adaptive control method, to train the multiple neural networks of our method. Furthermore, a thorough theoretical explanation of convergence and stability analysis for this method is performed. Convergence and stability analysis shows that stability can be guaranteed for the class of nonlinear systems with BIBO and bounded nonlinearities. Finally, computer simulations for an SISO and a 2-inputs 2-outputs MIMO nonlinear systems are executed and the effectiveness of this control method is confirmed.

The preeliminary version of our method has been presented in [89] in the scope of SISO. The proposed method of this chapter will be discussed in the more general scope of MIMO.

4.2 Nonlinear SAC with Multiple Neural Networks

In this chapter, we consider the nonlinear system in (2.18), (2.19) which satisfies assumption 2-2. To overcome the problems of the overlarge size of the neural network and the time consuming calculation process of the method proposed in chapters 2 and 3 and references [22, 84, 86], we will apply several parallel small-scale neural networks, called multiple neural networks, instead of one large neural network. Each of these small-scale neural networks has an identical structure. Thus, the control input vector of the neural network in (2.21)–(2.23) is replaced with the control input vector of the multiple neural networks which will be synthesized as

$$\begin{aligned} \bar{u}_p(t) &= \alpha \hat{u}_p(t) && (4.1) \\ \hat{u}_p(t) &= f_{zoh}\left(\sum_{v=1}^{n_v} \hat{u}_{p_v}(k)\right) && (4.2) \end{aligned}$$

where n_v is the number of small-scale neural networks and $\hat{u}_{p_v}(k) = [\hat{u}_{p_{v_1}}(k), \cdots, \hat{u}_{p_{v_{n_j}}}(k)]^T$ is the discrete-time output vector of the multiple neural networks. The control input of SAC in (2.21) will be calculated using (2.12)–(2.17) in section 2.2. The nonlinear SAC with multiple neural networks and the multiple neural networks are represented in Figs.4.1 and 4.2, respectively.

As in chapter 2 and references [22,84], in our proposed method, we also implement a sampler in front of the multiple neural networks with an appropriate sampling period ΔT to obtain the discrete-time multi-input of the multiple neural networks, and a zero-order to transform the sum of discrete-time output $\hat{u}_{p_v}$ of the multiple neural networks, as in (4.2) back to continuous-time output $\hat{u}_p(t)$.

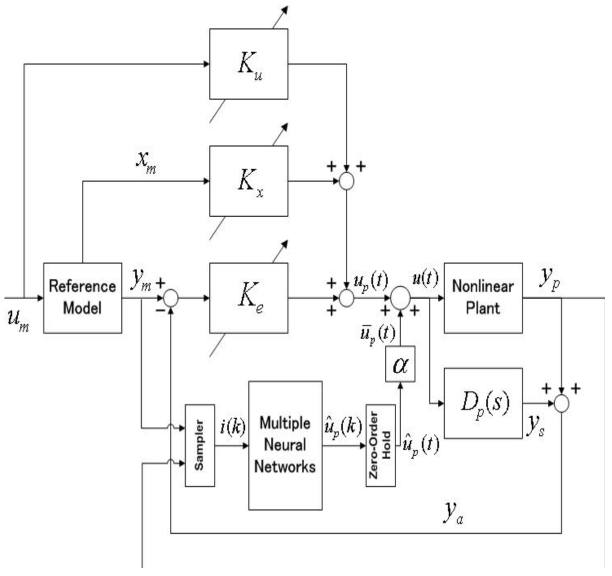

Figure 4.1: Structure of the nonlinear SAC system with multiple neural networks

Consequently, we can assume that the discrete-time output $\hat{u}_p(k)$ of the multiple neural networks is as follows

$$\begin{aligned}\hat{u}_{p_v} &= \hat{h}(y_m^T(k-1),\\ &\quad y_p^T(k-1),\cdots,y_p^T(k-n))\end{aligned}\tag{4.3}$$

where $\hat{h}(\cdot)$ is an unknown nonlinear function vector and n is the number of past outputs of the plant.

Using the above approach, the multiple neural networks will be trained. As in chapter 2, the training process is done by adjusting the weights of the multiple neural networks, using the standard backpropagation algorithm described in chapters 2 and 3 and references [84, 86], until the output error $e(t)$ given in (2.25) satisfies the relation in (2.26).

4.3 Composition and Learning of Multiple Neural Networks

Each small-scale neural network of the multiple neural networks has an identical structure. Expanding from chapter 2 and reference [84], let the v-th parallel small-scale neural network ($v = 1, \cdots, n_v$) consists of three layers: an input layer, an output layer and a hidden layer. Let $i_{vi}(k)$ be the input to the i-th neuron in the input layer ($i = 1, \cdots, n_i$), $h_{vq}(k)$ be the input to the

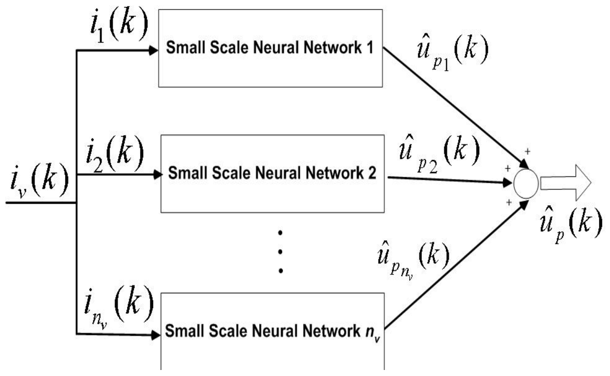

Figure 4.2: Structure of the multiple neural networks

q-th neuron in the hidden layer ($q = 1, \cdots, n_{q_v}$), $o_{vj}(k)$ be the input to the j-th neuron in the output layer ($j = 1, \cdots, n_j$), where $n_i(k)$, $n_{q_v}(k)$, and n_j are the number of neurons in the input layer, hidden layer, and output layer, respectively. Furthermore, let m_{viq} be the weights between the input layer and the hidden layer, m_{vqj} be the weights between the hidden layer and the output layer. A small-scale neural network is represented in Fig.4.3.

The control input is given by the sum of the output of the simple adaptive controller and the output of the multiple neural networks. The multiple neural networks are used to compensate for the nonlinearity of the plant dynamics that is not taken into consideration in the usual SAC. The role of the multiple neural networks is to construct a linearized model by minimizing the output error caused by the nonlinearities in the control systems. Refering to (4.3), the input vector $i_v(k)$ of the multiple neural networks is given as

$$i_v(k) = [y_m^T(k-1), y_p^T(k-1), \cdots, y_p^T(k-n)]^T. \tag{4.4}$$

Therefore, the nonlinear function of the system can be approximated by the multiple neural networks. Furthermore, the value of n should be chosen appropriately according to practical nonlinear systems.

Adopting from chapter 2 and reference [84], we also can obtain

$$h_{vq}(k) = \sum_i i_{vi}(k) m_{viq}(k) \tag{4.5}$$

$$o_{vj}(k) = \sum_q S_1(h_{vq}(k)) m_{vqj}(k) \tag{4.6}$$

$$\hat{u}_{p_{v_j}}(k) = S_2(o_{vj}(k)) \tag{4.7}$$

where $S_1(\cdot)$ is a sigmoid function, $S_2(\cdot)$ is a pure linear function, and $j =$

$1, 2, \cdots, n_j$. The sigmoid function is chosen as

$$S_1(X) = \frac{2}{1 + \exp(-\mu X)} - 1 \tag{4.8}$$

where $\mu > 0$, and the pure linear function is chosen as

$$S_2(X) = X. \tag{4.9}$$

Consider the case when $S_1(X) = a$. Then the derivative of the sigmoid function $S_1(\cdot)$ and the pure linear function $S_2(\cdot)$ are as follows

$$\begin{aligned} S_1'(X) &= \frac{\mu}{2}(1 - a^2) && (4.10) \\ S_2'(X) &= 1. && (4.11) \end{aligned}$$

The objective of training is to minimize an error function $E(k)$ by taking the error gradient with respect to the parameters or the weight vector $m(k)$, that are to be adapted. The error function is defined as

$$\begin{aligned} E(k) &= \frac{1}{2} e^T(k) e(k) \\ &= \frac{1}{2} \sum_{j}^{n_j} \left[y_{m_j}(k) - y_{p_j}(k) \right]^2 && (4.12) \end{aligned}$$

then, the weights are adapted by using

$$\varDelta m_v(k) = -c \cdot \frac{\partial E(k)}{\partial m_v(k)} \tag{4.13}$$

where $c > 0$ is the learning parameter. For the learning process, (4.13) will be expanded as follows

$$\begin{aligned} \varDelta m_{vqj}(k) &= -c \cdot \frac{\partial E(k)}{\partial y_{p_j}(k)} \cdot \frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_{v_j}}(k)} \cdot \frac{\partial \hat{u}_{p_{v_j}}}{\partial S_2(o_{vj}(k))} \\ &\quad \cdot \frac{\partial S_2(o_{vj}(k))}{\partial o_{vj}(k)} \cdot \frac{\partial o_{vj}(k)}{\partial m_{vqj}(k)} && (4.14) \\ \varDelta m_{viq}(k) &= -c \cdot \sum_{j}^{n_j} \frac{\partial E(k)}{\partial y_{p_j}(k)} \cdot \frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_{v_j}}(k)} \\ &\quad \cdot \frac{\partial \hat{u}_{p_{v_j}}(k)}{\partial S_2(o_{vj}(k))} \cdot \frac{\partial S_2(o_{vj}(k))}{\partial o_{vj}(k)} \\ &\quad \cdot \frac{\partial o_{vj}(k)}{\partial S_1(h_{vq}(k))} \cdot \frac{\partial S_1(h_{vq}(k))}{\partial h_{vq}(k)} \\ &\quad \cdot \frac{\partial h_{vq}(k)}{\partial m_{viq}(k)} && (4.15) \end{aligned}$$

where

$$\begin{aligned}
\frac{\partial E(k)}{\partial y_{p_j}(k)} &= -\left[y_{m_j}(k) - y_{p_j}(k)\right] \\
\frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_{v_j}}(k)} &= J_{plant_{v_j}} \\
\frac{\partial \hat{u}_{p_{v_j}}(k)}{\partial S_2(o_{vj}(k))} &= 1 \\
\frac{\partial S_2(o_{vj}(k))}{\partial o_{vj}(k)} &= 1 \\
\frac{\partial o_{vj}(k)}{\partial m_{vqj}(k)} &= S_1(h_{vq}(k)) \\
\frac{\partial o_{vj}(k)}{\partial S_1(h_{vq}(k))} &= m_{vqj}(k) \\
\frac{\partial S_1(h_{vq}(k))}{\partial h_{vq}(k)} &= \frac{\mu}{2}\left[1 - S_1^2(h_q(k))\right] \\
\frac{\partial h_{vq}(k)}{\partial m_{viq}(k)} &= i_{vi}(k)
\end{aligned}$$

Furthermore, $J_{plant_{v_j}}$ represents the Jacobian of the plant. According to [38], this plant Jacobian can be estimated by using an identified parameter and the internal variables of the neural network model in the indirect neural adaptive control. In many cases, this $J_{plant_{v_j}}$ is clear from physical insight or can be estimated through some experiments, as mentioned and proposed in [4,9].

Therefore, considering for holding the fundamental design concept of SAC, i.e. without any identifiers, in this chapter we utilize from the direct neural adaptive control method [4,9]

$$J_{plant_{v_j}} = \mathrm{SGN}(\frac{\partial y_{p_j}(k)}{\partial \hat{u}_{p_{v_j}}(k)}) \tag{4.16}$$

where SGN(·) is a sign function.

4.4 Calculation Cost

This section will give an analysis and comparisson of calculation cost between a single large neural network and multiple neural networks. For simplicity, in the analysis we focus on the numbers of multiplication required to update each of the weights of the neural networks. We assume that it requires n_{miq} multiplications to update the weights between the input layer and the hidden layer, and n_{mqj} multiplications to update the weights between the hidden layer and the output layer.

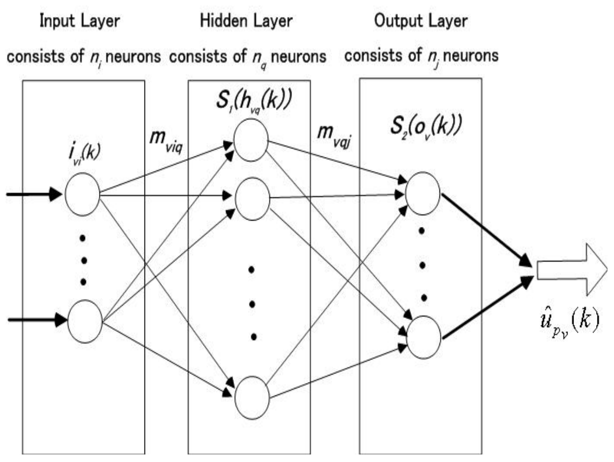

Figure 4.3: Structure of a small-scale neural network

First, let us consider a single large neural network with n_i neurons in the input layer, n_j neurons in the output layer, and n_q neurons in the hidden layer as described in section 2.4 of chapter 2. Then, the total number $n_{singleNN}$ of multiplication required in one sampling time to update its weight is

$$n_{singleNN} = n_i n_q n_{miq} + n_j n_q n_{mqj} \tag{4.17}$$

Now, let us consider multiple neural networks with n_v parallel small-scale neural networks described in section 4.3. Each small-scale neural network consists of n_i neurons in the input layer, n_j neurons in the output layer, and n_{q_v} neurons in the hidden layer. In one sampling time, n_v parallel small-scale neural networks will perform calculation simultaneously. Thus, to update its weight, the number $n_{multiNN}$ of multiplication required in one sampling time is

$$n_{multiNN} = n_i n_{q_v} n_{miq} + n_j n_{q_v} n_{mqj} \tag{4.18}$$

The relations between the number of neurons in a single large neural network and a small-scale neural network of multiple neural networks are given as

$$n_q = n_v n_{q_v} \tag{4.19}$$

Then Eq.(4.18) can be rewritten as

$$\begin{aligned} n_{multiNN} &= \frac{1}{n_v}(n_i n_q n_{miq} + n_j n_q n_{mqj}) \\ &= \frac{1}{n_v} n_{singleNN} \end{aligned} \tag{4.20}$$

Eq.(4.20) shows that in one sampling time, multiple neural networks perform multiplication $\frac{1}{n_v}$ times less than a single large neural network. This means that the time for multiple neural networks required for one sampling time is smaller than the time required by a single large neural network.

4.5 Convergence and Stability

The stability analysis of SAC for a controllable and observable linear plant with unknown parameters and disturbances has been presented in [13], where the plant is as described in (2.38), (2.39). This plant (2.38), (2.39) is controllable and observable and fulfills the ASPR condition in assumption 2-1. Thus, the theorem 2-1 given in [13] will hold.

For the stability analysis of our method, we will follow and expand the stability proof presented in section 2.6 and [84]. As mentioned in assumption 2-2(b), the PFC in (2.7), (2.9) is incorporated with the nonlinear system in (2.18), (2.19) to form the augmented plant, as in (2.6), which its linear part is ASPR. However, for convenience, first it is necessary for the PFC in (2.7), (2.9) to be transformed into a state-space form as described in (2.40), (2.41). Then, by applying (2.40), (2.41) to (2.7), (2.18), (2.19), the augmented plant can be described as (2.42)–(2.46).

The nonlinear part of the system in (2.42), (2.43) will be compensated and minimized using the control input of the multiple neural networks $\bar{u}_p(t)$, to form a linearized system. The control input of SAC $u_p(t)$ will perform model matching of the linearized system to a given linear reference model. The nonlinearity compensation, minimization, and the linear model matching processes will be performed simultaneously. Therefore, it is necessary in our method that the control system is able to keep its stability while performing those processes. We use the following theorem 4-1 to prove the stability of our method.

Theorem 4-1: Assume that the nonlinear augmented plant in (2.42), (2.43) satisfies assumption 2-2, then the control system described in section 4.2 is globally stable with respect to boundedness if the boundaries of the learning parameter c of the multiple neural networks are set as [84]

$$0 < c < \frac{2}{n_{q_v}(k)} \tag{4.21}$$

for the weights m_{vqj} between the hidden layer and the output layer, and as

$$0 < c < \frac{2}{n_{q_v}(k)} \left[\frac{1}{m_{vqj,\max} \cdot i_{vi,\max}} \right]^2 \tag{4.22}$$

for the weights m_{viq} between the input layer and the hidden layer, where

$$m_{vqj,\max} = \max_k \|m_{vqj}(k)\| \quad (4.23)$$

$$i_{vi,\max} = \max_k \|i_{vi}(k)\| \quad (4.24)$$

In other words, all values (states, gains, and errors) involved in the control of the nonlinear augmented plant are bounded. Furthermore, with a sufficient number of neurons in each of the parallel small-scale neural network and a sufficient number n_v of the parallel small-scale neural networks of the multiple neural networks, the remaining output tracking error $\mathbf{e}_y(t)$ caused by nonlinearity can be directly controlled and thus reduced via a sufficient number of training iterations of the multiple neural networks.

Proof: We start by defining the Lyapunov function of our method as follows

$$V_{SACNN}(t) = V_{SAC}(t) + V_{MultiNN}(t) \quad (4.25)$$

where $V_{SAC}(t)$ is the Lyapunov function of SAC of our method, which is a modification from the one presented in [13], and $V_{MultiNN}(t)$ is the Lyapunov function of the multiple neural networks of our method. Then, the derivative of Lyapunov function of our method becomes

$$\dot{V}_{SACNN}(t) = \dot{V}_{SAC}(t) + \dot{V}_{MultiNN}(t). \quad (4.26)$$

Following section 2.6 and [84], we start from the stability analysis of the SAC part of our method. The Lyapunov function and derivative of Lyapunov function of the SAC part of our method are expanded from the functions presented in [13] for SAC for linear plant with disturbances in (2.38), (2.39). We replace the terms $\delta_i(t)$ and $\delta_o(t)$ with $\hat{\delta}_i(x(t), u(t), \bar{u}_p(t))$ and $\hat{\delta}_o(x(t))$, respectively. As in section 2.6 and [84], the Lyapunov function of the SAC part of our method is described as (2.47). The derivative of the Lyapunov function in (2.47) is described as (2.52), (2.53).

For the derivative of Lyapunov function in (2.52), (2.53), as in section 2.6 and [84], we can directly apply the same method as the one used in [13] to prove the stability of our method if $\hat{\delta}_i(x(t), u(t), \bar{u}_p(t))$ and $\hat{\delta}_o(x(t))$ are bounded [84]. Refering to (2.46) and assumption 2-2(c), as $f_y(\cdot)$ is bounded by the assumption, then $\hat{\delta}_o(x_p(t))$ is also bounded. However, refering to (2.45) and assumption 2-2(c), eventhough $f_x(\cdot)$ is assumed to be bounded, $\hat{\delta}_i(x(t), u(t), \bar{u}_p(t))$ is bounded if and only if $\bar{u}_p(t)$ is also bounded. Therefore, based on (4.1), (4.2), to prove that $\bar{u}_p(t)$ is bounded, it is necessary to prove the convergence of the multiple neural networks.

To prove the convergence of the multiple neural networks part of our method, we use the method presented in section 2.6 and [84], that followed [41,42], and apply it to each parallel small-scale neural network that forms the multiple neural networks. The Lyapunov function $V_{NN_v}(k)$ and its derivative $\Delta V_{NN_v}(k)$ of the v-th parallel small-scale neural network is defined as

$$\begin{aligned} V_{NN_v}(k) &= \frac{1}{2}e^2(k) \qquad (4.27) \\ \Delta V_{NN_v}(k) &= V_{NN_v}(k+1) - V_{NN_v}(k) \\ &= \frac{1}{2}\left[e^2(k+1) - e^2(k)\right]. \qquad (4.28) \end{aligned}$$

By expanding (4.13) as follows

$$\Delta m_v(k) = c \cdot e(k) \cdot J_{plant_v} \cdot \frac{\partial o_{vj}(k)}{\partial m_v(k)} \tag{4.29}$$

then as shown in [41], $\Delta V_{NN_v}(k)$ in (4.28) can be represented as

$$\begin{aligned} &\Delta V_{NN_v}(k) = \Delta e(k)\left[e(k) + \frac{1}{2}\Delta e(k)\right] \\ &= \left[\frac{\partial e(k)}{\partial m_v(k)}\right]^T \cdot c \cdot e(k) \cdot J_{plant_v} \cdot \frac{\partial o_{vj}(k)}{\partial m_v(k)} \\ &\cdot \left\{ e(k) + \frac{1}{2}\left[\frac{\partial e(k)}{\partial m_v(k)}\right]^T \cdot c \cdot e(k) \cdot J_{plant_v} \cdot \frac{\partial o_{vj}(k)}{\partial m_v(k)} \right\} \end{aligned} \tag{4.30}$$

where the convergence is guaranteed if the boundary of c is chosen such that

$$0 < c < \frac{2}{J^2_{plant_v,\max} \cdot g_{v\max}^2} \tag{4.31}$$

as proven in [41], where $J_{plant_v,\max}$ is the maximum limit of the plant Jacobian, which refers to (4.16), will be

$$J_{plant_v,\max} = 1, \tag{4.32}$$

and

$$\begin{aligned} g_{v\max} : &= \max_k \|g_v(k)\| \qquad (4.33) \\ g_v(k) &= \frac{\partial o_{vj}(k)}{\partial m_v(k)} \qquad (4.34) \end{aligned}$$

where $\|\cdot\|$ is the usual Euclidean norm in R^n.

Furthermore, from (4.31)–(4.33), we choose (4.21)–(4.22) as the boundaries of the learning parameter for each type of weights of the parallel small-scale neural networks in the multiple neural networks. Proofs of (4.21)–(4.22) are presented in [84].

If the learning parameter c is set to be inside the boundaries in (4.21)–(4.22), $\Delta V_{NN_v}(k)$ in (4.30) will be negative definite. Thus, the convergence of each parallel small-scale neural network of the multiple neural networks part of our method can be guaranteed, and $\bar{u}_p(t)$ will be bounded. This means that $\hat{\delta}_i(x(t), u(t), \bar{u}_p(t))$ will be bounded too, and the stability of the SAC part of our method can also be guaranteed.

Then, in general, since the derivative of Lyapunov function of the multiple neural networks $\dot{V}_{MultiNN}(t)$ in (4.26) approximated as

$$\dot{V}_{MultiNN}(t) \cong \alpha \sum_{v}^{n_v} \frac{\Delta V_{NN_v}(k)}{\Delta T} \tag{4.35}$$

is negative definite, $\dot{V}_{SACNN}(t)$ in (4.26) is also negative definite. Thus, the convergence of the multiple neural networks part and the stability of our method can be guaranteed. This means that as the training progresses, the error function $E(k)$ in (4.12) will be minimized, and the output error $e(t)$ in (2.25) will satisfy the relation in (2.26). This shows that the nonlinearity of the system in (2.18)–(2.19) are compensated for and minimized using the control input of the multiple neural networks $\bar{u}_p(t)$.

For the SAC method using a single neural network in [84], finding a sufficient number of neurons in the single neural network is very important for the controller to be able to reduce the output error to satisfy the relation in (2.26). However, as stated in Sect. 3, this will cause problems if the number of neurons is too large. The use of multiple neural networks is proposed in this chapter to overcome these problems. From (4.35), it can be seen that the negative definiteness of (4.26) and (4.35) can be increased by increasing the number n_v of the parallel small-scale neural network of the multiple neural networks. Each parallel small-scale neural network of the multiple neural networks consists of a sufficiently small number of neurons.

Furthermore, (4.35) also shows that either increasing the value of α or reducing the sampling period ΔT will also increase the negative definiteness of (4.26) and (4.35).

4.6 Simulation Results and Discussion

For nonlinear systems, two cases are considered, one of an SISO system and one of an MIMO system.

4.6.1 SISO System

Let us consider the SISO nonlinear system from [84] described by

$$\begin{aligned}
\begin{bmatrix} \dot{x_{p1}} \\ \dot{x_{p2}} \end{bmatrix} &= \begin{bmatrix} x_{p2} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\
&+ \begin{bmatrix} 0 \\ 2\overset{10}{\underset{-10}{f_{sat}}}(x_{p1}\sin(x_{p1})) \end{bmatrix} \\
y_p &= x_{p1} + \sin(x_{p1})
\end{aligned}$$

where $\overset{n_{upper}}{\underset{n_{lower}}{f_{sat}}}(\cdot)$ is a saturation function with a lower limit at n_{lower} and an upper limit at n_{upper}. Then, the parameters of our method are set as

$$\begin{aligned}
T_p &= diag(5\times 10^3, 5\times 10^3, 5\times 10^3) \quad \text{(in (2.16))}, \\
T_i &= diag(5\times 10^4, 5\times 10^4, 5\times 10^4) \quad \text{(in (2.17))}, \\
\sigma &= 1 \quad \text{(in (2.17))}, \\
\alpha &= 1 \quad \text{(in (4.1))}, \\
\mu &= 2 \quad \text{(in (4.8))}, \\
c &= 0.001 \quad \text{(in (4.13))}, \\
D_p &= 0.001 \quad \text{(in (2.9))}, \\
\rho &= 1 \quad \text{(in (2.9))}
\end{aligned}$$

and PFC

$$D_p(s) = \frac{D_p}{1+\rho s} = \frac{0.001}{1+s}$$

is fixed to guarantee that assumption 2-2(b) is satisfied. Furthermore, we assume a first-order reference model with parameters

$$A_m = -10, \quad B_m = 10, \quad C_m = 1.$$

We estimate the value of J_{plant_j} for this SISO nonlinear system by previously carried experiments with the nonlinear system. From those experiments we get

$$J_{plant_{v1}} = -1.$$

For each of the parallel small-scale neural networks, the number of neurons in the input layer is 2, in the hidden layer is 2, in the output layer is 1, and the input $i_v(k)$ is given as

$$i_v(k)(k) = [y_m(k-1), y_p(k-1)]^T.$$

Furthermore, a sampling period of $0.01sec$ is selected to obtain the values of $i_v(k)$ from $[y_m(t), y_p(t)]$, where $i_v(k)$ denotes $i_v(t)$ at $t = k\Delta T$.

In the simulation, for this SISO nonlinear system, the training is performed using the parameters set above in 4501 iterations for $n_v = 1$ to $n_v = 3$. Figures 4.4–4.8 show the simulation results.

Furthermore, Tables 4.1 and 4.2 show the comparisson of simulation results using $n_v = 3$ and different values of α and sampling period ΔT.

4.6.2 MIMO System

Let us consider the MIMO two-input two-output nonlinear system from [84] described by

$$\begin{aligned}
\begin{bmatrix} \dot{x}_{p1} \\ \dot{x}_{p2} \\ \dot{x}_{p3} \\ \dot{x}_{p4} \end{bmatrix} &= \begin{bmatrix} -x_{p1} + x_{p4} \\ x_{p2} \\ -x_{p2} - x_{p3} \\ x_{p3} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \\
&\quad + \begin{bmatrix} 0 \\ \underset{-10}{\overset{10}{f_{sat}}}(x_{p1}x_{p3}) + \underset{-10}{\overset{10}{f_{sat}}}(2x_{p3}u_1) \\ \underset{-10}{\overset{10}{f_{sat}}}(x_{p1}^2) + \underset{-10}{\overset{10}{f_{sat}}}(2x_{p3}u_1) \\ 0 \end{bmatrix} \\
\begin{bmatrix} y_{p1} \\ y_{p2} \end{bmatrix} &= \begin{bmatrix} x_{p2} - x_{p3} \\ x_{p4} \end{bmatrix} + \begin{bmatrix} \underset{-10}{\overset{10}{f_{sat}}}(x_{p1}x_{p3}) \\ \underset{-10}{\overset{10}{f_{sat}}}(x_{p1}^2) \end{bmatrix}
\end{aligned}$$

where $\overset{n_{upper}}{\underset{n_{lower}}{f_{sat}}}(\cdot)$ is a saturation function with a lower limit at n_{lower} and an upper limit at n_{upper}. Then, the parameters of our method are set as

$$\begin{aligned}
T_p &= diag(1.7 \times 10^5, 1.7 \times 10^5, 1.7 \times 10^5, \\
&\quad 1.7 \times 10^5, 1.7 \times 10^5, 1.7 \times 10^5) \quad \text{(in (2.16))}, \\
T_i &= diag(1.7 \times 10^6, 1.7 \times 10^6, 1.7 \times 10^6, \\
&\quad 1.7 \times 10^6, 1.7 \times 10^6, 1.7 \times 10^6) \quad \text{(in (2.17))}, \\
\sigma &= 0.1 \quad \text{(in (2.16))}, \\
\alpha &= 10 \quad \text{(in (4.1))}, \\
\mu &= 2 \quad \text{(in (4.8))}, \\
c &= 0.01 \quad \text{(in (4.13))}, \\
D_p &= diag(0.002, 0.002) \quad \text{(in (2.9))}, \\
\rho &= 1 \quad \text{(in (2.9))}
\end{aligned}$$

and PFC

$$D_p(s) = \frac{D_p}{1+\rho s} = \begin{bmatrix} \frac{0.002}{1+s} & 0 \\ 0 & \frac{0.002}{1+s} \end{bmatrix}$$

is fixed to guarantee that assumption 2-2(b) is satisfied. Furthermore, we assume first-order reference models with parameters

$$\begin{aligned}
A_{m_1} = -10, \quad B_{m_1} = 10, \quad C_{m_1} = 1, \\
A_{m_2} = -10, \quad B_{m_2} = 10, \quad C_{m_2} = 1.
\end{aligned}$$

We also estimate the values of J_{plant_j} for this MIMO nonlinear system by previously carried experiments with the nonlinear system. From those experiments we get

$$J_{plant_{v_1}} = +1, \qquad J_{plant_{v_2}} = +1.$$

For each of the parallel small-scale neural networks, the number of neurons in the input layer is 8, in the hidden layer is 2, in the output layer is 2, and the input $i_v(k)$ is given as

$$i_v(k) = [y_m^T(k-1), y_p^T(k-1), y_p^T(k-2), y_p^T(k-3)]^T.$$

Furthermore, same as the previous case of the SISO nonlinear system, a sampling period of $0.01sec$ is also selected to obtain the values of $i_v(k)$ from $[y_m(t), y_p(t)]$.

In the simulation, for this MIMO nonlinear system, the training is performed using the parameters set above in 1453 iterations for $n_v = 1$ to $n_v = 3$. Figures 4.9–4.13 show the simulation results.

Furthermore, Tables 4.3 and 4.4 show the comparisson of simulation results using $n_v = 3$ and different values of α and sampling period ΔT.

4.6.3 Discussion

The selections of the first-order reference models for the simulations of the SISO and MIMO nonlinear systems are to emphasize the fact that low-order models do not affect the ability of the adaptive control system.

The simulation results of the SISO and MIMO nonlinear systems are shown in Figures 4.4–4.12 and Tables 4.1–4.4.

Figures 4.4 and 4.9 show the desired output $y_m(t)$ and the plant output $y_p(t)$ using only SAC. The result in Fig.4.4 shows that the error between $y_p(t)$ and $y_m(t)$ is large.

Figures 4.5 and 4.10 show the desired output $y_m(t)$ and the plant output $y_p(t)$ using our method with $n_v = 1$. From Figures 4.5 and 4.10, it can be seen that the error of the system has been reduced and the plant output $y_p(t)$ can follow closer the desired output $y_m(t)$ compared to using only SAC.

For $n_v = 2$, Figures 4.6 and 4.11 show that the error of the system has been more reduced, where the plant output $y_p(t)$ can follow closer the desired output $y_m(t)$, compared to using our method with $n_v = 1$.

Figure 4.7 and 4.12 show that when using $n_v = 3$ for our method the error of the system has been very much reduced and the plant output $y_p(t)$ can follow closely the desired output $y_m(t)$.

Thus, Figures 4.4–4.8, for the SISO nonlinear, and Figures 4.9–4.13, for the MIMO nonlinear system, show that, with the same numbers of training iteration, values of α, and sampling periods ΔT, the sum of the square error of the system is decreasing as the number n_v of the parallel small-scale neural networks is increased. Figures 4.8 and 4.13 show the comparisson of the error function $E(k)$ using SAC with multiple neural networks and with a single neural network. From Figures 4.8 and 4.13, we can see that the error function $E(k)$ of our method is close enough to the error function $E(k)$ of the previous method of SAC using a single neural network.

Furthermore, from Tables 4.1, 4.2, 4.3, and 4.4, it can also be seen that, using the weights of the multiple neural networks resulted from the training performed previously using parameters set in subsections 4.6.1 and 4.6.2, the sum of the square error of the system is decreased as either the value of α is

Table 4.1: Comparison of the sum of square error with $n_v = 3$, $\Delta T = 0.01 sec$, and different values of α (SISO system)

Values of α	**Sum of Square Error** ($n_v = 3$, $\Delta T = 0.01$ **sec**)
0.1	1.2311
0.5	0.4338
1	0.3181

Table 4.2: Comparison of the sum of square error with $n_v = 3$, $\alpha = 1$, and different values of sampling period ΔT (SISO system)

Sampling Time ΔT **(sec)**	**Sum of Square Error** ($n_v = 3$, $\alpha = 1$)
1	7.9946
0.1	5.6626
0.01	0.3181

increased or the sampling period ΔT is decreased.

4.7 Conclusions

This chapter proposed a control method for nonlinear systems using SAC with multiple neural networks. The control input was given by the sum of the output of a simple adaptive controller and the output of multiple neural networks. Furthermore, thorough theoretical analysis and explanation of calculation cost, convergence, and stability for this method have been presented and discussed. As shown in the calculation cost analysis, it was obvious that the calculation process time required for the training process of the multiple neural networks was smaller than using a single neural network. The convergence and stability analysis showed that stability can be guaranteed for the class of nonlinear systems with BIBO and bounded nonlinearities. Finally, computer simulations for an SISO and a 2-inputs 2-outputs MIMO nonlinear systems were executed and the effectiveness of this control method has been confirmed.

For applications to real nonlinear systems, each parallel small-scale neural network in our method can be developed on one dedicated circuit or microchip. Applications of our method to real nonlinear systems will be considered in our future research.

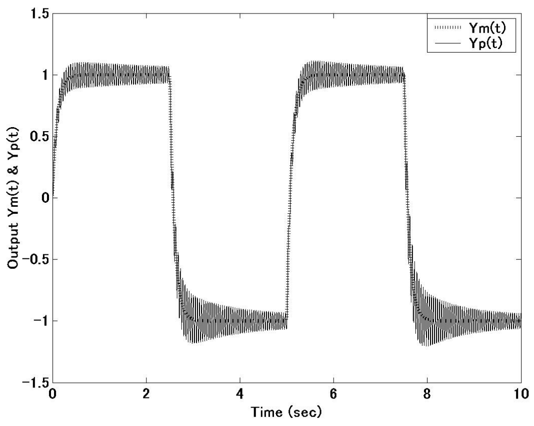

Figure 4.4: $y_m(t)$ and $y_p(t)$ using only SAC (SISO system)

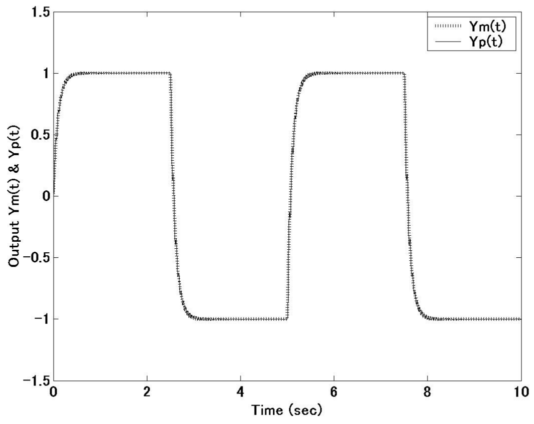

Figure 4.5: $y_m(t)$ and $y_p(t)$ using SAC and 1 small-scale neural network (SISO system)

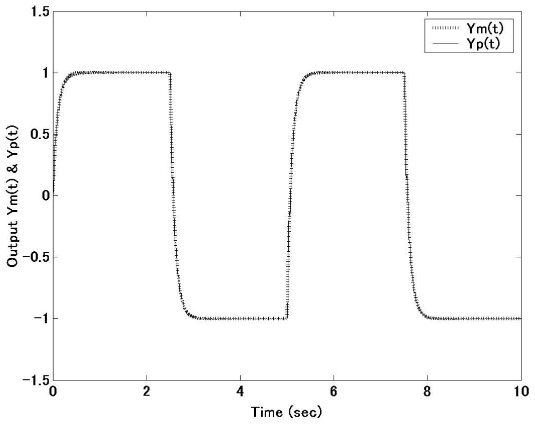

Figure 4.6: $y_m(t)$ and $y_p(t)$ using SAC and 2 parallel small-scale neural networks (SISO system)

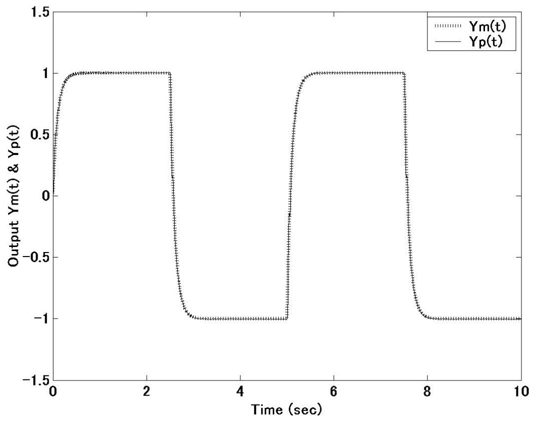

Figure 4.7: $y_m(t)$ and $y_p(t)$ using SAC and 3 parallel small-scale neural networks (SISO system)

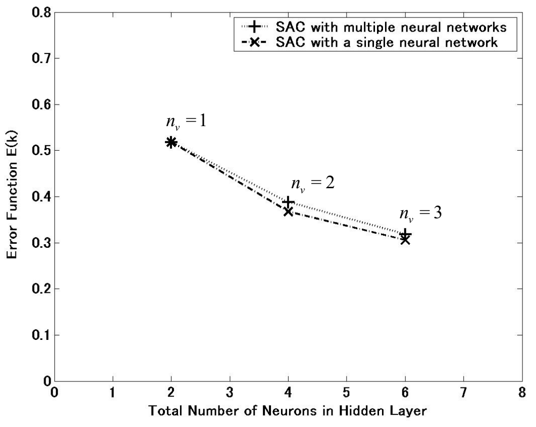

Figure 4.8: Comparison of $E(k)$ using SAC with multiple neural networks and with a single neural network (SISO system)

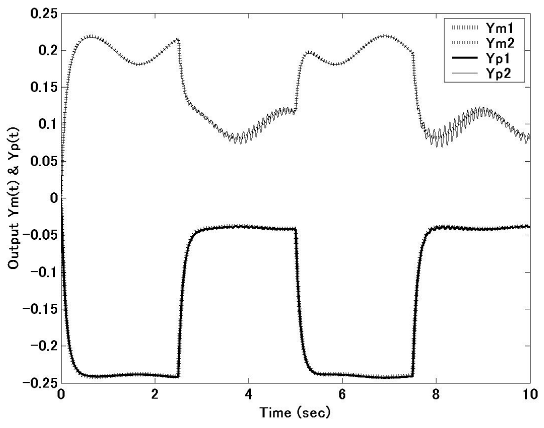

Figure 4.9: $y_m(t)$ and $y_p(t)$ using only SAC (MIMO system)

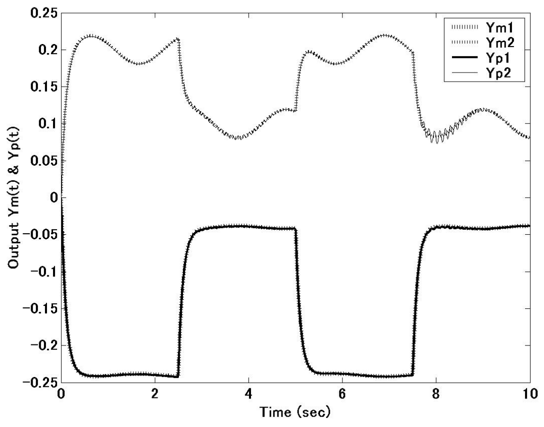

Figure 4.10: $y_m(t)$ and $y_p(t)$ using SAC and 1 small-scale neural network (MIMO system)

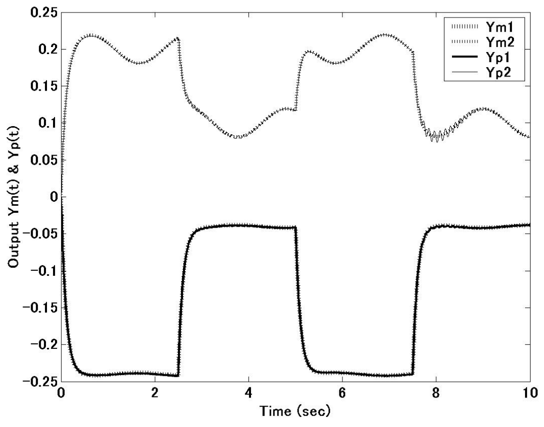

Figure 4.11: $y_m(t)$ and $y_p(t)$ using SAC and 2 parallel small-scale neural networks (MIMO system)

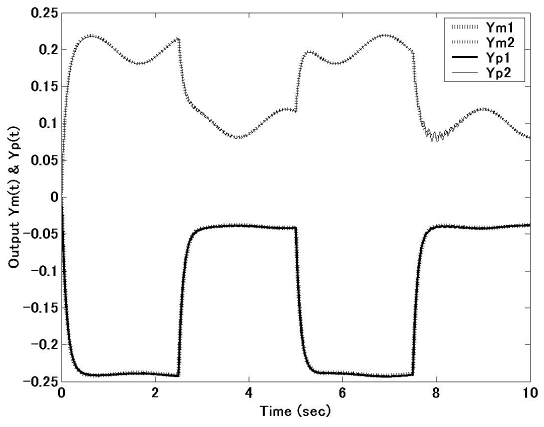

Figure 4.12: $y_m(t)$ and $y_p(t)$ using SAC and 3 parallel small-scale neural networks (MIMO system)

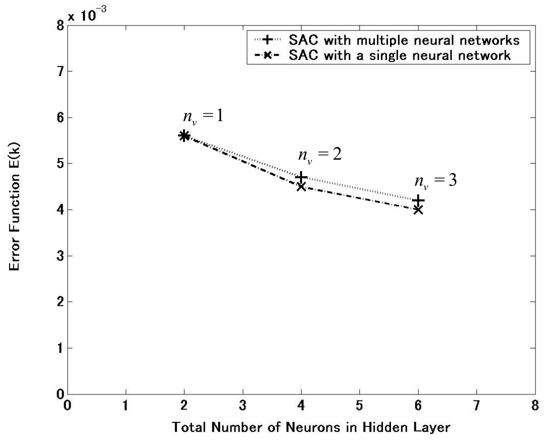

Figure 4.13: Comparison of $E(k)$ using SAC with multiple neural networks and with a single neural network (MIMO system)

Table 4.3: Comparison of the sum of square error with $n_v = 3$, $\Delta T = 0.01 sec$, and different values of α (MIMO system)

Values of α	**Sum of Square Error** **(**$n_v = 3$**,** $\Delta T = 0.01$ **sec)**
1	0.0146
5	0.0068
10	0.0042

Table 4.4: Comparison of the sum of square error with $n_v = 3$, $\alpha = 10$, and different values of sampling period ΔT (MIMO system)

Sampling Time ΔT **(sec)**	**Sum of Square Error** **(**$n_v = 3$**,** $\alpha = 10$**)**
1	0.0262
0.1	0.0072
0.01	0.0042

Chapter 5

Adaptive SMC Using SAC for Nonlinear Systems

5.1 Introduction

The ignorance of plant structure and critical parameter values, plant disturbances, uncertainties, and changing control regimes have caused difficulties in designing appropriate controllers for real plants. Adaptive control methods were developed to solve such difficulties [2,5,7,14,17]. In the beginning, the researches of adaptive control methods were focussed for linear plants. However, recently, their focus has been transferred to nonlinear plants [26].

Among the existing adaptive control methods, SAC procedure was developed by Sobel et al. [19, 20]. It provides a simplified method to design and develop adaptive controllers. It is called 'simple' because it does not require any observers or identifiers in its control structure [13, 34, 35]. Furthermore, it can use a reference model that is allowed to be of very low order compared to the controlled plant. For linear plants with unknown structures, SAC is an important class of adaptive control scheme [13,34,36]. However, for nonlinear plants, using conventional SAC is difficult to ensure a perfect plant output that follows the output of a reference model [22, 84]. For nonlinear plants with unknown structures, a method of SAC using neural networks has been developed previously [22, 84].

On the other hand, variable structure control with sliding mode, which is commonly known as SMC, is a nonlinear control strategy that is well known for its robust characteristics [53]. It has a good ability in controlling nonlinear systems with parameter uncertainties and disturbances to follow the desired trajectories. It can switch the control law to drive the system states from any initial state onto a user-specified sliding surface, and to maintain the states on the surface for all subsequent time [53, 71].

The general approach of conventional SMC has two drawbacks [55, 62, 69, 71]. One of the drawbacks is the chattering phenomenon that is highly undesirable and discussed in [55,62,71]. Another drawback is the difficulty in calculating its equivalent control law. This is caused by the requirement of thorough knowledge of the controlled plant parameters and dynamics [55, 62, 69]. Since those parameters and dynamics are difficult to obtain or even unknown, the calculation of the equivalent control law of SMC is very difficult and causes computational burden [73–75]. Recently, intelligent techniques based on fuzzy logic and neural networks have been applied to SMC to overcome this problem [62,64,66,68,73]. However, those methods still require complex calculation process and consume time to calculate the control law of SMC.

In this chapter, a new method of adaptive SMC strategy using SAC for nonlinear systems with no explicit knowledge of parameters and dynamics other than the assumption that the systems have BIBO, bounded nonlinearities and bounded first and second derivatives of the output nonlinearities is proposed. This method is proposed to deal with:

1. the difficulties in the conventional SAC to control nonlinear plants with unknown structures,
2. the difficulties in the standard SMC caused by the requirement of thorough knowledge of the controlled plant parameters and dynamics,
3. and the complexities and time consuming processes in the existing methods of SMC using intelligent techniques.

In this proposed method, the role of SAC is to construct an equivalent control input of adaptive SMC. To construct a corrective control input, this chapter applies a method using the sign function with a modified sliding surface. Hence, except the assumption that the systems have BIBO, bounded nonlinearities and bounded first and second derivatives of the output nonlinearities, no explicit prior knowledge of the plant is required for calculating the equivalent control law. Furthermore, the stability analysis of the proposed method is performed. The stability analysis shows that stability can be guaranteed for systems with BIBO, bounded nonlinearities, and bounded first and second derivatives of the output nonlinearities. Finally, computer simulations are executed and the effectiveness of our control method is confirmed. Computer simulations also show that the proposed control method can keep the chattering phenomenon minimal.

The preeliminary versions of our proposed method have been presented in [74, 75] in the scope of SISO. The proposed method of this chapter will be discussed in a more general scope of MIMO.

5.2 General SMC

General SMC based on state-space formulation is presented briefly in this section. First let us consider a nonlinear plant described in (2.18), (2.19). We further assume that the plant (2.18), (2.19) is BIBO, controllable, and observable.

The steps in designing a sliding mode controller are:

1. to construct a sliding surface that represents a desired system dynamics,
2. and to develop a switching control law such that a sliding mode exists on every point of the sliding surface, and any state outside the surface is driven to reach the surface in a finite time.

The control objective is to determine a control input $u_p(t)$ such that the state vector $x_p(t)$ tracks a given bounded desired state vector $\hat{x}_p(t) \in R^{n_p \times 1}$. Therefore, the states error can be obtained as

$$\begin{aligned} e_{x_p}(t) &= \hat{x}_p(t) - x_p(t) \\ &= \left[e_{x_p}(t), \dot{e}_{x_p}(t), \cdots, \overset{n_p-1}{e_{x_p}}(t) \right]^T . \end{aligned} \tag{5.1}$$

Then, the sliding surface vector $S(t) \in R^{n_j \times 1}$ in the space of state error can be obtained as

$$\begin{aligned} S(t) &= \begin{bmatrix} S_1(t) \\ \vdots \\ S_{n_j}(t) \end{bmatrix} = \begin{bmatrix} c_1^T e_{x_p}(t) \\ \vdots \\ c_{n_j}^T e_{x_p}(t) \end{bmatrix} \\ &= \left[\; c_1, \cdots, c_{n_j} \; \right]^T e_{x_p}(t) \end{aligned} \tag{5.2}$$

where $c_j = \left[c_{j_1}, \cdots, c_{j_{n_p}} \right]^T$ is the slope of the sliding surface ($j = 1, \cdots, n_j$). The coefficients $c_{j_1}, \cdots, c_{j_{n_p}}$ describe the dynamics of the sliding surface. Generally, c_j is chosen to force the state error to converge to zero when the state is on the sliding surface. Any state that reaches this surface will then remain on it for all subsequent time, and a sliding mode is said to occur.

When a system is in the sliding mode, its dynamics are governed only by the dynamics of the sliding surface. Therefore the coefficients $c_{j_1}, \cdots, c_{j_{n_p}}$

must be chosen such that the system in the sliding mode produces the desired behaviour [55].

On the other hand, the process of SMC can be divided into two phases, the approaching phase with $S(t) \neq 0$ and the sliding phase with $S(t) = 0$. A sufficient condition to guarantee that the trajectory of the error vector $e_{x_p}(t)$ will translate from the approaching phase to the sliding phase is to select the control strategy such that

$$S^T(t)\dot{S}(t) \leq -\eta \|S(t)\| \tag{5.3}$$

where η is a small positive constant and $\|\cdot\|$ is the usual Euclidean norm. Condition (5.3) is called the reaching condition [57]. Corresponding to the two phases, two types of control law can be derived separately. In the sliding phase, we have $S(t) = 0$ and $\dot{S}(t) = 0$, then the equivalent control input vector $u_{eq}(t)$ will force the system dynamics to stay on the sliding surface. The derivative of the sliding surface vector $\dot{S}(t)$ is derived from (5.2) as

$$\begin{aligned} \dot{S}(t) &= \left[\begin{array}{c} c_1, \cdots, c_{n_j} \end{array}\right]^T \left[\dot{\hat{x}}_p(t) - \dot{x}_p(t)\right] \\ &= \left[\begin{array}{c} c_1, \cdots, c_{n_j} \end{array}\right]^T \left[\dot{\hat{x}}_p(t) - A_p x_p(t) - f_x(x_p(t), u_{eq}(t)) - B_p u_{eq}(t)\right] \\ &= 0. \end{aligned} \tag{5.4}$$

Then, from (5.4), the equivalent control input vector $u_{eq}(t)$ is chosen as

$$\begin{aligned} u_{eq}(t) &= -\left(\left[\begin{array}{c} c_1, \cdots, c_{n_j} \end{array}\right]^T B_p\right)^{-1} \left[\begin{array}{c} c_1, \cdots, c_{n_j} \end{array}\right]^T \\ &\quad \left[A x_p(t) - \dot{\hat{x}}_p(t) + f_x(x_p(t), u_{eq}(t))\right]. \end{aligned} \tag{5.5}$$

In the approaching phase, where $S(t) \neq 0$, in order to satisfy the reaching condition (5.3), the corrective control input $u_c(t)$ (or the so-called the switching function) must be added.

First, let the Lyapunov function be selected as

$$V_{SMC}(t) = \frac{S^T(t)S(t)}{2}. \tag{5.6}$$

It can be noted that this function is positive definite. It is aimed that the derivative of the Lyapunov function is negative definite. This can be achieved if one can assure that

$$\begin{aligned} \dot{S}(t) &= -k_s \operatorname{sign}(S(t)) \\ &= -k_s \left[\operatorname{sign}(S_1(t)), \cdots, \operatorname{sign}\left(S_{n_j}(t)\right)\right]^T \end{aligned} \tag{5.7}$$

where $k_s \in R^{n_j \times n_j}$ is a positive definite constant diagonal matrix, and

$$\operatorname{sign}(S_j(t)) = \begin{cases} +1, \text{ if } S_j(t) > 0 \\ 0, \text{ if } S_j(t) = 0 \\ -1, \text{ if } S_j(t) < 0 \end{cases} \quad (j = 1, \cdots, n_j). \tag{5.8}$$

Substituting (5.7) into the derivative of (5.6), the derivative of the Lyapunov function is obtained as follows

$$\dot{V}_{SMC}(t) = S^T(t)\dot{S}(t) = -S^T(t)\, k_s \left[\operatorname{sign}(S_1(t)), \cdots, \operatorname{sign}\left(S_{n_j}(t)\right)\right]^T. \tag{5.9}$$

Furthermore, the reaching condition (5.3) is achieved if

$$S^T(t)\, k_s \left[\operatorname{sign}(S_1(t)), \cdots, \operatorname{sign}\left(S_{n_j}(t)\right)\right]^T \geq \eta \left\| S(t) \right\|. \tag{5.10}$$

Again, the time derivative of (5.2) can be represented as

$$\dot{S}(t) = \left[\ c_1, \cdots, c_{n_j}\ \right]^T \left[\dot{\hat{x}}_p(t) - A_p x_p(t) - f_x(x_p(t), u(t)) - B_p u(t)\right]. \tag{5.11}$$

Then, substituting (5.11) into the left hand side of (5.7), the control input vector of SMC can be written as

$$\begin{aligned} u(t) &= -\left(\left[\ c_1, \cdots, c_{n_j}\ \right]^T B_p\right)^{-1} \left[\ c_1, \cdots, c_{n_j}\ \right]^T \\ &\quad \left[A x_p(t) - \dot{\hat{x}}_p(t) + f_x(x_p(t), u_{eq}(t))\right] \\ &\quad + \left(\left[\ c_1, \cdots, c_{n_j}\ \right]^T B_p\right)^{-1} k_s \left[\operatorname{sign}(S_1(t)), \cdots, \operatorname{sign}\left(S_{n_j}(t)\right)\right]^T \\ &= u_{eq}(t) + u_c(t) \end{aligned} \tag{5.12}$$

where

$$u_c(t) = \left(\left[\ c_1, \cdots, c_{n_j}\ \right]^T B_p\right)^{-1} k_s \left[\operatorname{sign}(S_1(t)), \cdots, \operatorname{sign}\left(S_{n_j}(t)\right)\right]^T \tag{5.13}$$

is the corrective control input vector. Let

$$K_s = \left(\left[\ c_1, \cdots, c_{n_j}\ \right]^T B_p\right)^{-1} k_s \tag{5.14}$$

then the final form of the corrective control input vector $u_c(t)$ can be written as

$$u_c(t) = K_s \left[\operatorname{sign}(S_1(t)), \cdots, \operatorname{sign}\left(S_{n_j}(t)\right)\right]^T \tag{5.15}$$

where $K_s \in R^{n_j \times n_j}$ is called the switching gain matrix.

5.3 Adaptive SMC Using SAC

SAC, explained in section 2.2, can control linear plants with unknown structures perfectly [13,34]. However, when using only SAC to control the nonlinear plant (2.18), (2.19) to follow the output of the reference model (2.4), (2.5), the problem of output errors will arise [22, 84].

On the other hand, SMC is known to have a good ability in controlling nonlinear systems with parameter uncertainties and disturbances to follow the desired trajectories. However, according to (5.5), thorough knowledge of parameters and dynamics of the nonlinear plant (2.18), (2.19), such as A_p, B_p, C_p, $f_x(\cdot)$, and $f_y(\cdot)$ are required to construct the equivalent control input vector $u_{eq}(t)$ of SMC. Since such information is difficult to obtain or unknown, the calculation of the equivalent control law of SMC is very difficult and causes computational burden [73–75].

Therefore, to solve the problems mentioned above, we implement SAC law (2.6)–(2.10), (2.12)–(2.17), to construct the equivalent control law, to form adaptive SMC. First, it is necessary to use assumption 2-2 and add the following assumption.

Assumption 5-1

(a) The first and second derivatives of the nonlinear part $f_y(\cdot)$ of the nonlinear plant in (2.18), (2.19) are bounded.

Thus, by using (2.12), the equivalent control input vector $u_{eq}(t)$ can be described as

$$u_{eq}(t) \;=\; u_p(t) = K(t)r(t). \tag{5.16}$$

In our proposed method of adaptive SMC using SAC, we consider that the sliding surface $S(t)$ in (5.2), that using state error $e_{x_p}(t)$ in (5.1), is not necessarily known. Therefore, based on [68], we use the augmented plant error $e_y(t)$ in (2.8) to form the modified sliding surface described as

$$S_y(t) = \begin{bmatrix} S_{y1}(t) \\ \vdots \\ S_{y_{n_j}}(t) \end{bmatrix} = \begin{bmatrix} c_{y1}^T \bar{e}_{y1}(t) \\ \vdots \\ c_{y_{n_j}}^T \bar{e}_{y_{n_j}}(t) \end{bmatrix} \tag{5.17}$$

where $c_{y_j} = \left[c_{y_{j1}}, c_{y_{j2}}\right]^T$ is the slope of the modified sliding surface ($j = 1, \cdots, n_j$) and $\bar{e}_{y_j}(t)$ is the vector consisting of the augmented plant error in (2.8) of the j-th output of the plant and its derivatives described as

$$\bar{e}_{y_j}(t) = \left[e_{y_j}(t), \dot{e}_{y_j}(t)\right]^T. \tag{5.18}$$

The derivatives of the augmented plant error in ([?]) are estimated using differentiators.

The approaching phase is performed when $S_y(t) \neq 0$, and the sliding phase is performed when $S_y(t) = 0$. Furthermore, by applying (5.17) to (5.15), then putting the result together with (5.16) into (5.12), the control input vector of adaptive SMC can be expressed as

$$\begin{aligned} u(t) &= u_{eq}(t) + u_c(t) \\ &= K(t)r(t) + K_{s_y}\left[\operatorname{sign}\left(S_{y1}(t)\right), \cdots, \operatorname{sign}\left(S_{y_{n_j}}(t)\right)\right]^T \\ &= K(t)r(t) + K_{s_y}\operatorname{sign}(S_y(t)) \end{aligned} \tag{5.19}$$

where $K_{s_y} \in R^{n_j \times n_j}$ is the switching gain matrix of adaptive SMC.

5.4 Stability

The stability analysis of SAC for a controllable and observable linear plant with bounded disturbances and no explicit knowledge of parameters has been presented in [13], where theorem 2-1 given in [13] will hold.

For the stability analysis of our proposed method, we will modify and extend the stability proof of theorem 2-1 given in [13]. First, as mentioned in assumption 2-2(b), the PFC in (2.7), (2.9) is incorporated with the nonlinear system in (2.18), (2.19) to form the augmented plant, as in (2.6), which its linear part is ASPR. However, for convenience, it is necessary for the PFC in (2.7), (2.9) to be transformed into a state-space form as (2.40),(2.41). Then, by applying (5.19), (2.40), (2.41) to (2.18), (2.19), (2.6), (2.7), the augmented plant can be described as follows

$$\dot{x}(t) = Ax(t) + Bu_{eq}(t) + \hat{\delta}_i(x(t), u(t)) \tag{5.20}$$

$$y_a(t) = Cx(t) + \hat{\delta}_o(x(t)) \tag{5.21}$$

where

$$\begin{aligned} x &= \begin{bmatrix} x_p \\ x_s \end{bmatrix} \in R^{(n_p+1)\times 1}; \quad A = \begin{bmatrix} A_p & 0 \\ 0 & A_s \end{bmatrix}; \quad B = \begin{bmatrix} B_p \\ B_s \end{bmatrix}; \\ C &= \begin{bmatrix} C_p & D_p \end{bmatrix} \end{aligned} \tag{5.22}$$

and $\hat{\delta}_i(x(t), u(t))$ and $\hat{\delta}_o(x(t))$ represent the nonlinear part of the augmented plant described as follows

$$\begin{aligned}
\hat{\delta}_i(x(t),u(t)) &= \begin{bmatrix} f_x(x_p(t),u(t)) \\ 0 \end{bmatrix} + Bu_c(t) \\
&= \hat{\delta}_i'(x_p(t),u(t)) + Bu_c(t) \\
&= \begin{bmatrix} \delta_{i1}(x_p(t),u(t)) \\ \vdots \\ \delta_{in_p+1}(x_p(t),u(t)) \end{bmatrix} \qquad (5.23) \\
\hat{\delta}_o(x(t)) &= f_y(x_p(t)) \\
&= \begin{bmatrix} \delta_{o1}(x_p(t)) \\ \vdots \\ \delta_{on_j}(x_p(t)) \end{bmatrix} \qquad (5.24)
\end{aligned}$$

Then we use the following theorem 5-1 to prove the stability of our proposed method.

Theorem 5-1: Assume that the nonlinear augmented plant in (5.20), (5.21) satisfies assumption 2-2, then the adaptive control system in section 5.3 is globally stable with respect to boundedness. In other words, all values (states, gains, and errors) involved in the control of the nonlinear augmented plant are bounded. Furthermore, the remaining output tracking error $e_y(t)$ caused by nonlinearity can be directly controlled and thus reduced via the switching gain matrix K_{s_y}.

Proof: We start by defining the Lyapunov function of our proposed method as follows

$$V_{ASMC}(t) = V_{SAC}(t) + V_{SMC'}(t) \qquad (5.25)$$

where $V_{SAC}(t)$ is the Lyapunov function of SAC of our method, which is a modification from the one presented in [13], and $V_{SMC'}$, which is derived from (5.6), is the Lyapunov function of SMC of our method. Then, the derivative of Lyapunov function of our proposed method becomes

$$\dot{V}_{ASMC}(t) = \dot{V}_{SAC}(t) + \dot{V}_{SMC'}(t). \qquad (5.26)$$

For the Lyapunov function of SAC of our method $V_{SAC}(t)$ and its derivative $\dot{V}_{SAC}(t)$, we use the following lemma 5-1 derived from section 2.6.

Lemma 5-1: The Lyapunov function of SAC of our method $V_{SAC}(t)$ and its derivative $\dot{V}_{SAC}(t)$ are developed by replacing the disturbance in the Lyapunov function of SAC in [13] with the nonlinear part represented by $\hat{\delta}_i(x(t),u(t))$ and $\hat{\delta}_o(x(t))$. First, as in [13], the Lyapunov function of SAC of our method is chosen as

$$V_{SAC}(t) = e_x^T(t)Pe_x(t) + tr\left\{\left[K_i(t) - \tilde{K}\right] T_i^{-1} \left[K_i(t) - \tilde{K}\right]^T\right\} \qquad (5.27)$$

where P is a real symmetric positive definite matrix, and $tr(\cdot)$ is a trace function. $e_x(t)$ is given as

$$e_x(t) = \hat{x}(t) - x(t) \tag{5.28}$$

where $\hat{x}(t)$ are the ideal target states of the system, and

$$\tilde{K} = [\tilde{K}_e \quad \tilde{K}_x \quad \tilde{K}_u] \tag{5.29}$$

are the unknown ideal gains of SAC. Then, the derivative of the Lyapunov function in (5.27) becomes

$$\begin{aligned} \dot{V}_{SAC}(t) \;=\;& -e_x^T(t)Qe_x(t) - 2\sigma tr\left[(K_i(t) - \tilde{K})T_i^{-1}(K_i(t) - \tilde{K})^T\right] \\ & -2e_y^T(t)e_y(t)e_y^T(t)T_{p_{e_y}}e_y(t) \\ & -2e_y^T(t)e_y(t)\left[x_m^T(t)T_{p_{x_m}}x_m(t) + u_m^T(t)T_{p_{u_m}}u_m(t)\right] \\ & -2\sigma tr\left[(K_i(t) - \tilde{K})T_i^{-1}\tilde{K}^T\right] - 2e_x^T(t)PF(t) \\ & -2\hat{\delta}_o^T(x_p(t))(K_i(t) - \tilde{K})r(t) - 2\hat{\delta}_o^T(x_p(t))e_y(t)e_y^T(t)T_{p_{e_y}}e_y(t) \\ & -2\hat{\delta}_o^T(x_p(t))e_y(t)\left[x_m^T(t)T_{p_{x_m}}x_m(t) + u_m^T(t)T_{p_{u_m}}u_m(t)\right] \end{aligned} \tag{5.30}$$

where Q is a real matrix, and $F(t)$ is given as

$$F(t) = E_{Bias}(t) - B\tilde{K}_e\hat{\delta}_o(x(t)) + \hat{\delta}_i(x(t), u(t)) \tag{5.31}$$

where $E_{Bias}(t)$ is a bias term as explained in [13].

Proof: The steps of development of the derivative of Lyapunov function (5.30) from (5.27) in lemma 5-1 are presented in Appendix 5A.

To define the Lyapunov function of SMC of our method $V_{SMC'}$ and its derivative $\dot{V}_{SMC'}$, we start by using the following lemma 5-2.

Lemma 5-2: We assume that there exists an unknown sliding surface vector $S_x(t)$ in the space of state error $e_x(t)$ in (5.28) described as follows

$$\begin{aligned} S_x(t) &= \begin{bmatrix} S_{x1}(t) \\ \vdots \\ S_{xn_j}(t) \end{bmatrix} = \begin{bmatrix} c_{x1}^T e_x(t) \\ \vdots \\ c_{xn_j}^T e_x(t) \end{bmatrix} \\ &= \begin{bmatrix} c_{x1}, \cdots, c_{xn_j} \end{bmatrix}^T e_x(t) \end{aligned} \tag{5.32}$$

where $c_{xj} = \left[c_{xj_1}, \cdots, c_{xj_{(n_p+1)}}\right]^T$ is the slope of the unknown sliding surface ($j = 1, \cdots, n_j$), and the derivative of the unknown sliding surface $\dot{S}_x(t)$ is given as

$$\begin{aligned} \dot{S}_x(t) &= \begin{bmatrix} c_{x1}, \cdots, c_{xn_j} \end{bmatrix}^T \left[\dot{\hat{x}}(t) - \dot{x}(t)\right] \\ &= \begin{bmatrix} c_{x1}, \cdots, c_{xn_j} \end{bmatrix}^T \left[\dot{\hat{x}}(t) - Ax(t) - Bu_{eq}(t) - \hat{\delta}_i(x(t), u(t))\right]. \end{aligned} \tag{5.33}$$

The relation between $S_x(t)$ in (5.32) and $S_y(t)$ in (5.17) is given as

$$S_{y_j}(t) = S_{xj}(t) - c_{y_j}^T \bar{\delta}_{oj}(x_p(t)) \tag{5.34}$$

where $j = 1, \cdots, n_j$, and

$$\bar{\delta}_{oj}(x_p(t)) = \begin{bmatrix} \delta_{oj}(x_p(t)) \\ \dot{\delta}_{oj}(x_p(t)) \end{bmatrix}. \tag{5.35}$$

Proof: The detailed proof of (5.34) is presented in Appendix 5B.

Then, the Lyapunov function of SMC of our method $V_{SMC'}$ is given as

$$V_{SMC'} = \frac{S_y^T(t) S_y(t)}{2} \tag{5.36}$$

and its derivative $\dot{V}_{SMC'}$ is described as follows

$$\dot{V}_{SMC'} = S_y^T(t)\dot{S}_y(t) = \sum_{j=1}^{n_j} S_{y_j}(t)\dot{S}_{y_j}(t) \tag{5.37}$$

where

$$\begin{aligned} \dot{S}_{y_j}(t) &= \dot{S}_{xj}(t) - c_{y_j}^T \dot{\bar{\delta}}_{oj}(x_p(t)) \\ &= c_{xj}^T \dot{\hat{x}}(t) - c_{xj}^T A x(t) - c_{xj}^T B u_{eq}(t) - c_{xj}^T \hat{\delta}_i(x(t), u(t)) \\ &\quad - c_{y_j}^T \dot{\bar{\delta}}_{oj}(x_p(t)). \end{aligned} \tag{5.38}$$

Then, we assume that the linear part of the plant (5.20), (5.21) can be approximated by the equivalent control input $u_{eq}(t)$, and apply (5.19) into (5.38), so that $\dot{S}_{y_j}(t)$ becomes as follows

$$\begin{aligned} \dot{S}_{y_j}(t) &= -c_{xj}^T \hat{\delta}_i'(x_p(t), u(t)) - c_{y_j}^T \dot{\bar{\delta}}_{oj}(x_p(t)) - c_{xj}^T B u_c(t) \\ &= -c_{xj}^T \hat{\delta}_i'(x_p(t), u(t)) - c_{y_j}^T \dot{\bar{\delta}}_{oj}(x_p(t)) - c_{xj}^T B K_{s_y} \operatorname{sign}(S_y(t)). \end{aligned} \tag{5.39}$$

Thus, by applying (5.39) to (5.37), we can obtain

$$\begin{aligned} \dot{V}_{SMC'} &= -\sum_{j=1}^{n_j} \Big[S_{y_j}(t) c_{xj}^T \hat{\delta}_i'(x_p(t), u(t)) + S_{y_j}(t) c_{y_j}^T \dot{\bar{\delta}}_{oj}(x_p(t)) \\ &\quad + S_{y_j}(t) c_{xj}^T B K_{s_y} \operatorname{sign}(S_y(t)) \Big] \\ &= -S_y^T(t) k_{s_y} \operatorname{sign}(S_y(t)) \\ &\quad - \sum_{j=1}^{n_j} \Big[S_{y_j}(t) c_{xj}^T \hat{\delta}_i'(x_p(t), u(t)) + S_{y_j}(t) c_{y_j}^T \dot{\bar{\delta}}_{oj}(x_p(t)) \Big] \end{aligned} \tag{5.40}$$

where $k_{s_y} \in R^{n_j \times n_j}$ defined as

$$k_{s_y} = \left[\begin{array}{c} c_{x1}, \cdots, c_{xn_j} \end{array} \right]^T BK_{s_y} \tag{5.41}$$

is a positive definite constant matrix.

The stability of our method requires that $\dot{V}_{SMC'}(t)$ in (5.26) to be negative definite. For the derivative of the Lyapunov function of SAC of our method $\dot{V}_{SAC}(t)$ given in lemma 5-1, (5.30), and (5.31), we can apply directly the same method as in [13] to prove the stability of SAC of our method, since it is known from (5.19), (5.23), (5.24), and assumption 2-2(c) that $\hat{\delta}_i(x(t), u(t))$ and $\hat{\delta}_o(x(t))$ are bounded. For the derivative of the Lyapunov function of SMC of our method $\dot{V}_{SMC'}(t)$ given in (5.37) and (5.40), since $\dot{\hat{\delta}}_{oj}(x_p(t))$ is bounded by assumption 5-1(a), its negative definiteness can be reached by setting the values of the switching gain matrix K_{s_y} to be suitable and large enough so that the matrix k_{s_y} will have large enough values. The reaching condition (5.3) can be achieved if

$$\begin{aligned} S_y^T(t) k_{s_y} \operatorname{sign}(S_y(t)) \quad \geq \quad & \eta \left\| S_y^T(t) \right\| - \sum_{j=1}^{n_j} \Big[S_{y_j}(t) c_{xj}^T \hat{\delta}_i'(x_p(t), u(t)) \\ & + S_{y_j}(t) c_{y_j}^T \dot{\hat{\delta}}_{oj}(x_p(t)) \Big] . \end{aligned} \tag{5.42}$$

Furthermore, increasing K_{s_y}, in (5.40) and (5.41), will increase k_{s_y} and the negative definiteness of $\dot{V}_{SMC'}(t)$ in (5.40), which will make the control objective in (2.10) achievable.

5.5 Computer Simulation

As for the nonlinear plants, two cases are considered, one of SISO and one of MIMO.

5.5.1 SISO System

Let us consider the SISO nonlinear plant from [84] described by

$$\begin{aligned} \begin{bmatrix} \dot{x}_{p1} \\ \dot{x}_{p2} \end{bmatrix} &= \begin{bmatrix} x_{p2} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ &\quad + \begin{bmatrix} 0 \\ 2 \overset{10}{\underset{-10}{f_{sat}}} (x_{p1} \sin(x_{p1})) \end{bmatrix} \\ y_p &= x_{p1} + \sin(x_{p1}) \end{aligned}$$

where $\overset{n_{upper}}{\underset{n_{lower}}{f_{sat}}}(\cdot)$ is a saturation function with a lower limit at n_{lower} and an upper limit at n_{upper}. Then, the parameters are set as

$$\begin{aligned}
T_p &= diag(10^3, 10^3, 10^3) \quad \text{(in (2.16))}, \\
T_i &= diag(10^4, 10^4, 10^4) \quad \text{(in (2.17))}, \\
\sigma &= 1 \quad \text{(in (2.17))}, \\
c_y &= [15 \quad 1]^T \quad \text{(in (5.17))}, \\
K_{s_y} &= 149 \quad \text{(in (5.19))}
\end{aligned}$$

and PFC

$$D_p(s) = \frac{D_p}{1+\rho s} = \frac{0.001}{1+s}$$

is fixed to guarantee that assumption 5-2(b) is satisfied. Furthermore, we assume a first-order reference model (2.4), (2.5) with parameters

$$A_m = -10, \quad B_m = 10, \quad C_m = 1.$$

The selection of the first-order model here is to emphasize the fact that low-order models do not affect the ability of the adaptive control system.

Figure 5.1 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using only SAC. The result in Fig.5.1 shows that the error between $y_p(t)$ and $y_m(t)$ is large.

Figure 5.2 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using our proposed method of adaptive SMC using SAC. It can be seen that the error of the system has been reduced, and the plant output $y_p(t)$ can follow very closely the desired output $y_m(t)$.

Furthermore, Fig.5.3 shows the curve of e_y versus $\dot{e}_y$ of using only SAC and Fig.5.4 shows the one of using our proposed method. It can be seen that by using our proposed method, e_y and $\dot{e}_y$ can be minimized, then, be driven onto the sliding surface and finally to the origin ($e_y = 0$ and $\dot{e}_y = 0$).

5.5.2 MIMO System

Let us consider the SISO nonlinear plant from [84] described by

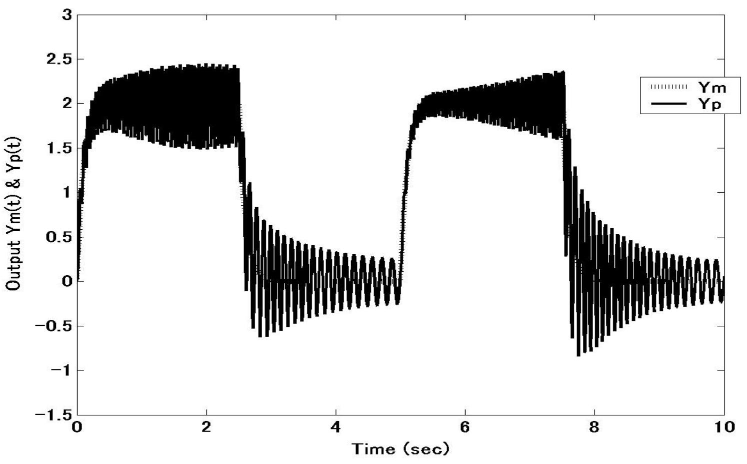

Figure 5.1: $y_m(t)$ and $y_p(t)$ using only SAC (SISO system)

$$\begin{aligned}
\begin{bmatrix} \dot{x_{p_1}} \\ \dot{x_{p_2}} \\ \dot{x_{p_3}} \\ \dot{x_{p_4}} \end{bmatrix} &= \begin{bmatrix} -x_{p_1} + x_{p_4} \\ x_{p_2} \\ -x_{p_2} - x_{p_3} \\ x_{p_3} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \\
&+ \begin{bmatrix} 0 \\ \overset{10}{\underset{-10}{f_{sat}}}(x_{p_1} x_{p_3}) + \overset{10}{\underset{-10}{f_{sat}}}(2 x_{p_3} u_1) \\ \overset{10}{\underset{-10}{f_{sat}}}(x_{p_1}^2) + \overset{10}{\underset{-10}{f_{sat}}}(2 x_{p_3} u_1) \\ 0 \end{bmatrix} \\
\begin{bmatrix} y_{p_1} \\ y_{p_2} \end{bmatrix} &= \begin{bmatrix} x_{p_2} - x_{p_3} \\ x_{p_4} \end{bmatrix} + \begin{bmatrix} \overset{10}{\underset{-10}{f_{sat}}}(x_{p_1} x_{p_3}) \\ \overset{10}{\underset{-10}{f_{sat}}}(x_{p_1}^2) \end{bmatrix}
\end{aligned}$$

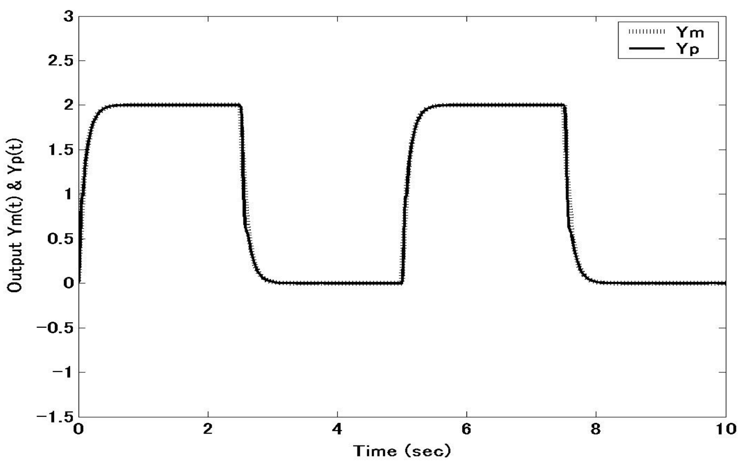

Figure 5.2: $y_m(t)$ and $y_p(t)$ using adaptive SMC with SAC (SISO system)

where $\overset{n_{upper}}{\underset{n_{lower}}{f_{sat}}}(\cdot)$ is a saturation function with a lower limit at n_{lower} and an upper limit at n_{upper}. Then, the parameters are set as

$$\begin{aligned}
T_p &= diag(1.8\times10^5, 1.8\times10^5, 1.8\times10^5,\\
&\quad 1.8\times10^5, 1.8\times10^5, 1.8\times10^5) \quad \text{(in (2.16))},\\
T_i &= diag(1.8\times10^6, 1.8\times10^6, 1.8\times10^6,\\
&\quad 1.8\times10^6, 1.8\times10^6, 1.8\times10^6) \quad \text{(in (2.17))},\\
\sigma &= 0.1 \quad \text{(Eq (2.17))},\\
c_{y_1} &= [1 \;\; 1]^T \quad \text{(in (5.17))},\\
c_{y_2} &= [1 \;\; 1]^T \quad \text{(in (5.17))},\\
K_{s_y} &= \begin{bmatrix} 2 & 0.1 \\ 0.3 & 5 \end{bmatrix} \quad \text{(in (5.19))}
\end{aligned}$$

and PFC

$$D_p(s) = \frac{D_p}{1+\rho s} = \begin{bmatrix} \frac{0.002}{1+s} & 0 \\ 0 & \frac{0.002}{1+s} \end{bmatrix}$$

is fixed to guarantee that assumption 5-2(b) is satisfied. Furthermore, we assume first-order reference models with parameters

$$\begin{aligned}
A_{m_1} = -10, \quad B_{m_1} = 10, \quad C_{m_1} = 1,\\
A_{m_2} = -10, \quad B_{m_2} = 10, \quad C_{m_2} = 1.
\end{aligned}$$

The selection of the first-order models here is to emphasize the fact that low-order models do not affect the ability of the adaptive control system.

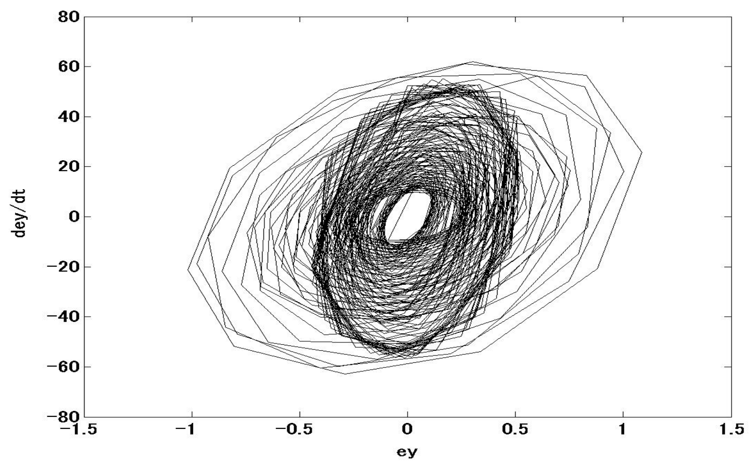

Figure 5.3: e_y versus $\dot{e}_y$ using only SAC (SISO system)

Figure 5.5 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using only SAC. The result in Fig.5.5 shows that the errors between $y_p(t)$ and $y_m(t)$ are large.

Figure 5.6 shows the desired output $y_m(t)$ and the plant output $y_p(t)$ using our proposed method of adaptive SMC using SAC. It can be seen that the errors of the system have been reduced, and the plant output $y_p(t)$ can follow very closely the desired output $y_m(t)$.

Furthermore, Figs.5.7 and 5.8 show the curves of e_{y_j} versus $\dot{e}_{y_j}$ of using only SAC, and Figs.5.9 and 5.10 show the ones of using our proposed method. It can be seen that by using our proposed method, e_{y_j} and $\dot{e}_{y_j}$ can be minimized, then, be driven onto the sliding surface and finally to the origin ($e_y = 0$ and $\dot{e}_y = 0$).

5.6 Conclusion

In this chapter, a new method of adaptive SMC strategy using SAC for non-linear systems with unknown parameters and dynamics other then the assumption that the nonlinear systems have BIBO, bounded nonlinearities, and bounded first and second derivatives of the output nonlinearities, has been proposed. This method was proposed to deal with:

1. the difficulties in the conventional SAC to control nonlinear plants with unknown structures,

2. the difficulties in the standard SMC caused by the requirement of thor-

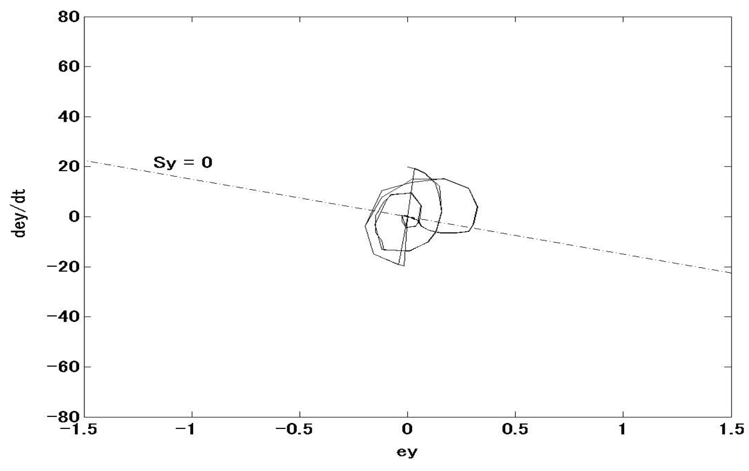

Figure 5.4: e_y versus $\dot{e}_y$ using adaptive SMC with SAC (SISO system)

ough knowledge of the controlled plant parameters and dynamics,

3. and the complexities and time consuming processes in the existing methods of SMC using intelligent techniques.

In this proposed method, the role of SAC was to construct an equivalent control input of adaptive SMC. To construct a corrective control input, this chapter applied a method using the sign function with a modified sliding surface. Hence, except the assumption that the systems have BIBO, bounded nonlinearities and bounded first and second derivatives of the output nonlinearities, no explicit prior knowledge of the plant was required for calculating the equivalent control law. Furthermore, the stability analysis of the proposed method has been performed. The stability analysis showed that stability could be guaranteed for systems with BIBO, bounded nonlinearities, and bounded first and second derivatives of the output nonlinearities. Finally, computer simulations were executed and the effectiveness of our control method has been confirmed. Computer simulations also showed that the proposed control method could keep the chattering phenomenon minimal.

We limited ourselves to present and discuss our proposed method up to theoretical explanations and computer simulations. Further problems of chattering avoidance and other things related to the implementation to real systems will be formalized and solved in further papers with a more applicative point of view.

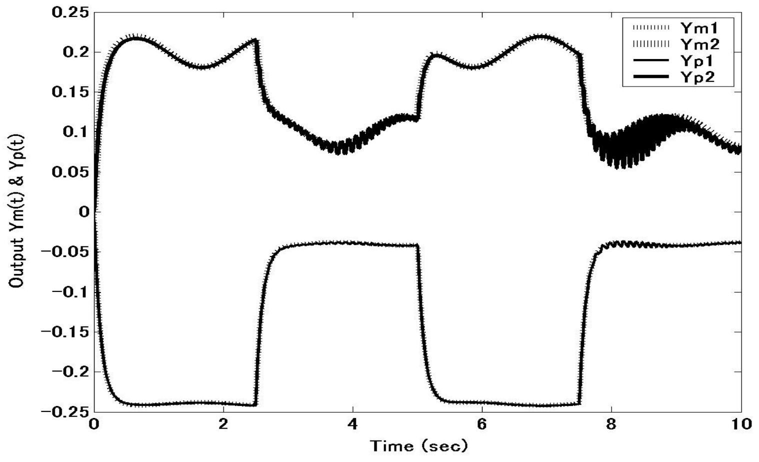

Figure 5.5: $y_m(t)$ and $y_p(t)$ using only SAC (MIMO system)

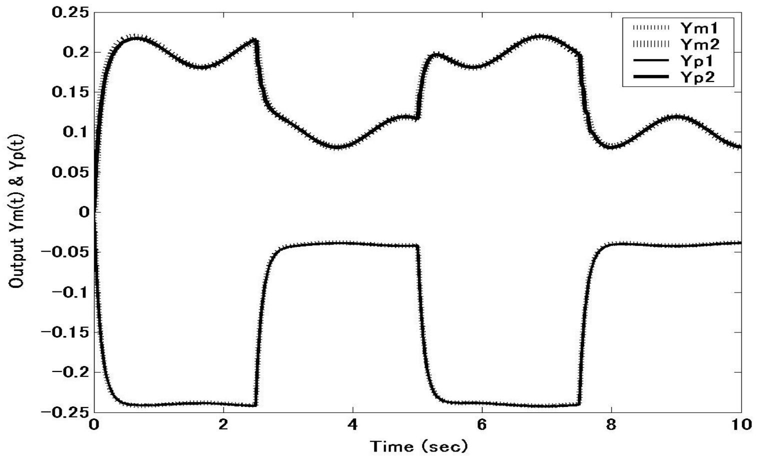

Figure 5.6: $y_m(t)$ and $y_p(t)$ using adaptive SMC with SAC (MIMO system)

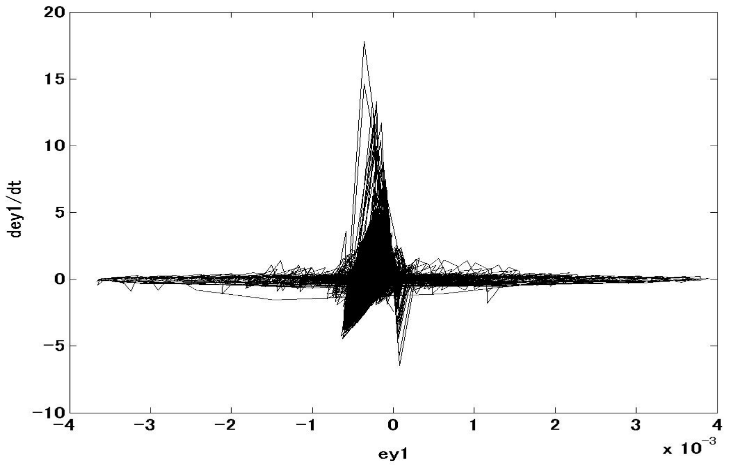

Figure 5.7: e_{y_1} and $\dot{e}_{y_1}$ using only SAC (MIMO system)

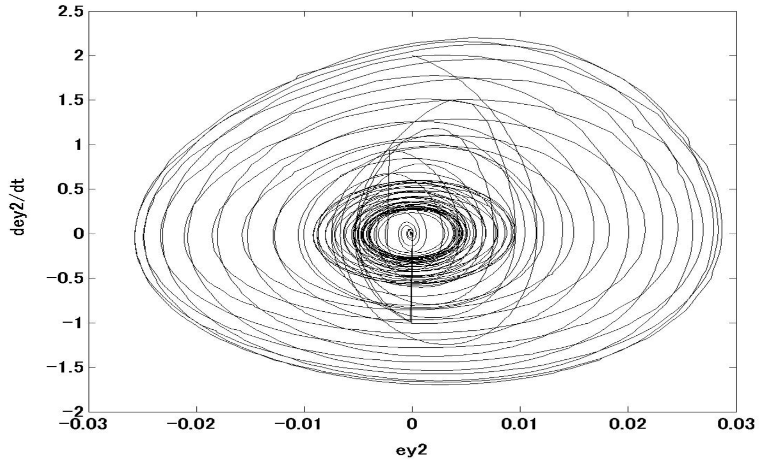

Figure 5.8: e_{y_2} and $\dot{e}_{y_2}$ using only SAC (MIMO system)

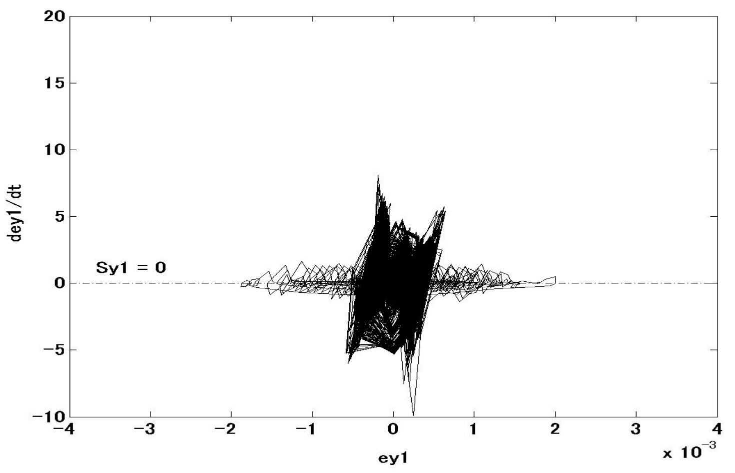

Figure 5.9: e_{y_1} and $\dot{e}_{y_1}$ using adaptive SMC with SAC (MIMO system)

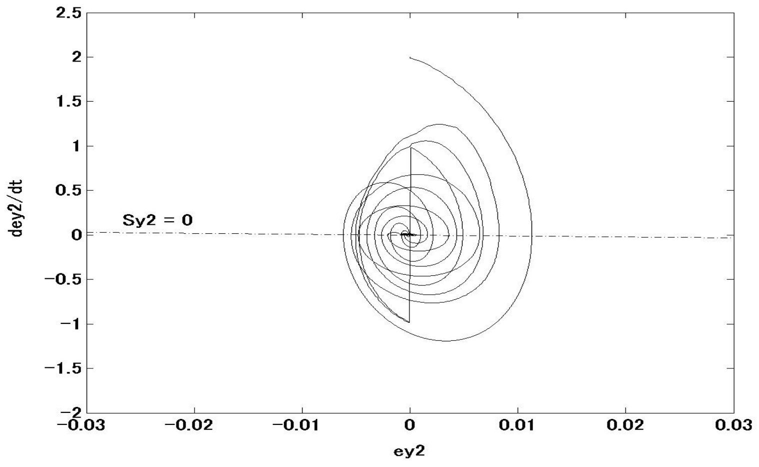

Figure 5.10: e_{y_2} and $\dot{e}_{y_2}$ using adaptive SMC with SAC (MIMO system)

Appendix 5A

Development of The Derivative of Lyapunov Function (5.30) from (5.27) in Lemma 5-1

Following [13], let (5.27) redescribed as

$$V_{SAC}(t) = V_{SAC_1}(t) + V_{SAC_2}(t) \tag{5.43}$$

where

$$\begin{aligned} V_{SAC_1}(t) &= e_x^T(t)Pe_x(t) && (5.44)\\ V_{SAC_2}(t) &= tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}\left[K_i(t)-\tilde{K}\right]^T\right\}. && (5.45)\end{aligned}$$

Then

$$\dot{V}_{SAC_1}(t) = \dot{e}_x^T(t)Pe_x(t) + e_x^T(t)P\dot{e}_x(t). \tag{5.46}$$

According to [13], $\dot{e}_x(t)$ can be expanded as

$$\dot{e}_x(t) = (A - B\tilde{K}_eC)e_x(t) - B\left[K(t)-\tilde{K}\right]r(t) - F(t). \tag{5.47}$$

Substituting $\dot{e}_x$ from (5.47), (5.46) becomes

$$\begin{aligned} \dot{V}_{SAC_1}(t) &= e_x^T(t)(P(A-B\tilde{K}_eC) + (A-B\tilde{K}_eC)^TP)e_x(t)\\ &\quad -2e_x^T(t)PB(K(t)-\tilde{K})r(t) - 2e_x^T(t)PF(t). && (5.48)\end{aligned}$$

Applying the positive real properties gives:

$$\begin{aligned} \dot{V}_{SAC_1}(t) &= -e_x^T(t)Qe_x(t) - 2e_x^T(t)C^T(K(t)-\tilde{K})r(t) - 2e_x^T(t)PF(t) && (5.49)\\ \dot{V}_{SAC_2}(t) &= 2tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}\dot{K}_i^T(t)\right\}. && (5.50)\end{aligned}$$

or, substituting $\dot{K}_i^T(t)$ from (2.17), (5.50) becomes

$$\dot{V}_{SAC_2}(t) = 2tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}(e_y(t)r^T(t)T_i - \sigma K_i(t))^T\right\} \tag{5.51}$$

and thus,

$$\begin{aligned}\dot{V}_{SAC_2}(t) &= -2\sigma tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}K_i^T(t)\right\}\\ &\quad +2e_y^T(t)(K_i(t)-\tilde{K})r(t). \end{aligned} \tag{5.52}$$

Substituting in the second term of (5.52)

$$K_i(t) = K(t) - K_p(t) = K(t) - e_y(t)r^T(t)T_p \tag{5.53}$$

gives

$$\begin{aligned}\dot{V}_{SAC_2}(t) &= -2\sigma tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}K_i^T(t)\right\} + 2e_y^T(t)(K(t)-\tilde{K})r(t)\\ &\quad -2e_y^T(t)e_y(t)r^T(t)T_pr(t). \end{aligned} \tag{5.54}$$

According to [13], $e_y(t)$ can be defined as

$$e_y(t) = y_m(t) - y_a(t) = \hat{y}_a(t) - y_a(t) \tag{5.55}$$

where $\hat{y}_a(t)$ is the unknown ideal augmented plant output defined as

$$\hat{y}_a(t) = C\hat{x}(t) = C_m x_m(t) = y_m(t). \tag{5.56}$$

Thus, (5.55) becomes

$$e_y(t) = Ce_x(t) - \hat{\delta}_o(x(t)). \tag{5.57}$$

Adding and subtracting $2\sigma tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}\tilde{K}^T\right\}$ and substituting $e_y(t)$ from (5.57) into the second right-hand term of (5.54) gives

$$\begin{aligned}\dot{V}_{SAC_2}(t) &= -2\sigma tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}\left[K_i(t)-\tilde{K}\right]^T\right\}\\ &\quad -2\sigma tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}\tilde{K}^T\right\}\\ &\quad -2e_y^T(t)e_y(t)r^T(t)T_pr(t) + 2e_x^T(t)C^T(K(t)-\tilde{K})r(t)\\ &\quad -2\hat{\delta}_o(x(t))(K(t)-\tilde{K})r(t). \end{aligned} \tag{5.58}$$

Adding (5.49) and (5.58) gives

$$\begin{aligned}\dot{V}_{SAC}(t) &= -e_x^T(t)Qe_x(t) - 2\sigma tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}\left[K_i(t)-\tilde{K}\right]^T\right\}\\ &\quad -2e_y^T(t)e_y(t)r^T(t)T_pr(t) - 2\sigma tr\left\{\left[K_i(t)-\tilde{K}\right]T_i^{-1}\tilde{K}^T\right\}\\ &\quad -2e_x^T(t)PF(t) - 2\hat{\delta}_o(x(t))(K(t)-\tilde{K})r(t). \end{aligned} \tag{5.59}$$

Then, by subsituting $K(t) = K_i(t) + K_p(t) = K_i(t) + e_y(t)r^T(t)T_p$ and $r^T(t)T_pr(t) = e_y^T(t)T_{p_{e_y}}e_y(t) + x_m^T(t)T_{p_{x_m}}x_m(t) + u_m^T(t)T_{p_{u_m}}u_m$, we can obtain (5.30).

Appendix 5B

Proof of the relation between $S_x(t)$ and $S_y(t)$ in (5.34)

We start by writing (5.28) as follows

$$e_x(t) = \begin{bmatrix} e_x(t) \\ \dot{e}_x(t) \\ \vdots \\ \overset{n_p}{e_x}(t) \end{bmatrix} = \begin{bmatrix} \hat{x}_1(t) - x_1(t) \\ \vdots \\ \hat{x}_{n_p+1}(t) - x_{n_p+1}(t) \end{bmatrix}. \tag{5.60}$$

Then, we write (5.18) as follows

$$\bar{e}_{y_j}(t) = \begin{bmatrix} e_{y_j}(t) \\ \dot{e}_{y_j}(t) \end{bmatrix} = \begin{bmatrix} y_{m_j}(t) - y_{a_j}(t) \\ \dot{y}_{m_j}(t) - \dot{y}_{a_j}(t) \end{bmatrix}. \tag{5.61}$$

where $(j = 1, \cdots, n_j)$. We consider that

$$C = \begin{bmatrix} C_1 \\ \vdots \\ C_{n_j} \end{bmatrix}. \tag{5.62}$$

where $C_j = \left[C_{j_1}, \cdots, C_{j_{n_p+1}}\right]$ and $(j = 1, \cdots, n_j)$. Thus, by applying (5.57) and (5.62) to (5.61), we can obtain

$$\begin{aligned} \bar{e}_{y_j}(t) &= \begin{bmatrix} C_j(\hat{x}(t) - x(t)) \\ C_j(\dot{\hat{x}}(t) - \dot{x}(t)) \end{bmatrix} - \begin{bmatrix} \delta_{oj}(x_p(t)) \\ \dot{\delta}_{oj}(x_p(t)) \end{bmatrix} \\ &= \begin{bmatrix} C_j e_x(t) \\ C_j \dot{e}_x(t) \end{bmatrix} - \begin{bmatrix} \delta_{oj}(x_p(t)) \\ \dot{\delta}_{oj}(x_p(t)) \end{bmatrix} \end{aligned} \tag{5.63}$$

where $(j = 1, \cdots, n_j)$.

We apply (5.63) to $S_{y_j}(t)$ in (5.17) $(j = 1, \cdots, n_j)$. Thus, $S_{y_j}(t)$ can be described as

$$\begin{aligned}
S_{y_j}(t) &= c_{y_j}^T \bar{e}_{y_j}(t) \\
&= c_{y_j}^T \begin{bmatrix} C_j e_x(t) \\ C_j \dot{e}_x(t) \end{bmatrix} - c_{y_j}^T \begin{bmatrix} \delta_{oj}(x_p(t)) \\ \dot{\delta}_{oj}(x_p(t)) \end{bmatrix} \\
&= \begin{bmatrix} c_{y_{j_1}}, c_{y_{j_2}} \end{bmatrix} \begin{bmatrix} C_{j_1} e_x(t) + C_{j_2} \dot{e}_x(t) + \cdots + C_{j_{n_p+1}} \overset{n_p}{e_x}(t) \\ C_{j_1} \dot{e}_x(t) + C_{j_2} \ddot{e}_x(t) + \cdots + C_{j_{n_p+1}} \overset{n_p+1}{e_x}(t) \end{bmatrix} \\
&\quad - c_{y_j}^T \begin{bmatrix} \delta_{oj}(x_p(t)) \\ \dot{\delta}_{oj}(x_p(t)) \end{bmatrix}. \qquad (5.64)
\end{aligned}$$

From (5.64), by assuming that $\overset{n}{e_x}(t) \cong 0$ for $n > n_p$, we can obtain (5.34), where c_{xj} of $S_{xj}(t)$ in (5.32) can be approximated as follows

$$\begin{aligned}
c_{xj_1} &\cong c_{yj_1} C_{j_1} \\
c_{xj_2} &\cong c_{yj_1} C_{j_2} + c_{yj_2} C_{j_1} \\
&\vdots \\
c_{xj_{n_p+1}} &\cong c_{yj_1} C_{j_{n_p+1}} + c_{yj_2} C_{j_{n_p}}. \qquad (5.65)
\end{aligned}$$

Chapter 6

Conclusions

SAC procedure was developed by Sobel et al. as an attempt to simplify adaptive controllers, since no observers or identifiers are needed in the feedback loop. Furthermore, the reference model is allowed to be of a very low order compared to the controlled plant. For linear plants with unknown structures, SAC is an important class of adaptive control schemes. However, for nonlinear plants with unknown structures, it is difficult to ensure a perfect plant output that follows the output of a reference model by using the conventional SAC.

In this thesis, chapter 2 proposed a fundamental of a method of SAC using a neural network for a class of nonlinear systems with BIBO and bounded nonlinearity. The convergence and stability analysis of the proposed method was performed, and it showed the class of the nonlinear plant and the boundary where the convergence and stability of the proposed method could be guaranteed. Chapter 3 proposed a control method using a discrete-time SAC with neural network for SISO and MIMO configurations of a nonlinear magnetic levitation system. Furthermore, in this chapter, the magnetic levitation system was set to satisfy the assumptions required by chapter 2; thus, the stability analysis in chapter 2 could be applied. Chapter 4 proposed a control method for nonlinear systems using SAC with multiple neural networks. In this chapter, a thorough theoretical explanation of convergence and stability analysis for this proposed method was also presented and discussed. The convergence and stability analysis showed that stability can be guaranteed if the class of nonlinear systems was the one with BIBO and bounded nonlinearities and if a certain boundary as in chapter 2 is satisfied. For applications of the method proposed in this chapter 4 to real nonlinear systems, each parallel small-scale neural network in our method could be developed on one dedicated circuit or microchip. Thus, it was obvious that the calculation process time required for each training iteration of the multiple neural networks could be kept the same as using only one small-scale neural network. In those methods, the control input was given by the sum of the output of the simple adaptive controller and the output of the neural networks. The neural networks were used to compensate for the nonlinearity of the plant dynamics that was not taken into consideration in the usual SAC. The role of the neural networks was to construct a linearized model by minimizing the output error caused by the nonlinearities in the control systems. Computer simulations in chapters 2 and 4 and experiments in chapter 3 were executed, and the effectiveness of these control methods was confirmed. Furthermore, it can be seen that the methods in chapters 2–4 could be used to deal well with:

1. the difficulties in the conventional SAC to control nonlinear plants with unknown structures,
2. the choice of complexity and time consuming processes of the neural networks,
3. the convergences and stabilities of the control systems (related to the complexity of the neural networks which have been reduced in our methods),
4. and applications to a real nonlinear plant of magnetic levitation system.

In chapter 5, a new method of adaptive SMC strategy using SAC for nonlinear systems with unknown parameters and dynamics was proposed. In this proposed method, the role of SAC was to construct an equivalent control input of adaptive SMC. To construct a corrective control input, this chapter applied a method using the sign function with a modified sliding surface. Furthermore, the stability analysis of the proposed method was performed. The stability analysis showed that stability could be guaranteed for the class of nonlinear systems with BIBO and bounded nonlinearities. Computer simulations were executed, and the effectiveness of this control method was confirmed. Computer simulations also showed that the proposed control method could keep the chattering phenomenon minimal. Thus, it can be seen that this method was able to deal well with:

1. the difficulties in the conventional SAC to control nonlinear plants with unknown structures,
2. the difficulties in the conventional SMC caused by the requirement of thorough knowledge of the controlled plant parameters and dynamics,

3. the complexity and time consuming processes in the existing methods of SMC using intelligent techniques,
4. and the stability of the control system.

Publication List

Journal Papers (with Review)

1. **Muhammad Yasser**, Takashi Yahagi, Agus Trisanto, Ayman Haggag, and Jianming Lu, "A control method for nonlinear systems using simple adaptive control with multiple neural networks," *Journal of Signal Processing*, Japan. **(Accepted)**

2. **Muhammad Yasser**, Agus Trisanto, Ayman Haggag, Jianming Lu, and Takashi Yahagi, "A control method for nonlinear magnetic levitation systems using simple adaptive control with neural networks," *Journal of Signal Processing*, Vol. 11, No. 6, pp. 497–510, 2007.

3. Agus Trisanto, **Muhammad Yasser**, Ayman Haggag, Jianming Lu, and Takashi Yahagi, "Application of neural networks to MRAC for the nonlinear magnetic levitation system," *JSME International Journal Series C: Dynamics Control Robotics Design & Manufacturing*, Vol. 49, No. 4, pp. 1073–1083, 2006.

4. Agus Trisanto, **Muhammad Yasser**, Ayman Haggag, Jianming Lu, and Takashi Yahagi, "Discrete time PID controller with neural network for magnetic levitation system," *Journal of Signal Processing*, Vol. 10, No. 6, pp. 481–489, 2006.

5. **Muhammad Yasser**, Agus Trisanto, Jianming Lu, and Takashi Yahagi, "A method of simple adaptive control for nonlinear systems using neural networks," *IEICE Trans. Fundamentals*, Vol. E89-A, No. 7, pp. 2009-2018, 2006.

6. Jianming Lu, **Muhammad Yasser**, Jiunshian Phuah, and Takashi Yahagi, "Simple adaptive control for MIMO nonlinear continuous-time systems using neural networks," 電気学会論文誌 *C*, Vol. 124, No. 8, pp. 1599–1605, August 2004.

International Conference Papers (with Review)

1. **Muhammad Yasser**, Hiroo Sekiya, Takashi Yahagi, Ayman Haggag, Mohamed Ghoneim, and Jianming Lu, “Adaptive Sliding Mode Control with Simple Adaptive Control for Buck Converters”, *Proceedings of The RISP 2008 International Workshop on Nonlinear Circuits and Signal Processing (NCSP08)*, Gold Coast, Australia, March 2008. **(Accepted)**

2. **Muhammad Yasser**, Agus Trisanto, Ayman Haggag, Takashi Yahagi, Hiroo Sekiya, and Jianming Lu, “Simple adaptive control for SISO nonlinear systems using multiple neural networks”, *Proceedings of The International Conference on Instrumentation, Control and Information Technology (SICE Annual Conference 2007)*, pp.1287–1292, Takamatsu, Japan, September 2007.

3. **Muhammad Yasser**, Agus Trisanto, Ayman Haggag, Takashi Yahagi, Hiroo Sekiya, and Jianming Lu, “Sliding mode control using neural networks for SISO nonlinear systems”, *Proceedings of The International Conference on Instrumentation, Control and Information Technology (SICE Annual Conference 2007)*, pp.980–984, Takamatsu, Japan, September 2007.

4. **Muhammad Yasser**, Agus Trisanto, Ayman Haggag, Jianming Lu, Hiroo Sekiya, and Takashi Yahagi, “An adaptive sliding mode control using simple adaptive control for a class of SISO nonlinear systems with bounded-input bounded-output and bounded nonlinearity,” *Proceedings of The 45th IEEE Conference on Decision and Control (45th CDC)*, pp.1599–1604, San Diego, USA, December 2006.

5. Agus Trisanto, **Muhammad Yasser**, Jianming Lu, and Takashi Yahagi, “Implementation of a fuzzy PID controller using neural network on the magnetic levitation systems,” *Proceedings of The 2006 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2006)*, pp.669–672, Tottori, Japan, December 2006.

6. Agus Trisanto, **Muhammad Yasser**, Ayman Haggag, Jianming Lu, and Takashi Yahagi, “Performance improvement of discrete time PID controller using neural network,” *Proceedings of The Joint 3rd International Conference on Soft Computing and 7th Intelligent Symposium on Advanced Intelligent Systems (SCIS & ISIS 2006)*, pp.35–39, Tokyo, Japan, September 2006.

7. **Muhammad Yasser**, Agus Trisanto, Ayman Haggag, Jianming Lu, Hiroo Sekiya, and Takashi Yahagi, “An application of adaptive sliding mode control using simple adaptive control for SISO magnetic levitation system,” *Proceedings of The Indonesia-Japan Joint Scientific Symposium 2006 (IJJSS 2006)*, Depok, Indonesia, September 2006.

8. Agus Trisanto, **Muhammad Yasser**, Jianming Lu, and Takashi Yahagi, "Application of neural network to MRAC for MIMO magnetic levitation systems," *Proceedings of The Indonesia-Japan Joint Scientific Symposium 2006 (IJJSS 2006)*, Depok, Indonesia, September 2006.

9. **Muhammad Yasser**, Agus Trisanto, Jianming Lu, Hiroo Sekiya, and Takashi Yahagi, "Adaptive sliding mode control using simple adaptive control for SISO nonlinear systems," *Proceedings of The 2006 IEEE International Symposium on Circuits and Systems (ISCAS 2006)*, pp.2153–2156, Kos Island, Greece, May 2006.

10. **Muhammad Yasser**, Jiunshian Phuah, Jianming Lu, and Takashi Yahagi, "Combination of simple adaptive control method and neural networks for MIMO nonlinear magnetic levitation system," *Proceedings of The 2005 IEEE International Workshop on Nonlinear Signal and Image Processing (NSIP 2005)*, vol.1, pp. 351–356, Sapporo, Japan, May 2005.

11. Jiunshian Phuah, Jianming Lu, **Muhammad Yasser**, and Takashi Yahagi, "Neuro-sliding mode control for magnetic levitation systems," *Proceedings of The 2005 IEEE International Symposium on Circuits and Systems (ISCAS 2005)*,pp.5130–5133, Kobe, Japan, May 2005.

12. **Muhammad Yasser**, Jiunshian Phuah, Jianming Lu, and Takashi Yahagi, "Simple adaptive control for SISO nonlinear system using neural networks for magnetic levitation plant," *Proceedings of The 47th 2004 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS 2004)*, pp. III-113–III-116, Hiroshima, Japan, July 2004.

13. **Muhammad Yasser**, Jiunshian Phuah, Jianming Lu, and Takashi Yahagi, "A method of simple adaptive control for MIMO nonlinear continuous-time systems using multifraction neural network," *Proceedings of The 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, vol. 1, pp. 23–28, Kobe, Japan, July 2003.

Domestic Conference Papers (with Review)

1. **Muhammad Yasser**, Jiunshian Phuah, Jianming Lu, and Takashi Yahagi, “Simple adaptive control for MIMO nonlinear continuous-time systems using neural networks,” *Proceedings of* 第 *17*回ディジタル信号処理シンポジウム講演論文集 *CD-ROM*, A4-3, Hakodate, Japan, November 2002.

Curriculum Vitae

Muhammad Yasser was born in Jakarta, Republic of Indonesia, on January 19, 1978. He received the B.E. degree from the University of Indonesia, Indonesia, and the M.E. degree from Chiba University, Japan, in 2001 and 2005, respectively. In April 2005, he joined the Graduate School of Science and Technology of Chiba University as a Ph.D. student. His current research interests are in the theory and applications of adaptive control, sliding-mode control, and neural networks, and also in the theory and applications of control for power electronics. He is a student member of IEEE (USA), IET (UK), IEICE (Japan), IEEJ (Japan), and Research Institute of Signal Processing (Japan).

References

[1] K.S. Narendra and R.V. Monopoli, Applications of Adaptive Control, Academic Press, 1980.

[2] Ch.I. Byrnes and A. Kurzhanski (Eds.), Modelling and Adaptive Control, Proceedings of the IISA Conference Sopron, Hungary, July 1986, Springer-Verlag, 1988.

[3] W.T. Miller, III, R.S. Sutton, and P.J. Werbos, Neural Networks for Control, The MIT Press, 1990.

[4] K. Warwick, G.W. Irwin, and K.J. Hunt, Neural Network for Control and Systems, IEE Control Engineering series 46, Peter Peregrinus Ltd., 1992.

[5] J. Lu and T. Yahagi, "New design method for model reference adaptive control for nonminimum phase discrete-time systems with disturbances," *IEE Proceeding-D, Control Theory and Applications*, Vol. 140, No. 1, pp. 34–40, 1993.

[6] J.M. Mendel, A Prelude to Neural Networks: Adaptive and Learning Systems, PTR Prentice Hall, 1994.

[7] K.J. Åström and B. Wittenmark, Adaptive Control (2nd ed.), Addison-Wesley, 1995.

[8] M. Krsti'c, I. Kanellakopoulos, and P. Kokotovi'c, Nonlinear and Adaptive Control Design, John Wiley & Sons, Inc., 1995.

[9] J.A.K. Suykens, J.P.L. Vandewalle, and B.L.R. De Moor, Artificial Neural Networks for Modelling and Control of Non-Linear Systems, Kluwer Academic Publishers, 1996.

[10] R. Zbikowski and K.J. Hunt, Neural Adaptive Control Technology, World Scientific, 1996.

[11] S. Omatu, M. Khalid, and R. Yusof, Neuro-Control and Its Applications, Advances in Industrial Control, Springer, 1996.

[12] C. Edwards and S.K. Spurgeon, Sliding Mode Control, Theory and Applications, Taylor & Francis, 1998.

[13] H. Kaufman, I. Bar-Kana, and K. Sobel, Direct Adaptive Control Algorithms, Theory and Applications (2nd ed.), Springer, 1998.

[14] J. Lu and T. Yahagi, "Discrete-time model reference adaptive control for nonminimum phase systems with disturbances," *Trans. ASME, Journal of Dynamic Systems, Measurement, and Control*, Vol. 120, No. 3, pp. 117–123, 1998.

[15] K.D. Young and U. Ozguner (Eds.), Variable Structure Systems, Sliding Mode and Nonlinear Control, Lecture Notes in Control and Information Sciences 247, Springer, 1999.

[16] H. Khalil, Nonlinear Systems, Third Edition, Prentice Hall, Inc., 2000.

[17] W. M. Haddad and T. Hayakawa, "Direct adaptive control for non-linear uncertain systems with exogenous disturbances," *Int. J. Adapt. Control Signal Processing*, Vol. 16, pp. 151-172, 2002.

[18] W. Perruquetti and J.P. Barbot, Sliding Mode Control in Engineering, Control Engineering Series, Marcel Dekker, Inc., 2002.

[19] K. Sobel, H. Kaufman, and L. Mabius, "Model reference output adaptive control systems without parameter identification," *Proc. of the* 18^{th} *IEEE Conference on Decision and Control (*18^{th} *CDC)*, pp. 347–351, 1979.

[20] K. Sobel, H. Kaufman, and L. Mabius, "Implicit adaptive control for a class of MIMO systems," *IEEE Aerospace Electron Syst.*, Vol. AES-18, No. 5, pp. 576–590, 1982.

[21] J. Lu, J. Phuah, and T. Yahagi, "SAC for nonlinear systems using Elman recurrent neural networks," *IEICE Trans. Fundamentals*, Vol. E85-A, No. 8, pp. 1831–1840, 2002.

[22] J. Lu, M. Yasser, J. Phuah, and T. Yahagi, "Simple Adaptive Control for MIMO Nonlinear Continuous-Time Systems Using Neural Network," *Trans. IEE of Japan*, Vol. 124-C, No. 8, pp. 1599–1605, 2004.

[23] K.S. Narendra and L.S. Valavani, "Stable adaptive controller design-direct control," *IEEE Trans. On Automatic Control*, Vol. AC-23, No. 4, pp. 570–583, 1978.

[24] K.S. Narendra, Y.H. Lin, and L.S. Valavani, "Stable adaptive controller design, part II: proof of stability," *IEEE Trans. On Automatic Control*, Vol. AC-25, No. 3, pp. 440–448, 1980.

[25] K.S. Narendra and Y.H. Lin, “Stable discrete adaptive control,” *IEEE Trans. On Automatic Control*, Vol. AC-25, No. 3, pp. 456–461, 1980.

[26] H. Wu, J. Hu, and Y. Xie, “Characteristic model-based all-coefficient adaptive control method and its applications,” *IEEE Trans. Syst., Man, and Cyber.-Part C*, Vol. 37, No. 2, pp. 213-221, 2007.

[27] K. Narendra and L. Valavani, “Direct and indirect model reference adaptive control,” *Automatica*, Vol. 15, pp. 653–664, November 1979.

[28] I. Landau, “A survey of model reference adaptive technique: Theory and applications,” *Automatica*, Vol. 10, pp. 353–379, 1974.

[29] R. Monopoli, “Model reference adaptive control with an augmented error signal,” *IEEE Trans. On Automatic Control*, Vol. AC-19, No. 5, pp. 474–484, 1974.

[30] A. Morse, “Global stability of parameter adaptive control systems,” *IEEE Trans. On Automatic Control*, Vol. AC-25, pp. 433–439, June 1980.

[31] A. Feuer and A. Morse, “Adaptive control of single-input, single-output linear systems,” *IEEE Trans. On Automatic Control*, Vol. AC-23, pp. 557–569, August 1978.

[32] C. Rohrs, L. Valavani, M. Athans, and G. Stein, ”Robustnes of continuous time adaptive control algorithms in the presence of unmodeled dynamics,” *IEEE Trans. On Automatic Control*, Vol. AC-30, pp. 881–889, September 1985.

[33] D. Lindorff and R. Carroll, “Survey of adaptive control using liapunov design,” *Int. J. Control*, Vol. 18, No. 5, pp. 897–914, 1973.

[34] I. Bar-Kana and H. Kaufman, “Global stability and performance of a simplified adaptive algorithm,” *Int. J. Control*, Vol. 42, No. 6, pp. 1491–1505, 1985.

[35] I. Bar-Kana and H. Kaufman, “Technical Note: Simple adaptive control of uncertain systems,” *Int. J. Adaptive Control and Signal Processing*, Vol. 2, pp. 133–143, 1988.

[36] Z. Iwai and I. Mizumoto, “Robust and simple adaptive control systems,” *Int. J. Control*, Vol. 55, No. 6, pp. 1453–1470, 1992.

[37] S. Haykin, Neural Networks, A Comprehensive Foundation, IEEE Coumputer Press, IEEE Press, Macmillan College Publishing Company, Inc., 1994.

[38] D.H. Nguyen and B. Widrow, "Neural networks for self-learning control systems," *IEEE Contr. Syst. Mag.*, pp. 18–23, April 1990.

[39] G.L. Plett, "Adaptive inverse control of linear and nonlinear systems using dynamic neural networks," *IEEE Trans. Neural Networks*, Vol. 14, No. 2, pp. 360–376, 2003.

[40] G.A. Rovithakis and M.A. Christodoulou, "Adaptive control of unknown plants using dynamical neural networks," *IEEE Trans. Syst., Man, and Cyber.*, Vol. 24, No. 3, pp. 400-412, 1994.

[41] C.C. Ku and K.Y. Lee, "Dynamics recurrent neural networks for dynamic systems control," *IEEE Trans. Neural Networks*, Vol. 6, No. 1, pp. 144–156, 1995.

[42] Y.C. Chen and C.C. Teng, "A model reference control using a fuzzy neural network," *ELSEVIER Fuzzy Sets and Systems 73*, pp. 291–312, 1995.

[43] S. Jagannathan, F.L. Lewis, and O. Pastravanu, "Discrete-time model reference adaptive control of nonlinear dynamical systems using neural networks," *Int. J. Control*, Vol. 64, No. 2, pp. 217–239, 1996.

[44] G.A. Rovithakis and M.A. Christodoulou, "Neural adaptive regulation of unknown nonlinear dynamical systems," *IEEE Trans. Syst., Man, and Cyber.-Part B*, Vol. 27, No. 5, pp. 810-822, 1997.

[45] K.S. Narendra and S. Mukhopadhyay, "Adaptive control using neural networks and approximate models," *IEEE Trans. Neural Networks*, Vol. 8, No. 3, pp. 475–485, 1997.

[46] G.P. Liu, V. Kadirkamanathan, and S.A. Billings, "Variable neural networks for adaptive control of nonlinear systems," *IEEE Trans. Syst., Man, and Cyber.-Part C*, Vol. 29, No. 1, pp. 34-43, 1999.

[47] M.A. Brdys and G.J. Kulawski, "Dynamic neural controllers for induction motor," *IEEE Trans. Neural Networks*, Vol. 10, No. 2, pp. 340–355, 1999.

[48] Y. Yi, D.M. Vilathgamuwa, and M.A. Rahman, "Implementation of an artificial-neural-network-based real-time adaptive controller for an interior permanent-magnet motor drive," *IEEE Trans. Ind. Applicat.*, Vol. 39, No. 1, pp. 96–104, 2003.

[49] H. Qiao, J. Peng, Z.B. Xu, and B. Zhang, "A reference model approach to stability analysis of neural networks," *IEEE Trans. Syst., Man, and Cyber.-Part B*, Vol. 33, No. 6, pp. 925-936, 2003.

[50] G.A. Rovithakis, "Robust redesign of a neural network controller in the presence of unmodeled dynamics," *IEEE Trans. Neural Networks*, Vol. 15, No. 6, pp. 1482–1490, 2004.

[51] Q. Zhu and L. Guo, "Stable adaptive neurocontrol for nonlinear discrete-time systems," *IEEE Trans. Neural Networks*, Vol. 15, No. 3, pp. 653–661, 2004.

[52] S.S. Ge and C. Wang, "Adaptive neural control of uncertain MIMO nonlinear systems," *IEEE Trans. Neural Networks*, Vol. 15, No. 3, pp. 674–692, 2004.

[53] V.I. Utkin, "Variable structure systems with sliding mode," *IEEE Trans. Automat. Contr.*, vol. 22, pp. 212–222, 1977.

[54] H. N. Iordanou and B. W. Surgenor, "Experimental evaluation of the robustness of discrete sliding mode control versus linear quadratic control," *IEEE Trans. Contr. Syst. Tech.*, vol. 5, pp. 254-260, 1997.

[55] J.E. Slotine and S.S. Sastry, "Tracking control of nonlinear systems using sliding surface with application to robotic manipulators," *Int. J. Contr.*, Vol. 38, No. 2, pp. 465–492, 1983.

[56] J.E. Slotine, *1 "Sliding controller design for nonlinear systems," *Int. J. Contr.*, Vol. 40, No. 2, pp. 421–434, 1984.

[57] L. C. Fu and T. L. Liao, "Globally stable robust tracking of nonlinear systems using variable structure control and with an application to a robotic manipulator," *IEEE Trans. Automat. Contr.*, Vol. 35, No. 12, pp. 1345–1350, 1990.

[58] A. Levant, "Sliding order and sliding accuracy in sliding mode control," *Int. J. Contr.*, Vol. 58, No. 6, pp. 1247–1263, 1993.

[59] J. Ackermann and V.I. Utkin, "Sliding mode control design based on Ackermann's formula," *Proc. of the* 33^{th} *IEEE Conference on Decision and Control (*33^{th} *CDC)*, pp. 3622–3627, 1994.

[60] J. Ackermann and V.I. Utkin, "Sliding mode control design based on Ackermann's formula," *IEEE Trans. Automat. Contr.*, Vol. 43, No. 2, pp. 234–237, 1998.

[61] G. Bartolini, A. Ferrara, E. Usai, and V.I. Utkin, "On multi-input chattering-free second-order sliding mode control," *IEEE Trans. Automat. Contr.*, Vol. 45, No. 9, pp. 1711–1717, 2000.

[62] M. Ertugrul and O. Kaynak, "Neuro-sliding mode control of robotic manipulators," *Mechatronics*, Vol. 10, pp. 239–263, 2000.

[63] S.H. Lee, K.W. Min, and Y.C. Lee, "Modified sliding mode control using a target derivative of the Lyapunov function," *ELSEVIER Engineering Structures 27*, pp. 49–59, 2000.

[64] S. Tong and H.X. Li, "Fuzzy adaptive sliding-mode control for MIMO nonlinear systems," *IEEE Trans. Fuzzy Syst.*, Vol. 11, No. 3, pp. 354–360, 2003.

[65] A. Levant, "High-order sliding modes, differentiation and output-feedback control," *Int. J. Contr.*, Vol. 76, Nos. 9/10, pp. 924–941, 2003.

[66] C.H. Tsai, H.Y. Chung, and F.M. Yu, "Neuro-sliding mode control with its applications to seesaw systems," *IEEE Trans. Neural Networks*, Vol. 15, No. 1, pp. 124–134, 2004.

[67] G. Bartolini, E. Punta, and T. Zolessi, "On multi-input chattering-free second-order sliding mode control," *IEEE Trans. Automat. Contr.*, Vol. 49, No. 6, pp. 922–933, 2004.

[68] M. A. Hussain and P. Y. Ho, "Adaptive sliding mode control with neural network based hybrid models," *J. Process Control*, Vol. 14, pp. 157-176, 2004.

[69] E.M. Jafarov, M.N.A. Parlakci, and Y. Istefanopulos, "Variable structure PID-controller design for robot manipulators," *IEEE Trans. Contr. Syst. Tech.*, Vol. 13, No.1, pp. 122–130, 2005.

[70] S.C. Tan, Y.M. Lai, M.K.H. Cheung, and C.K. Tse, "On the practical design of a sliding mode voltage controlled buck converter," *IEEE Trans. Power Electr.*, Vol. 20, No. 2, pp. 425–437, 2005.

[71] J. Phuah, J. Lu, and T. Yahagi, "Chattering free sliding mode control in magnetic levitation system," *IEEJ Trans. EIS*, Vol. 125, No. 4, pp. 600–606, 2005.

[72] S.C. Tan, Y.M. Lai, C.K. Tse, and M.K.H. Cheung, "A fixed-frequency pulsewidth modulation based quasi-sliding-mode controller for buck converters," *IEEE Trans. Power Electr.*, Vol. 20, No. 6, pp. 1379–1392, 2005.

[73] J. Phuah, J. Lu, M. Yasser, and T. Yahagi, "Neuro-sliding mode control for magnetic levitation systems," *Proc. of the 2005 IEEE International Symposium on Circuits and Systems (ISCAS 2005)*, pp. 5130–5133, 2005.

[74] M. Yasser, A. Trisanto, J. Lu, H. Sekiya, and T. Yahagi, "Adaptive sliding mode control using simple adaptive control for SISO nonlinear systems," *Proc. of the 2006 IEEE International Symposium on Circuits and Systems (ISCAS 2006)*, pp. 2153–2156, 2006.

[75] M. Yasser, A. Trisanto, J. Lu, H. Sekiya, and T. Yahagi, "An adaptive sliding mode control using simple adaptive control for a class of SISO nonlinear systems with bounded-input bounded-output and bounded nonlinearity," *Proc. of the* 45th *IEEE Conference on Decision and Control (*45th *CDC)*, pp. 1599–1604, 2006.

[76] M. Shafiq, J. Lu, and T. Yahagi, "A method for adaptive control of nonminimum phase continuous-time systems based on pole-zero placement," *IEICE Trans. Fundamentals*, Vol. E80-A, No. 6, pp. 1109–1115, 1997.

[77] F. Allgower, J. Ashman, and A. Ilchmann, "High-gain Adaptive λ-tracking for Nonlinear Systems," *Automatica*, Vol. 33, No. 5, pp. 881–888, 1997.

[78] I. Mizumoto, Z. Iwai, and K. Kohara, "Adaptive Output Feedback Control for MIMO Nonlinear Systems based on Feedback Exponential Passivity," *Trans. of SICE*, Vol. 37, No. 2, pp. 115–124, 2001.

[79] J. Lu, J. Phuah, and T. Yahagi, "A method of model reference adaptive control for MIMO nonlinear systems using neural networks," *IEICE Trans. Fundamentals*, Vol. E84-A, No. 8, pp. 1933–1941, 2001.

[80] X. Sun and M. Rao, "A multivariable bilinear adaptive controller with decoupling design," *IEEE Trans. On Automatic Control*, Vol. 45, No. 4, pp. 714–719, 2000.

[81] G.F. Franklin, J.D. Powell, and M.L. Workman, Digital Control of Dynamic Systems, Addison Wesley Longman, 1998.

[82] H. Zhang and Z. Bien, "Adaptive fuzzy control of MIMO nonlinear systems," *ELSEVIER Fuzzy Sets and Systems 115*, pp. 191–204, 2000.

[83] D. Cho, Y. Kato and D. Spilman, "Sliding mode and classical controllers in magnetic levitation system," *IEEE Control Syst. Mag.*, Vol. 13, pp. 42–48, 1993.

[84] M. Yasser, A. Trisanto, J. Lu and T. Yahagi, "A method of simple adaptive control for MIMO nonlinear systems using neural networks," *IEICE Trans. Fundamentals*, Vol. E89-A, No. 7, pp. 2009–2018, 2006.

[85] T.R. Parks, Manual for Model 730 Magnetic Levitation System, Educational Control Products, 1991.

[86] M. Yasser, A. Trisanto, A. Haggag, J. Lu, and T. Yahagi, "A control method for nonlinear magnetic levitation systems using simple adaptive control with neural networks," *Journal of Signal Processing*, Vol. 11, No. 6, pp. 497–510, 2007.

[87] F.C. Chen and H. Khalil, "adaptive control of a class of nonlinear discrete-time systems using neural networks," *IEEE Trans. Automat. Contr.*, Vol. 40, No. 5, pp. 791–801, 1995.

[88] M. Yasser, J. Phuah, J. Lu, and T. Yahagi, "A method of simple adaptive control for MIMO nonlinear continuous-time systems using multifraction neural network," *Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, vol. 1, pp. 23–28, 2003.

[89] M. Yasser, A. Trisanto, A. Haggag, J. Lu, and T. Yahagi, "Simple adaptive control for SISO nonlinear systems using multiple neural networks," *Proceedings of the International Conference on Instrumentation, Control and Information Technology (SICE Annual Conference 2007)*, pp. 1287–1292, 2007.